Guia do usuário

AWS Identity and Access Management

AWS Identity and Access Management: Guia do usuário

Table of Contents

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, de maneira centralizada, permissões que controlam quais recursos da AWS os usuários poderão acessar. Você usa o IAM para controlar quem é autenticado (fez login) e autorizado (tem permissões) a usar os recursos.

Ao criar uma Conta da AWS, você começa com uma identidade de login que tem acesso completo a todos os recursos e Serviços da AWS na conta. Essa identidade é denominada usuário raiz da Conta da AWS e é acessada por login com o endereço de e-mail e a senha que você usou para criar a conta. É altamente recomendável não usar o usuário raiz para tarefas diárias. Proteja as credenciais do usuário raiz e use-as para executar as tarefas que somente ele pode executar. Para obter a lista completa das tarefas que exigem fazer login como usuário raiz, consulte Tarefas que exigem credenciais de usuário raiz.

Conteúdo

Vídeo de introdução ao IAM

O AWS Training and Certification fornece um vídeo de introdução de dez minutos ao IAM:

Introdução à AWS Identity and Access Management

Recursos do IAM

O IAM oferece os seguintes recursos:

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 S3 ou permissão para administrar apenas algumas instâncias do 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 ter mais informações, consulte Configurações compatíveis com o uso de chaves de segurança FIDO.

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, obtenham acesso temporário à 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.

Integrados com muitos serviços da AWS

Para obter uma lista dos produtos da AWS que funcionam com o IAM, consulte Serviços da AWS que funcionam com o IAM.

Finalmente consistente

O IAM, como muitos outros produtos da AWS, oferece consistência final. O IAM atinge alta disponibilidade ao replicar dados em vários servidores dentro de datacenters 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 ter mais informações, consulte As alterações que eu faço nem sempre ficam imediatamente visíveis.

Uso gratuito

O AWS Identity and Access Management (IAM) e o AWS Security Token Service (AWS STS) são recursos 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. Para obter informações sobre preços de outros produtos da AWS, consulte a página de preços da Amazon Web Services.

Acesso ao IAM

Você pode trabalhar com o AWS Identity and Access Management de qualquer uma das seguintes formas.

AWS Management Console

O console é uma interface baseada em navegador para gerenciar recursos do IAM e da AWS. Para obter mais informações sobre como acessar o IAM pelo console, consulte Como fazer login na AWS no Guia do usuário do Início de Sessão da AWS.

Ferramentas de linha de comando da AWS

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 AWS Tools for Windows 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 Utilização da AWS CloudShell para operação com o AWS Identity and Access Management.

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.

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.

Quando posso usar o IAM?

Quando você executa diferentes funções de trabalho

O AWS Identity and Access Management 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 ajudá-lo a solicitar as permissões corretas ao seu 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. Assim, você deve enviar solicitações ao administrador do IAM para alterar as permissões dos usuários de seu serviço. Revise as informações nesta página para entender os Introdução ao 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 você faz login na AWS usando suas credenciais de identidade. É necessário ser autenticado (fazer login na AWS) como o usuário raiz da Usuário raiz da conta da AWS, como usuário do IAM ou assumindo um perfil do IAM.

Você pode fazer login na AWS como uma identidade federada usando credenciais fornecidas por uma fonte de identidades. Os usuários do AWS IAM Identity Center (IAM Identity Center), a autenticação única da empresa e as suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como uma identidade federada, o administrador já configurou anteriormente a federação de identidades utilizando perfis do IAM. Quando você acessa a AWS usando a federação, está indiretamente assumindo um perfil.

É possível fazer login no ou no portal de acesso da AWS Management Console dependendo do tipo de usuário que você é.AWS Para obter mais informações sobre como fazer login na AWS, consulte Como fazer login na conta da Conta da AWS no Guia do usuário do Início de Sessão da AWS.

Se você acessar a AWS programaticamente, a AWS fornecerá um kit de desenvolvimento de software (SDK) e uma interface da linha de comando (CLI) para você assinar criptograficamente as solicitações usando as suas credenciais. Se você não utilizar as ferramentas da AWS, deverá assinar as solicitações por conta própria. Para obter mais informações sobre como usar o método recomendado para assinar solicitações por conta própria, consulteAssinar AWSsolicitações de API da noGuia do usuário do IAM.

Independentemente do método de autenticação usado, também pode ser exigido que você forneça mais informações de segurança. Por exemplo, a AWS recomenda o uso da autenticação multifator (MFA) para aumentar a segurança de sua conta. Para saber mais, consulte Autenticação multifator no Guia do usuário do AWS IAM Identity Center e Usar a autenticação multifator (MFA) naAWS no Guia do usuário do IAM.

Quando você fizer login como 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. Sempre que possível, recomendamos contar com credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de utilização específicos que exijam credenciais de longo prazo com usuários do IAM, recomendamos alternar as chaves de acesso. Para obter mais informações, consulte Alterne as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo no Guia do usuário do IAM.

Um grupo do IAM é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível utilizar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdmins e atribuir a esse grupo permissões para administrar atributos do IAM.

Usuários são diferentes de perfis. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas um perfil pode ser assumido por qualquer pessoa que precisar dele. Os usuários têm credenciais permanentes de longo prazo, mas os perfis fornecem credenciais temporárias. Para saber mais, consulte Quando criar um usuário do IAM (em vez de uma função) no Guia do usuário do IAM.

Quando você assume um perfil do IAM

Um perfil do IAM é uma identidade dentro da Conta da AWS que tem permissões específicas. Ele é semelhante a um usuário do IAM, mas não está associado a uma pessoa específica. É possível assumir temporariamente um perfil do IAM no AWS Management Console alternando perfis. É possível assumir um perfil chamando uma operação de API da AWS CLI ou da AWS, ou usando um URL personalizado. Para obter mais informações sobre métodos para o uso de perfis, consulte Uso de funções do IAM no Guia do usuário do IAM.

Perfis do IAM com credenciais temporárias são úteis nas seguintes situações:

- Acesso de usuário federado: para atribuir permissões a identidades federadas, você pode criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas pelo mesmo. Para obter mais informações sobre perfis para federação, consulte Criar uma função para um provedor de identidade de terceiros no Guia do usuário do IAM. Se você usar o IAM Identity Center, deverá configurar um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o Centro de identidade do IAM correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte Conjuntos de permissões no Guia do usuário do AWS IAM Identity Center.
- Permissões temporárias para usuários do IAM: um usuário ou um perfil do IAM pode assumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso entre contas: é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, alguns Serviços da AWS permitem que você anexe uma política diretamente a um atributo (em vez de usar um perfil como proxy). Para saber a

diferença entre perfis e políticas baseadas em atributo para acesso entre contas, consulte Como os perfis do IAM diferem das políticas baseadas em atributo no Guia do usuário do IAM.

- Acesso entre serviços: alguns Serviços da AWS usam atributos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicações no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões do principal de chamada, usando um perfil de serviço ou um perfil vinculado ao serviço.
  - Sessões de Acesso Direto (FAS): ao utilizar um usuário ou perfil do IAM para realizar ações no AWS, você é considerado uma entidade principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. As FAS usam as permissões da entidade principal que chama umAWS service (Serviço da AWS), combinadas com o AWS service (Serviço da AWS) solicitante para fazer solicitações aos serviços downstream. As solicitações de FAS só são feitas quando um serviço recebe uma solicitação que requeira interações com outros Serviços da AWS ou com recursos da para ser atendida. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.
  - Perfil de serviço: um perfil de serviço é um perfil do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do usuário do IAM.
  - Perfil vinculado ao serviço: um perfil vinculado ao serviço é um tipo de perfil de serviço vinculado a uma AWS service (Serviço da AWS). O serviço pode assumir o perfil para executar uma ação em seu nome. Os perfis vinculados ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode exibir, mas não pode editar as permissões para perfis vinculados ao serviço.
- Aplicações em execução no Amazon EC2: é possível usar um perfil do IAM para gerenciar credenciais temporárias para aplicações em execução em uma instância do EC2 e fazer solicitações da AWS CLI ou da AWS API. É preferível fazer isso a armazenar chaves de acesso na instância do EC2. Para atribuir um perfil da AWS a uma instância do EC2 e disponibilizá-la para todas as suas aplicações, crie um perfil de instância que esteja anexado a ela. Um perfil de instância contém a perfil e permite que os programas em execução na instância do EC2 obtenham credenciais temporárias. Para mais informações, consulte Usar um perfil do IAM para conceder permissões a aplicações em execução nas instâncias do Amazon EC2 no Guia do usuário do IAM.

Para saber se deseja usar os perfis do IAM, consulte Quando criar um perfil do IAM (em vez de um usuário) 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).

Essas políticas são usadas com um perfil do IAM:

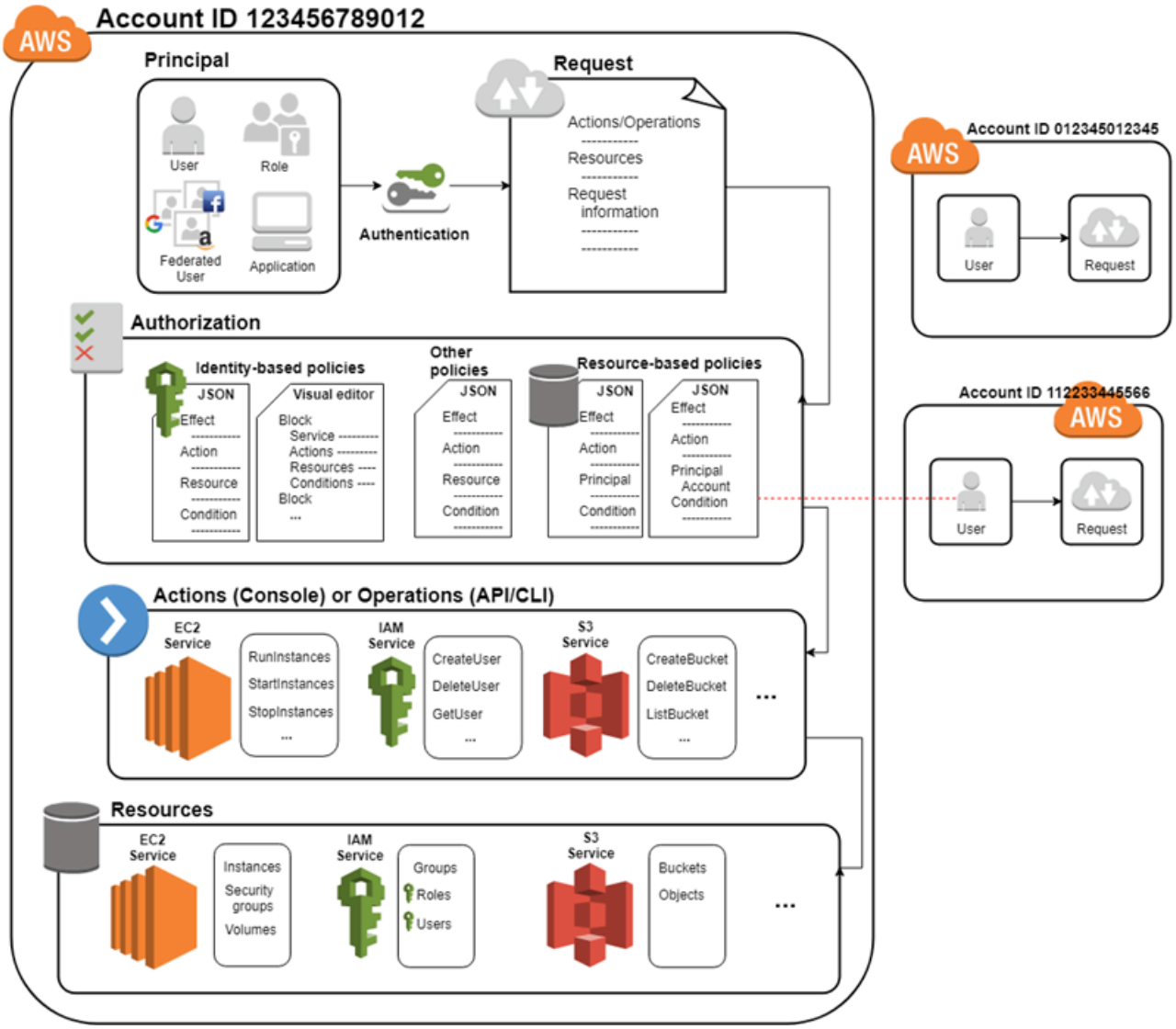
- Política de confiança: define quais entidades principais podem assumir o perfil e em que condições. A política de confiança é um tipo específico de política baseada em recursos para perfis do IAM. O perfil pode ter apenas um conjunto de políticas de confiança.
- Políticas baseadas em identidade (em 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. Você pode 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.
> 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 AWS IAM Identity Center.

Como o IAM funciona

O IAM fornece a infraestrutura necessária para controlar a autenticação e autorização de sua Conta da AWS. A infraestrutura do IAM é ilustrada no diagrama a seguir:



Primeiro, um usuário humano ou uma aplicação usa suas credenciais de login para se autenticar na AWS. A autenticação é fornecida combinando as credenciais de login com uma entidade principal (usuário do IAM, usuário federado, 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 é concedido 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. 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.

Uma vez 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.

Conceitos básicos

- Termos
- Entidade principal
- Solicitação
- Autenticação
- Autorização
- Ações ou operações
- Recursos

Termos

Estes termos do IAM são bastante usados quando se trabalha com a AWS:

Recurso do IAM

Os recursos do IAM são armazenados no próprio IAM. Você pode adicioná-los, editá-los e removê-los do IAM.

- user
- grupo
- função
- política
- objeto provedor de identidade

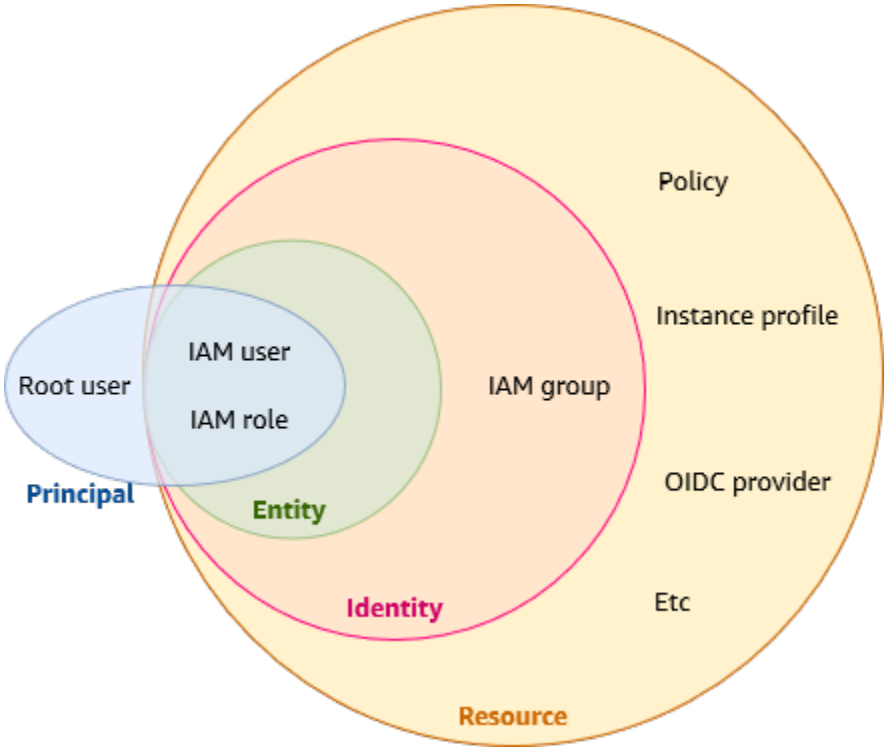
Entidade IAM

Recursos do IAM que AWS usa para autenticação. As entidades podem ser especificadas como entidade principal em uma política baseada em recursos.

- user
- função

Identidade do IAM

Um recurso do IAM que pode ser autorizado em políticas para realizar ações e acessar recursos. Identidades incluem usuários, grupos e funções.



Entidades principais

Uma pessoa ou aplicação que usa o Usuário raiz da conta da AWS, um usuário do IAM ou um perfil do IAM para fazer login e fazer solicitações à AWS. Os principais incluem usuários federados e funções assumidas.

Usuários humanos

Também conhecidos como identidades humanas,; as pessoas, os administradores, os desenvolvedores, os operadores e os consumidores de suas aplicações.

Workload

Uma coleção de códigos e recursos que fornece valor comercial, como uma aplicação ou um processo de backend. Pode incluir aplicações, ferramentas operacionais e componentes.

Entidade principal

Uma entidade principal é um usuário humano ou workload que pode fazer uma solicitação de uma ação ou operação em um recurso da AWS. Após a autenticação, a entidade principal pode receber credenciais permanentes ou temporárias para fazer solicitações à AWS, dependendo do tipo da entidade principal. Os usuários do IAM e o usuário raiz recebem credenciais permanentes, enquanto os perfis recebem credenciais temporárias. Como prática recomendada, aconselhamos que você exija que usuários humanos e workloads acessem recursos da AWS usando credenciais temporárias.

Solicitação

Quando uma entidade principal tenta usar o AWS Management Console, 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 as operações que o principal deseja executar. Pode ser uma ação no AWS Management Console, uma operação na AWS CLI ou na API da AWS.
- Recursos: o objeto de recurso da AWS no qual as ações ou operações são executadas.
- Principal – A pessoa ou o aplicativo que usou uma entidade (usuário ou função) para enviar a solicitação. As informações sobre a principal incluem as políticas associadas à entidade usada pela principal para fazer login.
- Dados do ambiente – Informações sobre o endereço IP, o agente de usuário, o status do SSL habilitado ou a hora do dia.
- 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.

O AWS reúne as informações da solicitação em um contexto de solicitação, que é usado para avaliar e autorizar a solicitação.

Autenticação

Uma entidade principal deve ser autenticada (conectado na AWS) usando suas credenciais para enviar uma solicitação para a AWS. Alguns serviços, como o Amazon S3 e o AWS STS, permitem algumas solicitações de usuários anônimos. No entanto, eles são a exceção à regra.

Para autenticar-se no console como um usuário usuário raiz, você deve fazer login com seu endereço de e-mail e senha. Como usuário federado, você é autenticado por seu provedor de identidade e recebe acesso aos recursos da AWS ao assumir perfis do IAM. Como usuário do IAM, fornece o ID da conta ou o alias e usa seu nome de usuário e senha. Para autenticar workloads da API ou da AWS CLI, você pode usar credenciais temporárias ao receber um perfil ou usar credenciais de longo prazo fornecendo sua chave de acesso e chave secreta. Também pode ser necessário fornecer informações adicionais de segurança. Como prática recomendada, a AWS aconselha o uso da autenticação multifator (MFA) e de credenciais temporárias para aumentar a segurança de sua conta. Para saber mais sobre as entidades do IAM que a AWS pode autenticar, consulte Usuários do IAM e Perfis do IAM.

Autorização

Você também deve ser autorizado (permitido) para concluir a solicitação. Durante a autorização, a AWS usa valores do contexto da solicitação para verificar a existência de políticas que se aplicam à solicitação. Em seguida, ela usa as políticas para determinar se deve permitir ou negar uma solicitação. A maioria das políticas é armazenada no AWS, como documentos JSON e especifica as permissões para as entidades principais. Há vários tipos de políticas que podem afetar a autorização de uma solicitação. Para fornecer aos usuários permissões para acessar os recursos da AWS em sua conta, você precisa somente de políticas baseadas em identidade. As políticas baseadas em recurso são populares para conceder acesso entre contas. Os outros tipos de política são recursos avançados e devem ser usados com cuidado.

O AWS verifica cada política que se aplica ao contexto da sua solicitação. Se uma única política de permissões incluir uma ação negada, o AWS negará toda a solicitação e interromperá a avaliação. Esse processo é chamado de negação explícita. Como as solicitações são negadas por padrão, o AWS só autorizará a solicitação se cada parte da solicitação tiver permissão das políticas de permissão aplicáveis. A lógica de avaliação para um solicitação em uma única conta segue estas regras gerais:

- 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 um SCP do Organizations, um limite de permissões do IAM ou uma política de sessão substitui a permissão. Se um ou mais desses tipos de política existir, todos eles devem permitir a solicitação. Caso contrário, ela será implicitamente negada.

- Uma negação explícita em qualquer política substitui todas as permissões.

Para saber mais sobre como todos os tipos de políticas são avaliadas, consulte Lógica da avaliação de política. Se você precisar fazer uma solicitação em uma conta diferente, uma política na outra conta deverá 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.

Ações ou operações

Depois que sua solicitação tiver sido autenticada e autorizada, a AWS aprovará as ações ou operações em sua solicitação. As operações são definidas por um serviço e incluem coisas que você pode fazer em um recurso, como visualizar, criar, editar e excluir esse recurso. Por exemplo, o IAM oferece suporte a aproximadamente 40 ações para um recurso de usuário incluindo as seguintes ações:

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

Para permitir que uma entidade principal execute uma operação, você deve incluir as ações necessárias em uma política aplicável à entidade principal ou ao recurso afetado. Para ver uma lista de ações, tipos de recursos e chaves de condição compatíveis com cada serviço, consulte Ações, recursos e chaves de condição de serviços da AWS.

Recursos

Depois que a AWS aprova as operações em sua solicitação, elas podem ser executadas nos 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. O serviço define um conjunto de ações que podem ser executadas em cada recurso. Se você criar uma solicitação para realizar uma ação não relacionada em um recurso, essa solicitação será negada. Por exemplo, se você solicitar a exclusão de uma função do IAM mas fornecer um recurso de grupo do IAM, a solicitação falhará. Para ver tabelas de serviços da AWS que identificam quais recursos são afetados por uma ação, consulte Ações, recursos e chaves de condição de serviços da AWS.

Visão geral do gerenciamento de identidades da AWS: usuários

Você pode conceder acesso à sua Conta da AWS para usuários específicos e fornecer permissões específicas para acessar recursos em sua Conta da AWS. É possível usar o IAM e o AWS IAM Identity Center para criar novos usuários ou federar usuários existentes na AWS. A principal diferença entre os dois é que os usuários do IAM recebem credenciais de longo prazo para seus recursos da AWS, enquanto os usuários do Centro de Identidade do IAM têm credenciais temporárias que são estabelecidas sempre que o usuário faz login na AWS. Como prática recomendada, exija que os usuários humanos usem a federação com um provedor de identidades para acessar a AWS usando credenciais temporárias em vez de um usuário do IAM. Um dos principais usos 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.

Tópicos

Somente primeiro acesso: credenciais do usuário raiz

Ao criar uma Conta da AWS, você começa com uma identidade de login que tem acesso completo a todos os recursos e Serviços da AWS na conta. Essa identidade, denominada usuário raiz da Conta da AWS, e é acessada por login com o endereço de e-mail e a senha que você usou para criar a conta. É altamente recomendável não usar o usuário raiz para tarefas diárias. Proteja as credenciais do usuário raiz e use-as para executar as tarefas que somente ele pode executar. Para obter a lista completa das tarefas que exigem login como usuário raiz, consulte Tarefas que exigem credenciais de usuário raiz no Guia do usuário do IAM. Somente as políticas de controle de serviço (SCPs) nas organizações podem restringir as permissões concedidas ao usuário raiz.

Usuários do IAM e usuários do Centro de Identidade do IAM

Os usuários do IAM não são contas separadas; eles são usuários dentro da sua conta. Cada usuário pode ter sua própria senha para o acesso ao AWS Management Console. 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 recebem credenciais de longo prazo para seus recursos da AWS. Como prática recomendada, não crie usuários do IAM com credenciais de longo prazo para usuários humanos. Em vez disso, exija que seus usuários humanos usem credenciais temporárias ao acessar a AWS.

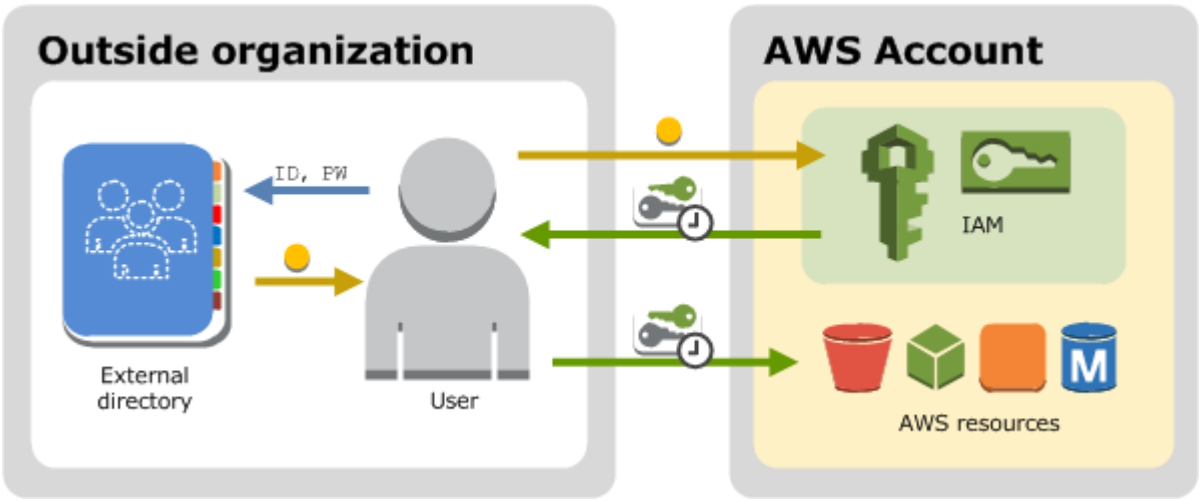
> **Note**
>
> Para cenários em que você precise de usuários do IAM com acesso programático e credenciais de longo prazo, recomendamos atualizar as chaves de acesso quando necessário. Para ter mais informações, consulte Atualização de chaves de acesso.

Em contraste, usuários no Centro de Identidade do AWS IAM recebem credenciais de curto prazo para seus recursos da AWS. Para gerenciamento de acesso centralizado, recomendamos que você usar o AWS IAM Identity Center (IAM Identity Center) para gerenciar o acesso às suas contas e as permissões nessas contas. O Centro de Identidade do IAM é configurado automaticamente com um diretório do Centro de Identidade como sua origem de identidade padrão, em que é possível criar usuários e grupos e atribuir o nível de acesso deles a seus recursos da AWS. Para obter mais informações, consulte O que é o AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center.

Federação de usuários existentes

Se os usuários em sua organização já tiverem uma forma de autenticação, por exemplo, ao fazer login na sua rede corporativa, você não precisará criar usuários do IAM separados ou usuários do Centro de Identidade do IAM para eles. Em vez disso, você pode criar a federação dessas identidades de usuários na AWS usando o IAM ou o AWS IAM Identity Center.

O diagrama a seguir mostra como um usuário pode obter credenciais de segurança temporárias da AWS para acessar recursos em sua Conta da AWS.



A federação é especialmente útil nos seguintes 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 AWS Management Console para seus usuários. Para ter mais informações, consulte Cenários comuns para credenciais temporárias.

  Se o seu diretório corporativo não for compatível com SAML 2.0, você poderá criar um aplicativo identity broker para fornecer acesso de logon único (SSO) ao AWS Management Console para seus usuários. Para ter mais informações, consulte Habilitar o acesso do agente de identidades personalizado ao console da AWS.

  Se o seu diretório corporativo for o Microsoft Active Directory, você poderá usar o AWS IAM Identity Center 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 AWS IAM Identity Center 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 AWS IAM Identity Center.
- 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 ter mais informações, consulte Federação OIDC.

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

Métodos de controle de acesso

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

Tipo de acesso do usuário	Por que devo usar?	Onde posso obter mais informações?
Acesso de autenticação única para usuários humanos, como usuários da sua força de trabalho, aos recursos da AWS que usam o Centro de Identidade do IAM	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. Como prática recomendada de segurança, aconselha-se conceder a seus usuários humanos credenciais limitadas a recursos da AWS, conforme necessário.	Para obter informações sobre como configurar o Centro de Identidade do IAM, consulte Getting Started (Conceitos básicos) no Guia do usuário do AWS IAM Identity Center Para obter mais informações sobre a MFA no Centro de Identidade do IAM, consulte Multi-factor authentication (Autenticação multifator) no Guia do usuário do AWS IAM Identity Center

Tipo de acesso do usuário	Por que devo usar?	Onde posso obter mais informações?
	Isso facilita experiência de login dos usuários, e você mantém o controle sobre o acesso deles aos recursos por um único sistema. O Centro de Identidade do IAM oferece suporte a autentica ção multifator (MFA) para aumentar a segurança da conta.	
Acesso federado para usuários humanos, como usuários da sua força de trabalho, a serviços da AWS que usam provedores de identidade do IAM	O IAM oferece suporte a IdPs compatíveis com OpenID Connect (OIDC) ou SAML 2.0 (Security Assertion Markup Language 2.0). Depois de criar um provedor de identidad e do IAM, é necessário criar um ou mais perfis do IAM que podem ser atribuídos dinamicamente a um usuário federado.	Para obter mais informações sobre provedore s de identidade do IAM e federação, consulte Provedores de identidade e federação.

Tipo de acesso do usuário	Por que devo usar?	Onde posso obter mais informações?
Acesso entre contas para Contas da AWS	Você deseja compartilhar o acesso a determinados recursos da AWS com usuários em outras Contas da AWS. As funções 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). São chamadas de políticas baseadas em recursos.	Para obter mais informações sobre perfis do IAM, consulte Perfis do IAM. Para obter mais informações sobre funções vinculadas ao serviço, consulte Usar funções vinculadas ao serviço. Para obter informações sobre quais serviços dão suporte ao uso de perfis vinculados a serviços, consulte Serviços da AWS que funcionam com o IAM. Procure os serviços que têm Sim na coluna Função vinculada 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	Por que devo usar?	Onde posso obter mais informações?
Credenciais de longo prazo para usuários do IAM designados na sua Conta da AWS	Você poderá ter 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 dos casos de uso incluem o seguinte: • 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 • O AWS IAM Identity Center 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	Para obter mais informações sobre como configurar um usuário do IAM, consulte Criar um usuário do IAM na sua Conta da AWS. Para obter mais informações sobre chaves de acesso do usuário do IAM, consulte Gerenciamento de chaves de acesso de usuários do IAM. Para obter mais informações sobre credenciais específicas de serviço para o AWS CodeCommit ou Amazon Keyspaces, consulte Uso do IAM com CodeCommit: credenciais do Git, chaves SSH e chaves de acesso da AWS e Uso do IAM com o Amazon Keyspaces (para Apache Cassandra).

Tipo de acesso do usuário	Por que devo usar?	Onde posso obter mais informações?
	prazo, recomendamos atualizar as chaves de acesso quando necessário. Para ter mais informações, consulte Atualização de chaves de acesso.	

Visão geral do gerenciamento de acesso: permissões e políticas

A parte de gerenciamento de acesso do AWS Identity and Access Management (IAM) ajuda você a definir o que uma entidade principal tem permissão para fazer em uma conta. Uma entidade principal é uma pessoa ou uma aplicação que é autenticada usando uma entidade (usuário ou função) do IAM. O gerenciamento de acesso geralmente é referenciado como autorização. Você gerencia o acesso na AWS criando políticas e anexando-as às identidades do IAM (usuários, grupos de usuários ou funções) ou aos 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 função) do IAM para fazer 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. Para obter mais informações sobre os tipos e os usos de políticas, consulte Políticas e permissões no IAM.

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. 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 Organizations.

Políticas e usuários

Os usuários do IAM são identidades no serviço. 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 criando uma política baseada em identidade, que é uma política que é anexada ao usuário ou grupo ao qual o usuário pertence. O exemplo a seguir mostra uma política JSON que permite que o usuário execute todas as ações do Amazon DynamoDB (`dynamodb:*`) na tabela `Books` na conta `123456789012` na região `us-east-2`.

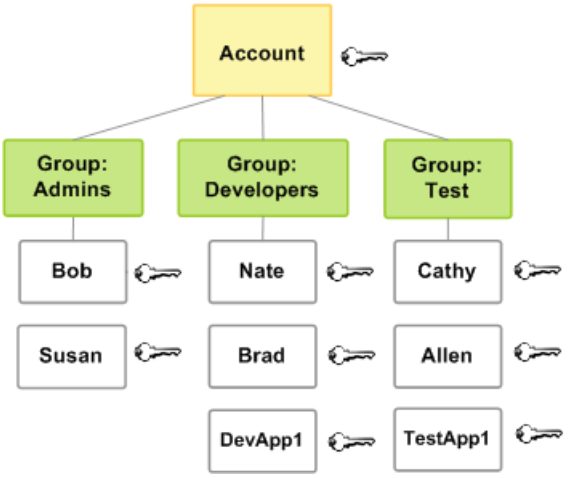
```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:123456789012:table/Books"
  }
}
```

Depois que você anexar essa política a seu usuário do IAM, o usuário terá somente essas permissões do DynamoDB. A maioria dos usuários têm várias políticas que juntas representam as permissões para esse usuário.

Por padrão, as ações ou os recursos que não são explicitamente permitidos são negados. Por exemplo, se a política anterior for a única política anexada a um usuário, esse usuário só poderá executar ações do DynamoDB na tabela `Books`. As ações em todas as outras tabelas são proibidas. Da mesma forma, o usuário não tem permissão para realizar nenhuma ação no Amazon EC2, Amazon S3 ou em qualquer outro produto da AWS. O motivo é que as permissões para trabalhar com esses serviços não estão incluídas na política.

Políticas e grupos

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



Os usuários ou os grupos podem ter várias políticas anexadas a eles que concedem diferentes permissões. Neste caso, as permissões dos usuários são calculadas com base na combinação de políticas. No entanto, o princípio básico ainda se aplica: se o usuário não recebeu uma permissão explícita para uma ação e um recurso, ele não terá essas permissões.

Usuários federados e funções

Usuários federados não têm identidades permanentes na sua Conta da AWS da mesma forma que os usuários do IAM. Para atribuir permissões a usuários federados, você pode criar uma entidade denominada função e definir permissões para a função. Quando um usuário federado fizer login na AWS, ele estará associado à função e terá as permissões definidas na função. Para obter mais informações, consulte Criar uma função para um provedor de identidade de terceiros (federação).

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 Criação de políticas do IAM e Edição de 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 recurso.

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. Como uma função do IAM é tanto uma identidade quanto um recurso que oferece suporte a políticas baseadas em recurso, 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 serviços oferecem suporte a políticas baseadas em recursos, consulte Serviços da AWS que funcionam com o IAM. Para saber mais sobre as políticas baseadas em recursos, consulte Políticas baseadas em identidade e em recurso.

O que é ABAC para a AWS?

O controle de acesso por atributo (ABAC) é uma estratégia de autorização que define permissões com base em atributos. Na AWS, esses atributos são chamados de tags. Você pode anexar tags a recursos do IAM, incluindo entidades (usuários ou funções) 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. Essas políticas de ABAC podem ser criadas para permitir operações quando a tag da entidade principal corresponder à tag de recurso. O ABAC é útil em ambientes que estão crescendo rapidamente e ajuda em situações em que o gerenciamento de políticas se torna um problema.

Por exemplo, é possível criar três funções com a chave de tag `access-project`. Defina o valor da tag da primeira função como `Heart`, a segunda como `Star` e a terceira como `Lightning`. Depois disso, é possível usar uma única política que permitirá o acesso quando a função e o recurso estiverem marcados com o mesmo valor para `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 Serviços da AWS que funcionam com o IAM.

Comparar o ABAC com o modelo de RBAC tradicional

O modelo de autorização tradicional usado no IAM é chamado de controle de acesso baseado em função (RBAC). O RBAC define permissões com base na função de trabalho de uma pessoa, conhecida fora da AWS como uma função. Na AWS, uma função geralmente se refere a uma função do IAM, que é uma identidade no IAM que você pode assumir. 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 de usuários ou funções do IAM). Como prática recomendada, conceda as permissões mínimas necessárias para o perfil de trabalho. Isso é conhecido como concessão de privilégio mínimo. Faça isso listando os recursos específicos que a função de trabalho pode acessar. A desvantagem de usar o modelo de RBAC tradicional é que, quando os funcionários adicionarem novos recursos, será 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 função do IAM para definir os recursos que qualquer pessoa com permissão para assumir a função pode acessar. Se um funcionário mudar de cargo na sua empresa, você atribuirá a ele outra função do IAM. Pessoas ou programas podem ser atribuídos a mais de uma função. 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 `Star` para especificar o novo recurso de contêiner. Caso contrário, os membros do projeto `Star` não terão permissão para acessar o novo contêiner.

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 a função 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 mudar e crescer rapidamente. Isso ocorre porque as permissões para novos recursos são concedidas automaticamente com base em atributos. Por exemplo, se a empresa já oferece suporte aos projetos `Star` e `Heart` que usam o ABAC, é fácil adicionar um novo projeto `Lightning`. Um administrador do IAM cria uma nova função com a etiqueta `access-project` = `Lightning`. Não é necessário alterar a política para oferecer suporte a um novo projeto. Qualquer pessoa com permissões para assumir a função pode criar e visualizar instâncias marcadas com `access-project` = `Lightning`. Além disso, um membro da equipe pode mudar do projeto `Heart` para o projeto `Lightning`. O administrador do IAM atribui ao usuário outra função 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 RBAC tradicional, é necessário escrever uma política que permita o acesso apenas a recursos específicos. No entanto, ao usar ABAC, será possível permitir ações em todos os recursos, mas somente se a tag de recurso for correspondente à tag do principal.
- Use atributos de funcionário do seu diretório corporativo com ABAC. É possível configurar seu provedor SAML ou OIDC 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 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.

Recursos de segurança fora do IAM

Você usa o IAM para controlar o acesso a tarefas que são executadas usando o AWS Management Console, 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 documentação a seguir:

- Conceitos básicos das instâncias do Linux do Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux
- Conceitos básicos das instâncias do Windows do Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Windows

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 documentação a seguir:

- Grupos de segurança do Amazon EC2 para instâncias do Linux no Guia do usuário do Amazon EC2 para instâncias do Linux
- Grupos de segurança do Amazon EC2 para instâncias do Windows no Guia do usuário do Amazon EC2 para instâncias do Windows
- 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.

Amazon WorkDocs

No Amazon WorkDocs, os usuários obtêm acesso a documentos compartilhados ao fazer login com um nome de usuário e senha.

Para obter mais informações, consulte Conceitos básicos do Amazon WorkDocs no Guia de administração do Amazon WorkDocs.

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 AWS Management Console. 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), desktop (Amazon WorkSpaces) ou site de colaboração (Amazon WorkDocs).

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.

Links rápidos para tarefas comuns

Use os links a seguir para obter ajuda com tarefas comuns associadas ao IAM.

Entrada para diferentes tipos de usuários

Entre no console do IAM escolhendo IAM user (Usuário do IAM) e inserindo o ID da sua conta da Conta da AWS ou o alias da conta. Na próxima página, insira seu nome de usuário do IAM e sua senha.

Para fazer login com seu usuário do Centro de Identidade do IAM, use a URL de login que foi enviada ao seu endereço de e-mail quando você criou o usuário do Centro do Usuário do IAM.

Para obter ajuda com o login utilizando um usuário do Centro de Identidade do IAM, consulte Fazer login no portal de acesso da AWS, no Guia do usuário do Início de Sessão da AWS.

Faça login no AWS Management Console 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, digite sua senha.

Consulte O que é o AWS Sign-In no Guia do usuário do Início de Sessão da AWS para obter ajuda para determinar seu tipo de usuário e sua página de login.

Gerenciar senhas para usuários do

Você precisa de uma senha para acessar o AWS Management Console, incluindo acesso a informações de faturamento.

Para o Usuário raiz da conta da AWS, consulte Alterar a senha do Usuário raiz da conta da AWS no Guia de referência do AWS Account Management

Para um usuário do IAM, consulte Gerenciamento de senhas de usuários do IAM.

Gerenciar permissões para usuários do

Você pode usar políticas para conceder permissões aos usuários do IAM na sua Conta da AWS. Quando são criados, os usuários do IAM não têm permissões, portanto, você precisa adicionar permissões para que eles possam usar os recursos da AWS.

Para conceder acesso, adicione as permissões aos seus usuários, grupos ou perfis:

- Usuários e grupos no AWS IAM Identity Center:

  Crie um conjunto de permissões. Siga as instruções em Create a permission set (Criação de um conjunto de permissões) no Guia do usuário do AWS IAM Identity Center.

- Usuários gerenciados no IAM com provedor de identidades:

  Crie um perfil para a federação de identidades. Siga as instruções em Criar um perfil para um provedor de identidades de terceiros (federação) no Guia do usuário do IAM.

- Usuários do IAM:
  - Crie um perfil que seu usuário possa assumir. Siga as instruções em Creating a role for an IAM user (Criação de um perfil para um usuário do IAM) no Guia do usuário do IAM.
  - (Não recomendado) Vincule uma política diretamente a um usuário ou adicione um usuário a um grupo de usuários. Siga as instruções em Adição de permissões a um usuário (console) no Guia do usuário do IAM.

Para obter mais informações, consulte Gerenciamento de políticas do IAM.

Listar todos os usuários na sua Conta da AWS e obter informações sobre suas credenciais

Consulte Obter relatórios de credenciais da sua Conta da AWS.

Adicionar autenticação multifator (MFA)

Para adicionar um dispositivo MFA virtual, consulte um dos procedimentos a seguir:

- Habilitar um dispositivo com MFA virtual para o Usuário raiz da conta da AWS (console)
- Habilitar um dispositivo com MFA virtual para um usuário do IAM (console)

Para adicionar uma chave de segurança FIDO, consulte um dos procedimentos a seguir:

- Habilitar uma chave de segurança FIDO para o usuário raiz da Conta da AWS (console)
- Habilitar uma chave de segurança FIDO para outro usuário do IAM (console)

Para adicionar um dispositivo MFA de hardware, consulte um dos procedimentos a seguir:

- Habilitar um token de hardware TOTP para o usuário raiz da Conta da AWS (console).
- Habilitar um token de hardware TOTP para outro usuário do IAM (console)

Obter uma chave de acesso

Você pode usar uma chave de acesso para fazer solicitações da AWS usando os AWS SDKs, as ferramentas de linha de comando da AWS ou as operações da API.

> **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.

Para obter orientação para ajudar você a proteger as chaves de acesso, consulte Proteção de chaves de acesso.

Para saber mais sobre como gerenciar chaves de acesso para um usuário do IAM, consulte Gerenciamento de chaves de acesso de usuários do IAM.

Para obter mais informações sobre as credenciais de segurança disponíveis para sua Conta da AWS, consulte Credenciais de segurança da AWS.

Etiquetar recursos do IAM

Você pode etiquetar os seguintes recursos do IAM:

- IAM users
- Perfis do IAM
- Políticas gerenciadas pelo cliente
- Provedores de identidade
- Certificados de servidor
- Dispositivos MFA virtuais

Para saber mais sobre etiquetas no IAM, consulte Recursos de etiquetas do IAM.

Para saber mais sobre como usar tags para controlar o acesso a recursos da AWS, consulte Controlar o acesso a recursos da AWS usando tags.

Visualizar as ações, os recursos e as chaves de condição de todos os serviços

Este conjunto de documentação de referência pode ajudar você a escrever políticas do IAM detalhadas. Cada produto da AWS define as ações, os recursos e as chaves de contexto de condição usados nas políticas do IAM. Para saber mais, consulte Ações, recursos e chaves de condição para serviços da AWS.

Comece a usar todos os recursos da AWS

Esse conjunto de documentação aborda principalmente o serviço IAM. Para conhecer as noções básicas da AWS e do uso de vários serviços para resolver um problema, como criar e executar seu primeiro projeto, consulte o Centro de recursos de conceitos básicos.

Pesquisa no console do IAM

Use a página de pesquisa do console do IAM como uma opção mais rápida para localizar recursos do IAM. É possível usar a pesquisa do console para localizar chaves de acesso relacionadas à sua conta, entidades do IAM (como usuários, grupos, perfis, provedores de identidade), políticas por nome etc.

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.

Usar a pesquisa no console do IAM

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, selecione o serviço do 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.

Ícones nos resultados de pesquisa no console do IAM

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

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

Criando atributos AWS Identity and Access Management com AWS 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 AWS CloudFormation provisiona e configura esses recursos para você.

Quando você usa o AWS CloudFormation, pode reutilizar o modelo para configurar os recursos do IAM repetidas vezes de modo consistente. Descreva seus recursos uma vez e depois provisione os mesmos recursos repetidamente em várias regiões e Contas da AWS.

IAM e os modelos do AWS CloudFormation

Para provisionar e configurar recursos para o IAM e os serviços relacionados, você deve entender os modelos do AWS CloudFormation. Os modelos são arquivos de texto formatados em JSON ou YAML. Esses modelos descrevem os atributos que você deseja provisionar nas suas pilhas

AWS CloudFormation. Se não estiver familiarizado com JSON ou YAML, você pode usar AWS CloudFormation Designer para ajudá-lo a começar a usar os modelos AWS CloudFormation. Para obter mais informações, consulte O que é o AWS 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 AWS 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 a AWS CloudFormation

Para saber mais sobre a AWS CloudFormation, consulte os seguintes recursos:

- AWS CloudFormation
- Guia do UsuárioAWS CloudFormation
- Referência de API AWS CloudFormation
- Guia do Usuário da Interface de Linha de ComandoAWS CloudFormation

Utilização da AWS CloudShell para operação com o AWS Identity and Access Management

O AWS CloudShell é um shell pré-autenticado baseado em navegador que você pode iniciar diretamente do AWS Management Console. 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 AWS Management Console, 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).

Obtenção de permissões do IAM para a 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.

Interação com o IAM usando a AWS CloudShell

Depois de iniciar a AWS CloudShell a partir do AWS Management Console, 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 AWS CloudShell SDK

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 AWS Management Console, 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, johndoe:

   ```
   aws iam add-user-to-group --group-name east_coast --user-name johndoe
   ```

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": "johndoe",
        "UserId": "AIDAYBDBW4JBXGEXAMPLE",
        "Arn": "arn:aws:iam::552108220995:user/johndoe",
        "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"
     }
 }
```

Usar o IAM com um AWS SDK

Os kits de desenvolvimento de software (SDKs) da AWS estão disponíveis para muitas linguagens de programação populares. Cada SDK fornece uma API, exemplos de código e documentação que facilitam a criação de aplicações em seu idioma preferido pelos desenvolvedores.

Documentação do SDK	Exemplos de código
AWS SDK for C++	Exemplos de código do AWS SDK for C++
AWS CLI	Exemplos de código do AWS CLI
AWS SDK for Go	Exemplos de código do AWS SDK for Go
AWS SDK for Java	Exemplos de código do AWS SDK for Java

Documentação do SDK	Exemplos de código
AWS SDK for JavaScript	Exemplos de código do AWS SDK for JavaScript
AWS SDK para Kotlin	Exemplos de código do AWS SDK para Kotlin
AWS SDK for .NET	Exemplos de código do AWS SDK for .NET
AWS SDK for PHP	Exemplos de código do AWS SDK for PHP
AWS Tools for PowerShell	Exemplos de código de ferramentas para PowerShell
AWS SDK for Python (Boto3)	Exemplos de código do AWS SDK for Python (Boto3)
AWS SDK for Ruby	Exemplos de código do AWS SDK for Ruby
AWS SDK para Rust	Exemplos de código do AWS SDK para Rust
SDK da AWS para SAP ABAP	Exemplos de código do SDK da AWS para SAP ABAP
AWS SDK for Swift	Exemplos de código do AWS SDK for Swift

Para obter exemplos específicos do IAM, consulte Exemplos de código para o IAM usando AWS SDKs.

Exemplo de disponibilidade

Você não consegue encontrar o que precisa? Solicite um código de exemplo no link Fornecer feedback na parte inferior desta página.

Configurações no 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.

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.

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 AWS IAM Identity Center. 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 AWS IAM Identity Center.

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

Tópicos

Cadastre-se em uma Conta da AWS

Se você ainda não tem Conta da AWS, siga as etapas a seguir para criar uma.

Para se cadastrar em uma Conta da AWS

1. Abra https://portal.aws.amazon.com/billing/signup.
2. Siga as instruções on-line.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica e digitar um código de verificação no teclado do telefone.

   Quando você se cadastra em uma Conta da AWS, um Usuário raiz da conta da AWS é criado. O usuário raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como uma prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que requerem o acesso de usuário-raiz.

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

Criar um usuário com acesso administrativo

Depois de se cadastrar em uma Conta da AWS, proteja seu Usuário raiz da conta da AWS, habilite o AWS IAM Identity Center e crie um usuário administrativo para não usar o usuário raiz em tarefas cotidianas.

Proteger seu Usuário raiz da conta da AWS

1. Faça login no AWS Management Console 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, digite sua 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. Ative a autenticação multifator (MFA) para o usuário raiz.c

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

Criar um usuário com acesso administrativo

1. Habilitar o IAM Identity Center.

   Para obter instruções, consulte Habilitar AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center.
2. No Centro de Identidade do IAM, conceda o acesso administrativo para um usuário.

   Para obter um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso dos usuários com o Diretório do Centro de Identidade do IAM padrão no Guia do usuário do AWS IAM Identity Center.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com seu usuário do Centro de Identidade do IAM, use a URL de login que foi enviada ao seu endereço de e-mail quando você criou o usuário do Centro do Usuário do IAM.

  Para obter ajuda com o login utilizando um usuário do Centro de Identidade do IAM, consulte Fazer login no portal de acesso da AWS, no Guia do usuário do Início de Sessão da AWS.

Atribuir acesso para usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

   Para obter instruções, consulte Criar um conjunto de permissões no Guia do usuário do AWS IAM Identity Center.
2. Atribua usuários para um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Adicionar grupos no Guia do usuário do AWS IAM Identity Center.

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. Tanto o usuário raiz quanto o usuário administrador têm permissões avançadas

que não são necessárias para as 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 Criação de funções 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 Políticas gerenciadas pela AWS 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.

Métodos de gerenciamento do IAM

É possível gerenciar o IAM usando o console da AWS, a interface de linha de comando da AWS ou por meio das interfaces de aplicações (APIs) nos SDKs associados. Ao se preparar, considere a quais métodos você deseja oferecer suporte e como planeja oferecer suporte a diferentes usuários.

Tópicos

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

Console do 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, os usuários federados 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 AWS Management Console.

- 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 AWS Management Console, 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.
- Usuários federados gerenciados 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 usuários federados 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.

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.

Se você estiver usando o Centro de Identidade do IAM, poderá obter credenciais de curto prazo na página inicial do portal de acesso da AWS, onde é possível escolher 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 o 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.

O ID da sua Conta da AWS e seu alias

Os usuários do IAM na conta iniciam sessão usando uma URL da web que inclui o alias ou um ID da conta. Se você não tiver o URL, a página de início de sessão da AWS exige que você forneça o alias da Conta da AWS ou o ID da conta.

Se você não souber o ID ou o alias da conta:

- Verifique o histórico do seu navegador. Se você tiver iniciado sessão anteriormente, ela poderá ter sido armazenada em seus sites recentes.
- Se você configurou a CLI da AWS ou um AWS SDK com as credenciais da sua conta, pode obter o ID da conta nos arquivos de configuração.
- Pergunte ao administrador local ou ao proprietário da conta, a AWS não pode fornecer IDs de conta aos usuários.

> **Tip**
>
> Para incluir a sua página de login da conta nos favoritos do navegador da web, você deve digitar manualmente o URL de login na entrada do favorito. Não use o atributo “marcar esta página” do seu navegador, pois ele captura informações específicas da sessão atual do navegador que interferem em futuras visitas à página de início de sessão.

Tópicos

Visualizar o ID da Conta da AWS

É possível visualizar o ID da sua Conta da AWS usando os métodos a seguir.

Visualizar o ID da conta usando o console

O ID da conta é exibido no painel do IAM na seção Conta da AWS. Existem outras maneiras de visualizar o ID da sua conta no console, dependendo do seu tipo de usuário. Se você assumiu um perfil, Credenciais de segurança não está disponível.

Tipo de usuário	Procedimento
Usuário raiz	Na barra de navegação no canto superior direito, escolha seu nome de usuário e depois Credenciais de segurança. O número da conta aparece em Identificadores da conta.
IAM user (Usuário do IAM)	Na barra de navegação no canto superior direito, escolha seu nome de usuário. O ID da conta é exibido acima do seu nome de usuário. Selecione Security credentials (Credenciais de segurança). O número da conta aparece em Detalhes da conta.

Tipo de usuário	Procedimento
Usuário federado	Na barra de navegação no canto superior direito, escolha seu nome de usuário. O ID da conta é exibido acima do seu nome de usuário.
Função assumida	Na barra de navegação no canto superior direito, escolha o ícone Suporte e selecione Support Center na lista. Seu número (ID) de conta de 12 dígitos conectada no momento aparece no painel de navegação Support Center (Central de Suporte).

Visualizar o ID da conta usando a 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

Visualizar o ID da conta usando a API

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

- GetCallerIdentity

Sobre alias de contas

Se quiser que o URL para a sua página de login contenha o nome da sua empresa (ou outro identificador amigável) em vez do ID da sua Conta da AWS, você pode criar um alias de conta. Essa seção fornece informações sobre o alias de Conta da AWS e lista as operações de API que você usa para criar um alias.

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

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

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

- 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)

Criar, excluir e listar um alias de Conta da AWS

É possível usar o AWS Management Console, a API do IAM ou a interface da linha de comando para criar ou excluir o alias da sua Conta da AWS.

> **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.

Criar ou editar o alias de uma conta (console)

Você pode criar, editar e excluir um alias de conta do AWS Management Console.

Permissões mínimas

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

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

Criar ou editar o alias de uma conta (console)

1. Faça login no AWS Management Console 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.

Note

Você pode ter somente um alias associado à sua Conta da AWS de cada vez. Se você criar um novo alias, o anterior será removido, e o URL de início de sessão associado ao alias anterior deixará de funcionar.

Excluir o alias de uma conta (console)

Você pode criar e excluir o alias de uma conta do AWS Management Console.

Permissões mínimas

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

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

Excluir o alias de uma conta (console)

1. Faça login no AWS Management Console 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.

> **Note**
>
> O único URL de início de sessão da conta é baseado no ID da sua conta. Qualquer tentativa de conexão com o URL do alias não é redirecionada.

Criar, excluir e listar alias (AWS CLI)

> **Note**
>
> Para usar os comandos a seguir, você deve ter pelo menos as seguintes permissões do IAM:
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`
> - `iam:DeleteAccountAlias`

Para criar um alias para o URL da página de login do AWS Management Console, execute o seguinte comando:

- `aws iam create-account-alias`

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

- `aws iam delete-account-alias`

Para exibir o alias de ID da sua Conta da AWS, execute o seguinte comando:

- `aws iam list-account-aliases`

Example Comandos de alias

Para exibir o alias de ID da sua Conta da AWS, execute o comando a seguir.

```
$ aws iam list-account-aliases
{
    "AccountAliases": [
        "myaccountalias"
    ]
}
```

Para criar um alias para o início de sessão do AWS Management Console, execute o seguinte comando:

```
$ aws iam create-account-alias \
    --account-alias myaliasname
```

Se for bem-sucedido, esse comando não produzirá uma saída.

Para excluir um alias de ID da Conta da AWS, execute o comando a seguir.

```
$ aws iam delete-account-alias \
--account-alias myaliasname
```

Se for bem-sucedido, esse comando não produzirá uma saída.

Criar, excluir e listar alias (API da AWS)

> **Note**
>
> Para usar estas operações de API, você deve ter pelo menos as seguintes permissões do IAM:
>
> - `iam:ListAccountAliases`
> - `iam:CreateAccountAlias`
> - `iam:DeleteAccountAlias`

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

- `CreateAccountAlias`

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

- `DeleteAccountAlias`

Para exibir o alias de ID da sua Conta da AWS, chame a seguinte operação:

- `ListAccountAliases`

Conceitos básicos do IAM

Use este tutorial para começar a usar o AWS Identity and Access Management (IAM). Você aprenderá a criar perfis, usuários e políticas usando o AWS Management Console.

O AWS Identity and Access Management é um recurso da sua Conta da AWS oferecido gratuitamente. Você pagará somente por outros produtos da AWS utilizados pelos usuários do IAM. Para obter informações sobre preços de outros produtos da AWS, consulte a página de preços da Amazon Web Services.

> **Note**
>
> Esse conjunto de documentação aborda principalmente o serviço IAM. Para conhecer as noções básicas da AWS e do uso de vários serviços para resolver um problema, como criar e executar seu primeiro projeto, consulte o Centro de recursos de conceitos básicos.

Conteúdos

Pré-requisitos

Antes de começar, é necessário concluir as etapas em Configurações no IAM. Este tutorial usa a conta de administrador criada no procedimento.

Criar seu primeiro 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. É possível organizar os usuários em grupos que compartilham as mesmas permissões.

> **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).

Com o objetivo de se familiarizar com o processo de criação de um usuário do IAM, este tutorial orienta você na criação de um usuário e de um grupo do IAM para acesso de emergência.

Para criar seu primeiro usuário 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, selecione o serviço do IAM.
3. No painel de navegação, selecione Usuários e Adicionar usuários.

   > **Note**
   >
   > Se o Centro de Identidade do IAM estiver habilitado, o AWS Management Console exibirá um lembrete de que é melhor gerenciar o acesso dos usuários no Centro de Identidade do IAM. Neste tutorial, o usuário do IAM que você criar será usado especificamente apenas quando suas credenciais de usuário no Centro de Identidade do IAM não estiverem disponíveis.

4. Em Nome do usuário, digite **`EmergencyAccess`**. Os nomes não podem conter espaços.
5. Marque a caixa de seleção ao lado de Fornecer acesso ao AWS Management Console: 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 retornar à página Definir permissões.
11. Em Grupos de usuários, selecione o nome do **`EmergencyAccessGroup`** que você criou anteriormente.
12. Selecione Avançar para ir 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 federada.

O novo usuário do IAM será exibido na lista Usuários. Selecione o link Nome de usuário para ver os detalhes do usuário. Em Resumo, copie o ARN do usuário para a área de transferência. Cole o ARN em um documento de texto para poder usá-lo no próximo procedimento.

Criar seu primeiro perfil

Os perfis do IAM são uma forma segura de conceder permissões a entidades nas quais você confia. Uma função do IAM tem algumas semelhanças com um usuário do IAM. Perfis e usuários são entidades principais 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. Usar perfis ajuda a seguir as práticas recomendadas do IAM. É possível usar etiquetas para:

- Permitir acesso das identidades da força de trabalho e das aplicações habilitadas pelo Centro de Identidade à AWS Management Console usando o AWS IAM Identity Center.
- Delegue permissões do IAM para o serviço da AWS executar ações em seu nome.

- Permitir que o código da aplicação em execução em uma instância do Amazon EC2 acesse ou modifique os recursos da AWS.
- Conceder acesso a outra Conta da AWS.

> **Note**
>
> Você pode usar o AWS Identity and Access Management Roles Anywhere para dar acesso a identidades de máquina. Usar o IAM Roles Anywhere significa que não é necessário gerenciar credenciais de longo prazo para workloads executadas fora da AWS. Para obter mais informações, consulte O que é o AWS Identity and Access Management Roles Anywhere, no Guia do usuário do AWS Identity and Access Management Roles Anywhere.

O Centro de Identidade do IAM e outros serviços da AWS criam automaticamente perfis para os respectivos serviços. Se você estiver usando usuários do IAM, recomendamos criar perfis para seus usuários assumirem quando fizerem login. Isso lhes concederá permissões temporárias durante a sessão, em vez de permissões de longo prazo.

O assistente do AWS Management Console que orienta ao longo das etapas da criação do perfil exibirá etapas ligeiramente diferentes se você criar um perfil para um usuário do IAM, para um serviço da AWS ou para um usuário federado. O acesso regular a Contas da AWS dentro de uma organização deve ser fornecido usando o acesso federado. Se você estiver criando usuários do IAM para fins específicos, como acesso de emergência ou acesso programático, conceda a esses usuários do IAM permissão somente para assumir um perfil e colocar esses usuários do IAM em grupos de perfis específicos.

Nesse procedimento, você criará um perfil para fornecer acesso SupportUser ao usuário do IAM EmergencyAccess. Antes de iniciar o procedimento, copie seu ARN do usuário do IAM para a área de transferência.

Para criar uma perfil para um usuário 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, selecione o serviço do IAM.
3. No painel de navegação do console do IAM, escolha Roles (Funções) e Criar função (Create role).

4. Escolha o tipo de perfil do Conta da AWS.
5. Em Selecionar entidade confiável, em Tipo de entidade confiável, escolha Política de confiança personalizada.
6. Na seção Política de confiança personalizada, revise a política de confiança básica. É a que usaremos para este perfil. Use o editor Editar instrução para atualizar a política de confiança:

   1. Em Adicionar ações para o STS, selecione Assumir perfil.
   2. Ao lado de Adicionar uma entidade principal, selecione Adicionar. A janela Adicionar entidade principal é aberta.

      Em Tipo de entidade principal, selecione Usuários do IAM.

      Em ARN, cole o ARN do usuário do IAM copiado para a área de transferência.

      Selecione Adicionar entidade principal.
   3. Verifique se a linha `Principal` na política de confiança agora contém o ARN especificado:

      ```
      "Principal": { "AWS": "arn:aws:iam::123456789012:user/username" }
      ```
7. Resolva os avisos de segurança, erros ou avisos gerais gerados durante a validação de política e depois escolha Avançar.
8. Em Adicionar permissões, marque a caixa de seleção ao lado da política de permissões a ser aplicada. Para este tutorial, selecionaremos a política de confiança SupportUser. Você pode então usar esse perfil para solucionar e resolver problemas com a Conta da AWS e abrir casos de suporte com a AWS. Não definiremos um limite de permissões no momento.
9. Escolha Next (Próximo).
10. Em Nomear, revisar e criar, conclua estas configurações:

    - Em Nome do perfil, insira um nome que identifique o perfil como SupportUserRole.
    - Em Descrição, explique o uso pretendido do perfil.

    Como outros recursos de AWS podem fazer referência à função, não é possível editar o nome da função depois de ela ser criada.
11. Selecione Criar perfil.

    Depois que o perfil for criado, compartilhe as informações do perfil com as pessoas que precisam dele. É possível compartilhar as informações do perfil da seguinte forma:

- 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.
- Salve as informações do link do perfil junto com as credenciais de usuário EmergencyAccess.

Para obter mais detalhes, consulte Fornecer informações ao usuário.

Criar sua primeira política do IAM

As políticas do IAM são anexadas a identidades (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 recurso, define suas permissões.

Para criar sua primeira política 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, selecione o serviço do IAM.
3. No painel de navegação, escolha Policies.

   Se essa for a primeira vez que você escolhe Políticas, a página Bem-vindo às políticas gerenciadas será exibida. Escolha Conceitos básicos.
4. Escolha Create policy (Criar política).
5. Na página Criar política, escolha Ações e, em seguida, selecione Importar política.
6. Na janela Importar política, na caixa Localizar políticas, digite **`power`** para reduzir a lista de políticas. Selecione a política PowerUserAccess.
7. Selecione Importar política. A política é exibida na guia JSON.
8. Escolha Next (Próximo).
9. Na página Analisar e criar, em Nome da política, digite **`PowerUserExamplePolicy`**. Em Description (Descrição), digite **`Allows full access to all services except those for user management`**. Em seguida, escolha Criar política para salvar a política.

Você pode anexar essa política a um perfil para fornecer aos usuários que assumem o perfil as permissões associadas à política. A política PowerUserAccess é bastante usada para fornecer acesso a desenvolvedores.

Acesso programático

Os usuários precisam de acesso programático se quiserem interagir com a AWS de fora do AWS Management Console. 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?	Para	Por
Identificação da força de trabalho (Usuários gerenciados no Centro de Identidade do IAM)	Use credenciais de curto prazo para assinar solicitaç ões programáticas para as APIs da AWS ou a AWS CLI (diretamente ou usando os AWS SDKs).	Siga as instruções para a interface que você deseja usar: • Para a AWS CLI, siga as instruções em Como obter credenciais de perfil do IAM para acesso à CLI no Guia do usuário do AWS IAM Identity Center. • Para as APIs da AWS, siga as instruções em Credencia is do SSO no Guia de

Qual usuário precisa de acesso programático?	Para	Por
		referência de AWS SDKs e ferramentas.
IAM	Use credenciais de curto prazo para assinar solicitaç ões programáticas para as APIs da AWS ou a AWS CLI (diretamente ou usando os AWS SDKs).	Siga as instruções em Uso de credenciais temporárias com recursos da AWS.
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.

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

O AWS Identity and Access Management (IAM) fornece uma série de recursos de segurança a serem 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. Como essas práticas recomendadas podem não ser adequadas ou suficientes no seu ambiente, trate-as como considerações úteis em vez de requisitos.

Para obter os maiores benefícios do IAM, reserve um 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 produtos da AWS.

Tópicos

Práticas recomendadas de segurança no IAM

Follow us on Twitter

As práticas recomendadas do AWS Identity and Access Management foram atualizadas em 14 de julho de 2022.

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

Tópicos

- 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 IAM Access Analyzer 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, aplicação do cliente, aplicação para dispositivos móveis ou ferramentas interativas de linha de comando.

Exija que seus usuários humanos usem credenciais temporárias ao acessar a AWS. Você pode 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 AWS IAM Identity Center (IAM Identity Center) para gerenciar o acesso às suas contas e as permissões nessas contas. Você pode 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 AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center.

Para obter mais informações sobre funções do , 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 back-end. Sua workload pode ter aplicações, ferramentas operacionais e componentes que exigem uma identidade para fazer solicitações aos Serviços da AWS, como solicitações de leitura de dados. 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 funcionam fora da AWS, você pode usar o AWS Identity and Access Management Roles Anywhere. Para obter mais informações sobre funções do , 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.

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 Uso de autenticação multifator (MFA) na 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 AWS IAM Identity Center.

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 Atualização de 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 Uso do IAM com 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 AWS Management Console. 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. Você pode 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 IAM.

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 Políticas gerenciadas pela AWS. Para obter mais informações

sobre políticas gerenciadas pela AWSque são projetadas para funções de trabalho específicas, consulte Políticas gerenciadas pela AWS 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

Você pode 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 as informações do último acesso.

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

Você pode 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 usando SSL. 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 AWS 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. Você pode 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 IAM Access Analyzer 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. Você pode 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 serviço (SCPs) do Organizations para estabelecer barreiras de proteção para permissões a fim de controlar o acesso de todos os usuários e funções do IAM em suas contas. As SCPs são um tipo de política de organização que você pode usar para gerenciar permissões em sua organização no AWS Organizations, nas UO ou a nível de conta. As barreiras de proteção para permissões que você

estabelece se aplicam a todos os usuários e funções nas contas abrangidas. No entanto, as SCPs por si só são insuficientes para conceder permissões às contas em sua organização. Para fazer isso, seu administrador 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 AWS Organizations, contas e barreiras de proteção do IAM.

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 garantir que não exponha suas credenciais altamente privilegiadas para uso não autorizado.

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

- Para um único e autônomo Conta da AWS, consulte Criar um usuário com acesso administrativo.
- 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 vários usuários Contas da AWS gerenciados por meio de Organizations, use os usuários da força de trabalho do IAM Identity Center. 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 AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center.

Tópicos

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 ter 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. Recomendamos enfaticamente habilitar a autenticação de múltiplos fatores (MFA) múltipla para suas credenciais de usuário raiz a fim de fornecer flexibilidade e resiliência adicionais em sua estratégia de segurança. Você pode registrar até oito dispositivos de MFA de qualquer combinação dos tipos de MFA atualmente compatíveis com seu usuário raiz Conta da AWS.

- 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 da conta Serviços da AWS, incluindo informações de cobrança.

Como apenas algumas tarefas exigem o usuário raiz e elas normalmente são executadas com pouca frequência, recomendamos entrar no AWS Management Console para realizar as tarefas de usuário raiz. Antes de criar chaves de acesso, avalie as alternativas às 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 garantir 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 seu usuário raiz Conta da AWS estiver perdida, danificada ou não estiver funcionando, você pode 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 conta Organizations

Conforme você migra para uma estratégia de várias contas com Organizations, cada um dos seus 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 de membros.

Credenciais seguras de usuário raiz para contas de membros

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

- Proteja as credenciais de usuário raiz das suas contas Organizations com MFA.
- Não redefina a senha do usuário raiz para suas contas, e apenas recupere o acesso a ela quando necessário, utilizando o processo de redefinição de senha. Quando você cria uma conta de membro em sua organização, o Organizations automaticamente cria um perfil do IAM (Identity and Access Management) na conta de membro que permite que a conta de gerenciamento tenha acesso temporário à conta de membro.

Para obter detalhes, consulte Acessando contas de membros em sua organização no Guia do usuário do Organizations.

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

Se você usa o Organizations para gerenciar várias contas, você pode aplicar um SCP para restringir o acesso ao usuário raiz da conta membro. Negar todas as ações do usuário raiz em suas contas de 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 garantir que o uso das credenciais do usuário raiz seja rastreado e realizar verificações de segurança que podem ajudar a prevenir o uso não autorizado.

- Se você deseja ser notificado sobre a atividade de login do usuário raiz em sua conta, você pode aproveitar o Amazon CloudWatch para criar uma regra de Eventos 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

- AWS Config usa regras para auxiliar na aplicação de práticas recomendadas do usuário raiz. Você pode usar as regras gerenciadas AWS para exigir que os usuários raiz tenham a autenticação multifator (MFA) ativada. AWS ConfigO também pode identificar as chaves de acesso do usuário raiz.
- O Security Hub fornece uma visão abrangente do estado de segurança na AWS e ajuda a avaliar o AWS ambiente em relação aos padrões e às práticas recomendadas do setor de segurança, como ter o MFA no usuário raiz e não ter as chaves de acesso do usuário raiz. Para obter detalhes sobre as regras disponíveis, consulte AWS Identity and Access Managementos controles no Guia do usuário do Security Hub.
- 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 Serviços da AWS que funcionam com o IAM.

Tópicos

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 AWS IAM Identity Center (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 Centro de Identidade do IAM 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 Políticas gerenciadas pela AWS para funções de trabalho no AWS Management Console para criar esses conjuntos de permissões:

- Administrador
- Faturamento
- 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 AWS IAM Identity Center. 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 AWS IAM Identity Center.

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, acesse o Guia do usuário do Amazon EC2 para instâncias do Linux.

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 de usuários exigem 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 de como essas políticas podem ser, consulte Exemplos de políticas baseadas em identidade do IAM e Uso do AWS Identity and Access Management no Guia do usuário do Amazon EC2 para instâncias do Linux.

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. John criou um bucket do Amazon S3 para a empresa chamado aws-s3-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 permitir isso, Mateo organiza logicamente os dados no aws-s3-

bucket usando um esquema de prefixo de chaves do Amazon S3, conforme mostrado na figura a seguir.

```
/aws-s3-bucket
    /home
        /zhang
        /major
    /share
        /developers
        /managers
```

Mateo divide `/aws-s3-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 `/aws-s3-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 `/aws-s3-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 `/aws-s3-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 `/aws-s3-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 tempo de execução, 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 chamado aws-s3-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: Conceder acesso ao console de faturamento
- 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: Conceder acesso ao console de faturamento

O proprietário da Conta da AWS (Usuário raiz da conta da AWS) pode conceder aos usuários e perfis do IAM acesso aos dados de AWS Billing and Cost Management para suas Conta da AWS. As instruções neste tutorial ajudarão você a configurar um cenário pré-testado. Este cenário ajudará você a obter experiência prática na configuração de permissões de faturamento sem a preocupação de afetar a conta de produção principal da AWS.

Pré-requisitos

Faça os seguintes preparativos antes de realizar as etapas neste tutorial:

- Crie uma Conta da AWS de teste.
- Faça login na sua Conta da AWS de teste como usuário raiz.
- Registre o número da Conta da AWS da sua conta de teste para poder usá-la no tutorial. Neste tutorial, usamos o número de conta de exemplo 111122223333. Sempre que uma etapa usar esse número de conta, substitua-o pelo número da sua conta de teste.

Etapa 1: Ativar o acesso ao IAM às informações de faturamento na sua conta de teste da AWS

Nesse cenário, você faz login na Conta da AWS de teste como usuário raiz para conceder ao IAM acesso às informações de faturamento. Quando você concede ao IAM acesso às informações de faturamento, ele permite que usuários e funções do IAM acessem o console AWS Billing and Cost Management. Essa configuração não concede aos usuários e perfis do IAM as permissões necessárias para essas páginas; ela concede acesso aos usuários ou perfis do IAM que têm as políticas do IAM necessárias. Se as políticas já estiverem vinculadas a usuários ou perfis do IAM, mas essa configuração não estiver habilitada, as permissões concedidas por essas políticas não estarão em vigor.

> **Note**
>
> Contas da AWS criadas usando AWS Organizations tem acesso do IAM às informações de faturamento habilitado por padrão.

Etapa 2: Criar grupos e usuários de teste

Nesse cenário, você concede aos usuários do IAM acesso ao console de cobrança e cria dois usuários:

- Pat Candella

  Pat é membro do departamento financeiro e trabalha com faturamento e pagamentos. Pat requer acesso total às informações de cobrança em sua Conta da AWS.

- Terry Whitlock

  Terry faz parte do seu departamento de suporte de TI. Na maioria das vezes, Terry não precisa acessar o console de faturamento, mas às vezes precisa de acesso para responder às perguntas dos funcionários do departamento financeiro.

Etapa 3: Criar um perfil para conceder acesso ao console do AWS Billing

Um perfil do IAM é uma identidade do IAM que você pode criar em sua conta que tem permissões específicas. Um perfil do IAM é semelhante a um usuário do IAM no sentido de ser 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, uma função não tem credenciais de longo prazo padrão, como uma senha ou chaves de acesso,

associadas a ela. Em vez disso, quando você assumir um perfil, ele fornecerá credenciais de segurança temporárias para sua sessão de perfil. Você pode usar funções para delegar acesso a usuários, aplicativos ou serviços que normalmente não têm acesso aos seus recursos da AWS. Nesse cenário, você cria um perfil que Terry Whitlock pode assumir para acessar o console de cobrança.

Etapa 4: Testar o acesso ao console

Depois de concluir as tarefas principais, você estará pronto para testar a política. Os testes garantem que a política funcione da maneira desejada. Ao testar o acesso de cada usuário, você pode comparar as experiências do usuário.

Pré-requisitos

Faça os seguintes preparativos antes de realizar as etapas neste tutorial:

- Crie uma Conta da AWS de teste.
- Faça login na sua Conta da AWS de teste como usuário raiz.
- Registre o número da Conta da AWS da sua conta de teste para poder usá-la no tutorial. Neste tutorial, usamos o número de conta de exemplo 111122223333. Sempre que uma etapa usar esse número de conta, substitua-o pelo número da sua conta de teste.

Etapa 1: Ativar o acesso ao IAM às informações de faturamento na sua conta de teste da AWS

Nesse cenário, você faz login na Conta da AWS de teste como usuário raiz para conceder ao IAM acesso às informações de faturamento. Quando você concede acesso às informações de faturamento, ele permite que usuários e funções do IAM acessem o console AWS Billing and Cost Management. Essa configuração não concede aos usuários e perfis do IAM as permissões necessárias para essas páginas; ela apenas concede acesso aos usuários ou perfis do IAM que têm as políticas do IAM necessárias.

> **Note**
>
> Contas da AWS criadas usando AWS Organizations tem acesso do IAM às informações de faturamento habilitado por padrão.

Como ativar o acesso ao console do Billing and Cost Management para a função e o usuário do IAM

1. Faça login no AWS Management Console 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 Billing and Cost Management.
5. Escolha Atualizar.

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

Na próxima etapa deste tutorial, você deve anexar políticas do IAM para conceder ou negar acesso a recursos de faturamento específicos.

Etapa 2: Criar grupos e usuários de teste

Sua conta da AWS de teste não tem nenhuma identidade definida, exceto para o usuário raiz. Para fornecer acesso às informações de faturamento, criamos identidades adicionais às quais podemos conceder permissão para acessar as informações de faturamento.

Criar grupos e usuários de teste

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 AWS Management Console como usuário raiz no Guia do usuário do Início de Sessão da AWS.

2. No painel de navegação, selecione Usuários e Adicionar usuários.

> **Note**
>
> Se o Centro de Identidade do IAM estiver habilitado, o AWS Management Console exibirá um lembrete de que é melhor gerenciar o acesso dos usuários no Centro de Identidade do IAM. Neste tutorial, os usuários do IAM que criamos aprenderão a fornecer acesso às informações de faturamento. Se você criou usuários no Centro de Identidade do IAM, atribua o conjunto de permissões de Faturamento a esses usuários ou grupos usando o Centro de Identidade do IAM em vez do IAM.

3. Em User name (Nome do usuário), digite **`pcandella`**. Os nomes não podem conter espaços.
4. Marque a caixa de seleção ao lado de Fornecer acesso ao AWS Management Console: opcional e escolha Quero criar um usuário do IAM.
5. Em Senha do console, selecione Senha gerada automaticamente.
6. Desmarque a caixa de seleção ao lado de O usuário deverá criar uma nova senha no próximo login (recomendado) e selecione Próximo. Como esse usuário do IAM está sendo testado, faremos o download da senha para uso durante o procedimento de verificação.
7. Na página Definir permissões, em Opções de permissões, selecione Adicionar usuário ao grupo. Em seguida, em Grupos de usuários, selecione Criar grupo.
8. Na página Criar grupo de usuários, em Nome do grupo de usuários, insira **`BillingGroup`**. Em seguida, em Políticas de permissões, selecione a política de função de trabalho da AWS gerenciada Billing.
9. Selecione Criar grupo de usuários para retornar à página Definir permissões.
10. Em Grupos de usuários, marque a caixa de seleção do **`BillingGroup`** que você criou.
11. Selecione Avançar para ir para a página Revisar e criar.
12. Na página Revisar e criar, revise a lista de associações de grupos de usuários para o novo usuário. Quando você estiver pronto para continuar, selecione Criar usuário.
13. Na página Recuperar senha, selecione Baixar arquivo .csv para salvar um arquivo .csv com as informações de acesso do usuário (URL da conexão, nome de usuário e senha).

    Salve esse arquivo para usar como referência ao fazer login na AWS como usuário do IAM
14. Selecione Retornar à lista de usuários
15. Repita esse procedimento usando as seguintes modificações para criar o usuário para Terry Whitlock e um grupo para usuários de suporte.

a. Na etapa 3, em Nome de usuário, digite **twhitlock**.

b. Na etapa 8, em Nome do grupo de usuário, digite **SupportGroup**. Em seguida, em Políticas de permissões, selecione a política de função de trabalho da AWS gerenciada SupportUser.

Você pode analisar os novos usuários, grupos e perfis do IAM nas listas do console. Para cada item que você criou, você pode selecionar o nome para ver seus detalhes. Quando você visualiza os detalhes do usuário, o console exibe Billing listado em Políticas de permissões para **pcandella**, e SupportUser listado em Políticas de permissões para **twhitlock**.

Para obter mais informações sobre o uso de políticas para conceder acesso a recursos do AWS Billing and Cost Management para aos usuários do IAM, consulte Usar políticas baseadas em identidade (políticas do IAM)AWS Billing no Guia do usuário do AWS Billing.

Etapa 3: Criar um perfil para conceder acesso ao console do AWS Billing

Você pode usar um perfil para conceder aos usuários do IAM acesso ao console de faturamento. Os perfis fornecem credenciais temporárias que os usuários podem assumir quando necessário. Neste tutorial, o usuário **twhitlock** precisa ser capaz de acessar as informações de faturamento quando uma solicitação de suporte do departamento financeiro exige que ele investigue um problema.

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 AWS Management Console como usuário raiz no Guia do usuário do Início de Sessão da AWS.

2. No painel de navegação, selecione Usuários e, em seguida, selecione o usuário do **twhitlock** para ver os detalhes dele. Copie o ARN do usuário do **twhitlock** para a área de transferência.

3. No painel de navegação, selecione Perfis e Criar novo perfil.

4. Na página Selecionar entidade confiável, selecione Política de confiança personalizada e, em Editar declaração, preencha os seguintes itens:

   - Adicionar ações para STS: verifique se AssumeRole está selecionado.
   - Adicionar uma entidade principal: selecione Adicionar para exibir a caixa de diálogo Adicionar entidade principal. Para o Tipo de entidade principal, selecione Usuários do IAM e, em ARN, cole o ARN do usuário twhitlock que você copiou para a área de transferência na etapa 16. Em seguida, selecione Adicionar entidade principal.

5. Selecione Próximo para acessar a página Adicionar permissões.
6. Em Políticas de permissões na caixa de filtro, digite **`Billing`** e selecione a política de função de trabalho da AWS gerenciada Billing.
7. Escolha Próximo para ir até a página Nomear, revisar e criar. Em Nome do perfil, insira **`TempBillingAccess`** e selecione Criar perfil.

   Você é notificado de que o perfil foi criado. Visualize o perfil para exibir os detalhes sobre ele. Na seção Resumo, observe o seguinte:

   - Por padrão, a duração máxima da sessão é 1 hora. Após esse período, o usuário que assumiu o perfil é revertido para as permissões da conta básica. Se o usuário quiser continuar usando as permissões do perfil, ele deverá trocar de perfil novamente. Você pode editar o perfil para aumentar a duração máxima. A sessão mais longa possível é de 12 horas.
   - Link para alternar perfis no console. Você pode copiar o link para fornecê-lo diretamente aos usuários que você adiciona como entidades principais na política de confiança. Você pode visualizar e editar a política de confiança na guia Relações de confiança.

Etapa 4: Testar o acesso ao console

Recomendamos que você teste o acesso fazendo login com os usuários de teste para conhecer o que os usuários podem experimentar. Use as etapas a seguir para fazer login usando as duas contas de teste para ver a diferença entre os direitos de acesso.

Para testar o acesso de faturamento fazendo login com ambos os usuários de teste

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.

2. Faça login com cada usuário usando as etapas descritas a seguir, para que você possa comparar as diferentes experiências do usuário.

   Acesso total

   a. Faça login no Conta da AWS como usuário do **`pcandella`**.
   b. Na barra de navegação, escolha pcandella@111122223333 e, em seguida, escolha Painel de faturamento.
   c. Navegue pelas páginas e escolha os vários botões para garantir que você tenha permissões de modificação totais.

   Sem acesso

   a. Faça login no Conta da AWS como usuário do **`twhitlock`**.
   b. Na barra de navegação, escolha twhitlock@111122223333 e, em seguida, escolha Painel de faturamento.
   c. Uma mensagem é exibida informando que você precisa de permissões. Nenhum dado de faturamento está visível.

   Troque de perfil para elevar o acesso

   a. Faça login no Conta da AWS como usuário do **`twhitlock`**.
   b. Na barra de navegação, escolha twhitlock@111122223333 e escolha Alternar perfil.

      A página Alternar perfil é aberta. Preencha as seguintes informações:

      - conta-111122223333
      - perfil-**`TempBillingAccess`**

Selecione Alternar perfil

Como alternativa, você pode usar o URL fornecido em Link para alternar perfis no console e abrir a página Alternar perfil.

c. O console exibe o Painel do AWS Billing e a barra de navegação exibe TempBillingAccess@111122223333.

Resumo

Agora você concluiu as etapas necessárias para fornecer aos usuários do IAM acesso ao console do AWS Billing. Como resultado, você viu em primeira mão como será a experiência do console de faturamento dos usuários. Agora é possível prosseguir para implementar essa lógica no ambiente de produção conforme sua conveniência.

Recursos relacionados

Para obter as informações relacionadas encontradas no Guia do usuário do AWS Billing, consulte os seguintes recursos:

- Como ativar o acesso ao console do AWS Billing
- Exemplos de políticas do Faturamento da AWS
- Como usar políticas baseadas em identidade (políticas do IAM) para o Faturamento da AWS
- Migrar o controle de acesso para o AWS Billing

Para obter informações relacionadas no Guia do usuário do IAM, consulte os seguintes recursos:

- Políticas gerenciadas e em linha
- Controlar o acesso de usuário do IAM ao AWS Management Console
- Anexar uma política a um grupo de usuários do IAM

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

Este tutorial ensina a usar um perfil para delegar acesso a recursos em Contas da AWS diferentes de sua propriedade chamadas Produção e Desenvolvimento. 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 AWS Management Console, da AWS CLI e da API.

> **Note**
>
> 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.

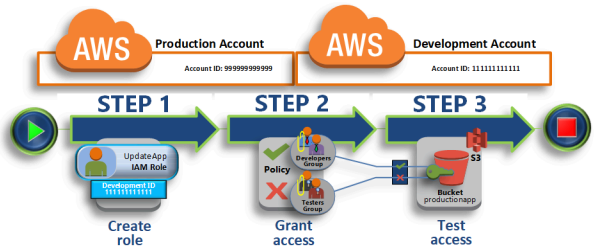
Neste tutorial, a conta de Produção gerencia aplicações ativas. Desenvolvedores e testadores usam a conta de Desenvolvimento como uma sandbox para testar aplicações livremente. Em cada conta, você armazena as informações da aplicação em buckets do Amazon S3. Você gerencia os usuários do IAM na conta de Desenvolvimento, na qual há dois grupos de usuários do IAM: Desenvolvedores e Testadores. Os usuários nos dois grupos de usuários têm permissões para trabalhar na conta de desenvolvimento e acessar os recursos. Periodicamente, um desenvolvedor deve atualizar as aplicações ativas na conta de Produção. Os desenvolvedores armazenam essas aplicações em um bucket do Amazon S3 chamado `productionapp`.

Ao final deste tutorial, você terá o seguinte:

- Os usuários na conta de Desenvolvimento (a conta confiável) com permissão para assumir uma função específica na conta de Produção.
- Uma função na conta de Produção (a conta de confiança) com permissão para acessar um bucket específico do Amazon S3.
- Um bucket `productionapp` criado na conta de Produção.

Os desenvolvedores podem usar a função no AWS Management Console para acessar o bucket `productionapp` na conta de Produção. Eles também podem acessar o bucket usando chamadas de API autenticadas por credenciais temporárias fornecidas pela função. Ocorre falha em tentativas semelhantes de usar a função feitas por um testador.

Esse fluxo de trabalho tem três etapas básicas:



Crie um perfil na conta de Produção

Primeiramente, você deve usar o AWS Management Console para estabelecer confiança entre a conta de Produção (número de ID 999999999999) e a conta de Desenvolvimento (número de ID 111111111111). Comece criando uma função do IAM denominada UpdateApp. Ao criar a função, você define a conta de Desenvolvimento como uma entidade confiável e especifica uma política de permissões que permite que usuários confiáveis atualizem o bucket `productionapp`.

Conceder acesso ao perfil

Nesta sessão, você modifica a política de grupo de usuários do IAM para negar aos Testadores acesso ao perfil `UpdateApp`. Pois os Testadores têm acesso PowerUser neste cenário e você deve negar explicitamente a capacidade de usar a função.

Teste o acesso alternando funções

Por fim, como Desenvolvedor, você usa a função `UpdateApp` para atualizar o bucket `productionapp` na conta de Produção. Você verá como acessar a função por meio do console da AWS da AWS CLI e da API.

Pré-requisitos

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

- Duas Contas da AWS distintas que você possa usar, uma para representar a conta de Desenvolvimento e outra para representar a conta de Produção.
- Usuários e grupos de usuários na conta de Desenvolvimento criados e configurados da seguinte forma:

Usuário	Grupo de usuários	Permissões
David	Desenvolvedores	Ambos os usuários podem fazer login e usar o AWS Management Console na conta de Desenvolvimento.
Jane	Testadores	

- Você não precisa de usuários ou grupos de usuários criados na conta de Produção.
- Um bucket do Amazon S3 criado na conta de Produção. Você pode chamá-lo de `ProductionApp` neste tutorial, mas como os nomes dos buckets do S3 devem ser globalmente exclusivos, você deve usar um bucket com outro nome.

Crie um perfil na conta de Produção

É possível permitir que os usuários de uma Conta da AWS acessem recursos em outra Conta da AWS. Para fazer isso, crie uma função que defina quem pode acessá-la e quais permissões concede aos usuários que alternam para ela.

Nesta etapa do tutorial, você cria a função na conta de Produção e especifica a conta de Desenvolvimento como uma entidade confiável. Você também limita as permissões da função a acesso somente leitura e gravação para o bucket `productionapp`. Qualquer pessoa com permissão para usar a função pode ler e gravar no bucket `productionapp`.

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

Para obter o ID da Conta da AWS de Desenvolvimento

1. Faça login no AWS Management Console como administrador da conta de Desenvolvimento e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Na barra de navegação, escolha Support (Suporte) e, em seguida, Support Center (Central de Suporte). Seu número (ID) de conta de 12 dígitos conectada no momento aparece no painel de navegação Support Center (Central de Suporte). Para esse cenário, você pode usar o ID de conta 111111111111 para a conta de Desenvolvimento. No entanto, é necessário usar um ID de conta válido se você usar este cenário em seu ambiente de teste.

Para criar uma função na conta de produção que possa ser usada pela conta Desenvolvimento

1. Faça login no AWS Management Console como administrador da conta de Produção 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 `productionapp`. 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:::productionapp`) pelo verdadeiro ARN para o seu bucket do Amazon S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:ListBucket",
        "s3:GetBucketLocation"
       ],
      "Resource": "arn:aws:s3:::productionapp"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "arn:aws:s3:::productionapp/*"
```

```
    }
  ]
}
```

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 `productionapp`. As permissões `GetObject`, `PutObject`, `DeleteObject` concedem aos usuários a capacidade de visualizar, atualizar e excluir o conteúdo no bucket `productionapp`.

4. Resolva os avisos de segurança, erros ou avisos gerais gerados durante a validação de política e depois escolha Next (Avançar).

> **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 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.

6. No painel de navegação, escolha Roles e Create role.
7. Selecione o tipo de perfil Uma Conta da AWS.
8. Em Account ID (ID da conta), digite o ID da conta de Desenvolvimento.

   Este tutorial usa o ID da conta de exemplo **111111111111** para a conta Desenvolvimento. 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 Uso de autenticação multifator (MFA) na AWS.

9. Selecione Next: Permissions (Próximo: permissões) para definir as permissões associadas à função.
10. 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 Next (Próximo).

11. (Opcional) Adicione metadados ao perfil anexando etiquetas como pares de chave-valor. Para obter mais informações sobre como usar etiquetas no IAM, consulte Recursos de etiquetas do IAM.
12. (Opcional) Em Description (Descrição), insira uma descrição para o nova função.
13. Depois de revisar a função, escolha Criar função.

    A função `UpdateApp` 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 a política de grupo dos Desenvolvedores e Testadores, você especifica o ARN da função para conceder ou negar permissões.

Para obter o nome de recurso da Amazon (ARN) para UpdateApp

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

   A conta de produção tem o ID de conta 999999999999, portanto, o ARN da função é `arn:aws:iam::999999999999:role/UpdateApp`. Certifique-se de fornecer o ID verdadeiro da Conta da AWS para a conta de Produção.

Nesse momento, você estabeleceu confiança entre as contas de Produção e de Desenvolvimento. Você fez isso criando uma função na conta de Produção que identifica a conta de Desenvolvimento como uma entidade principal confiável. Você também definiu o que os usuários que passarem para a função `UpdateApp` podem fazer.

Em seguida, modifique as permissões para os grupos de usuários.

Conceder acesso ao perfil

Nesse momento, os membros dos grupos de usuários Testadores e Desenvolvedores têm permissões que os deixam testar livremente as aplicações na conta de Desenvolvimento. Use as seguintes etapas necessárias para adicionar permissões a fim de permitir alternar para a função.

Para modificar o grupo de usuários Desenvolvedores para que possam alternar para a função UpdateApp

1. Acesse como administrador na conta de Desenvolvimento e abra o console do IAM.
2. Escolha User groups (Grupos de usuários) e, em seguida, escolha Developers (Desenvolvedores).
3. Escolha a guia Permissions (Permissões), Add permissions (Adicionar permissões) e Create inline policy (Criar política em linha).
4. Escolha a guia JSON.
5. Adicione a seguinte instrução da política para permitir a ação `AssumeRole` na função `UpdateApp` na conta Produção. Altere *`PRODUCTION-ACCOUNT-ID`* no elemento `Resource` para o ID da Conta da AWS real da conta de produção.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Allow",
       "Action": "sts:AssumeRole",
       "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"
     }
   }
   ```

   O efeito `Allow` permite explicitamente que o grupo Desenvolvedores acesse a função `UpdateApp` na conta Produção. Qualquer desenvolvedor que tente acessar a função é bem-sucedido.
6. Escolha Review policy (Revisar política).
7. Digite um Nome, como, por exemplo, **`allow-assume-S3-role-in-production`**.
8. Escolha Create policy (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 Testadores para garantir que não possam assumir a função. 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 grupo de usuários testadores para negar permissão de assumir a função **`UpdateApp`**

1. Escolha User groups (Grupos de usuários) e Testers (Testadores).
2. Escolha a guia Permissions (Permissões), Add permissions (Adicionar permissões) e Create inline policy (Criar política em linha).
3. Escolha a guia JSON.
4. Adicione a instrução de política a seguir para negar a ação `AssumeRole` na função `UpdateApp`. Altere *`PRODUCTION-ACCOUNT-ID`* no elemento `Resource` para o ID da Conta da AWS real da conta de produção.

   ```
   {
     "Version": "2012-10-17",
     "Statement": {
       "Effect": "Deny",
       "Action": "sts:AssumeRole",
       "Resource": "arn:aws:iam::PRODUCTION-ACCOUNT-ID:role/UpdateApp"
     }
   }
   ```

   O efeito `Deny` nega explicitamente que o grupo Testadores acesse a função `UpdateApp` na conta Produção. Qualquer testador que tentar acessar a função recebe uma mensagem de acesso negado.
5. Escolha Review policy (Revisar política).
6. Digite um nome, como **`deny-assume-S3-role-in-production`**.
7. Escolha Create policy (Criar política).

O grupo de usuários Desenvolvedores agora tem permissões para usar a função `UpdateApp` na conta de produção. O grupo de usuários Testadores é impedido de usar a função `UpdateApp`.

Em seguida, você verá como David, um desenvolvedor, pode acessar o bucket `productionapp` na conta de Produção. David pode acessar o bucket no AWS Management Console, na AWS CLI ou na API do AWS.

Teste o acesso alternando funções

Após concluir as duas primeiras etapas deste tutorial, você tem uma função que concede acesso a um recurso na conta de Produção. Você também tem um grupo de usuários na conta de Desenvolvimento com usuários que têm permissão para usar essa função. Esta etapa discute como testar a alternância para essa função no AWS Management Console, na AWS CLI e na API da AWS.

> **Important**
>
> Você só poderá alternar para uma função após fazer login como um usuário do IAM ou um usuário federado. Além disso, se você iniciar uma instância do Amazon EC2 para executar uma aplicação, a aplicação poderá assumir uma função por meio de seu perfil de instância. Você não pode alternar um perfil ao fazer login como Usuário raiz da conta da AWS.

Alternar funções (console)

Se David precisar trabalhar no ambiente de Produção no AWS Management Console, ele poderá fazê-lo usando Switch Role (Alternar função). Ele especifica o ID da conta ou o alias e o nome da função, e suas permissões mudam imediatamente para as permitidas pela função. Em seguida, ele pode usar o console para trabalhar com o bucket `productionapp`, mas não pode trabalhar com nenhum outro recurso em Produção. Enquanto David usa a função, ele também não pode usar seus privilégios de “power-user” (usuário avançado) na conta de Desenvolvimento. Isso porque apenas um conjunto de permissões pode ser ativado por vez.

> **Important**
>
> Trocar de função usando o AWS Management Console 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 não pode fornecer um valor para `ExternalId`. Para obter mais informações sobre este cenário, consulte Como usar um ID externo ao conceder acesso aos seus recursos da AWS

a terceiros e How to Enable Cross-Account Access to the AWS Management Console no blog de segurança da AWS.

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 Switch Roles (Trocar de função).
- 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 acessa o AWS Management Console usando seu usuário normal que está no grupo de usuários de Desenvolvimento.
2. Ele escolhe o link que oadministrador enviou por e-mailparaele. 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 a função, David deve digitar manualmente o número de ID da conta Produção (999999999999) e o nome da função (`UpdateApp`).

   Além disso, David deseja monitorar quais funções e permissões associadas estão ativas no IAM no momento. Para controlar essas informações, ele digita `PRODUCTION` 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 `UpdateApp` tenha permissões.
5. Quando concluído, David poderá voltar para as permissões originais. Para isso, ele escolhe o nome de exibição da função PRODUÇÃO na barra de navegação e depois escolhe Back to David @ 111111111111 (Voltar para David em 111111111111).
6. Da próxima vez que David desejar alternar funções e escolher o menu Identity (Identidade) na barra de navegação, ele verá a entrada “PRODUCTION” (Produção) no mesmo lugar 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 de Produção na linha de comando, ele poderá fazê-lo 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 a função, ele também não pode usar seus privilégios de “power-user” (usuário avançado) na conta de Desenvolvimento, pois apenas um conjunto de permissões pode estar ativo por 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 mudança de função executando o seguinte comando para mudar para a função `UpdateApp` na conta de Produção. 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/UpdateApp" --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.

> **Note**
>
> 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 Alternância para uma função do IAM (Tools for 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 `UpdateApp`.

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

```
aws s3 ls s3://productionapp
```

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 de Produção no código, ele faz uma chamada do tipo `AssumeRole` para assumir a função de `UpdateApp`. A chamada retorna credenciais temporárias que ele pode usar para acessar o bucket `productionapp` na conta de Produção. Com essas credenciais, David pode fazer chamadas de API para atualizar o bucket `productionapp`. No entanto, ele não pode fazer chamadas de API para acessar nenhum outro recurso na conta de Produção, mesmo tendo permissões de “power-user” (usuário avançado) na conta de Desenvolvimento.

Para assumir uma função

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

   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.
2. Com as credenciais temporárias, David faz uma chamada `s3:PutObject` para atualizar o bucket `productionapp`. Ele pode passar as credenciais para a chamada de API como o parâmetro `AuthParams`. Como as credenciais de função temporárias têm apenas acesso de leitura e gravação para o bucket `productionapp`, todas as outras ações na conta Produção são negadas.

Para obter um exemplo de código (usando Python), consulte Alternância para uma função do IAM (API da AWS).

Recursos relacionados

- Para obter mais informações sobre grupos de usuários e usuários do IAM, consulte Identidades do IAM (usuários, grupos de usuários e funções).
- 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. Em seguida, você modificou uma política de grupo de usuários para controlar quais usuários do IAM podem acessar a função. Como resultado, os desenvolvedores da conta de Desenvolvimento podem fazer atualizações no bucket `productionapp` na conta de Produção usando credenciais temporárias.

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

Neste tutorial, você usará o AWS Management Console 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 AWS Management Console 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 AWS Management Console 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, escolha Policies (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.

```
{
    "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 Next (Avançar).

> **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 Attach policy (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 de usuário do IAM ao AWS Management Console

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 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 O que é ABAC para a AWS?.

> **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

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 Serviços da AWS 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 AWS Billing and Cost Management.

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 ter 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.

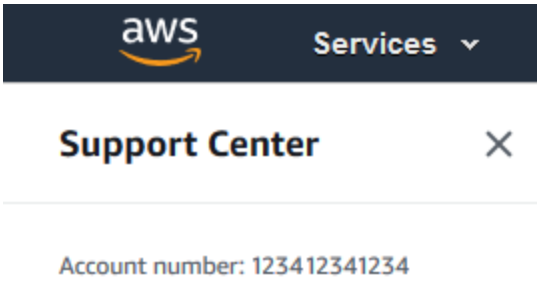
Pré-requisitos

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 AWS Management Console, 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 AWS Management Console. 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.

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

```
    "Statement": [
        {
            "Sid": "TutorialAssumeRole",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::account-ID-without-hyphens: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 Criação de grupos de usuários do IAM e Adicionar e remover usuários de um grupo de usuários 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 AWS Management Console e depois adicione as seguintes tags. Para obter mais informações sobre como criar e marcar um novo usuário, consulte Criação de usuários do IAM (console).

   Usuários de ABAC

Nome do usuário	Chave de tag de usuário	Valor de tag de usuário
`access-Arnav-peg-eng`	`access-project`	`peg`
	`access-team`	`eng`
	`cost-center`	`987654`
`access-Mary-peg-qas`	`access-project`	`peg`
	`access-team`	`qas`

Nome do usuário	Chave de tag de usuário	Valor de tag de usuário
	`cost-center`	`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`.

```
{
 "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 ter mais informações, consulte Chaves de contexto de múltiplos 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 Criação de uma função para delegar 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.

Funções de 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`
		`cost-center = 987654`
			Permite que os engenheiros leiam todos os recursos de engenharia e criem e gerenciem recursos de engenharia do Pegasus.

Projeto Pegasus de controle de qualidade	`access-peg-quality-assurance`	`access-project = peg`	Permite que a equipe de controle de qualidade leia todos os recursos de controle de qualidade e crie e gerencie

Função de trabalho	Nome do perfil	Tags de função	Descrição de função
		`access-team = qas` `cost-center = 987654`	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 engenheir os leiam todos os recursos de engenharia e criem e gerenciem recursos de engenharia do Unicorn.

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 AWS Management Console, consulte Alternância para uma função (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.

Combinações de tags de ABAC para a função **`test-access-peg-eng`**

Valor da tag **`access-project`**	Valor da tag **`access-team`**	Valor da tag **`cost-center`**	Tags adicionais	Comportamento esperado
(none)	(none)	(none)	(none)	Negado porque o valor da tag `access-project` não é correspondente ao valor do perfil de `peg`.
`uni`	`eng`	`987654`	(none)	Negado porque o valor da tag `access-project` não é correspondente ao valor do perfil de `peg`.
`peg`	`qas`	`987654`	(none)	Negado porque o valor da tag `access-team` não é correspondente ao valor do perfil de `eng`.
`peg`	`eng`	`123456`	(none)	Negado porque o valor da tag `cost-center` não é correspondente ao valor do perfil de `987654`.
`peg`	`eng`	`987654`	`owner` = `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.

Valor da tag **access-project**	Valor da tag **access-team**	Valor da tag **cost-center**	Tags adicionais	Comportamento esperado
peg	eng	987654	Name = Jane	Permitido porque todas as três tags necessárias estão presentes e seus valores são correspon dentes 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.

   Funções e tags de ABAC

Nome do 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

Nome do usuário	Nome do perfil	Nome de segredo	Tags de segredo

`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 AWS Management Console, consulte Alternância para uma função (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.

   Comportamento de visualização de segredo de ABAC para cada função

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

Nome do perfil	Nome de segredo	Comportamento esperado
	`test-access-uni-eng`	Negado
	`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 ter 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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TutorialAssumeSpecificRoles",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": [
                "arn:aws:iam::account-ID-without-hyphens:role/access-peg-
engineering",
                "arn:aws:iam::account-ID-without-hyphens: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 AWS Management Console, consulte Alternância para uma função (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.

   Atualização e exclusão de segredos de ABAC para cada função

Nome do perfil	Nome de segredo	Comportamento esperado
`access-peg-engineering`	`test-access-peg-eng`	Permitido
	`test-access-uni-eng`	Negado
	`test-access-uni-qas`	Negado
`access-peg-quality-assurance`	`test-access-peg-qas`	Permitido
	`test-access-uni-eng`	Negado
`access-uni-engineering`	`test-access-uni-eng`	Permitido
	`test-access-uni-qas`	Negado
`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 Determinar se uma solicitação é permitida ou negada em uma conta.

Recursos relacionados

Para obter informações relacionadas, consulte os recursos a seguir:

- O que é ABAC para a AWS?
- Chaves de contexto de condição globais da AWS
- Criação de usuários do IAM (console)
- Criação de uma função para delegar permissões a um usuário do IAM
- Recursos de etiquetas do IAM
- Controlar o acesso a recursos da AWS usando tags
- Alternância para uma função (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 baseado em atributo (ABAC) é uma estratégia de autorização que define 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 O que é ABAC para a AWS?.

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 AWS Management Console. 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 AWS Management Console. 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 Modificar uma função (console).

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.

```
{
    "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://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 `access-project` e `cost-center`. 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 de usuários federados SAML 2.0 ao AWS Management Console.

Etapa 4: Testar a criação de segredos

Federe no AWS Management Console usando a função `access-session-tags`. Para obter mais informações, consulte Habilitar o acesso de usuários federados SAML 2.0 ao AWS Management Console. 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 Determinar se uma solicitação é permitida ou negada em uma conta.

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 AWS Management Console 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:

Criar usuário		Criar e configurar conta de grupo de usuários		
Nome do usuário	Outras instruções	Nome do grupo de usuários	Adicionar usuário como membro	Outras instruções
MFAUser	Escolha apenas a opção para Habilitar acesso	EC2MFA	MFAUser	NÃO anexe quaisquer políticas nem conceda

Criar usuário		Criar e configurar conta de grupo de usuários		
Nome do usuário	Outras instruções	Nome do grupo de usuários	Adicionar usuário como membro	Outras instruções
	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.

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 AWS Management Console 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

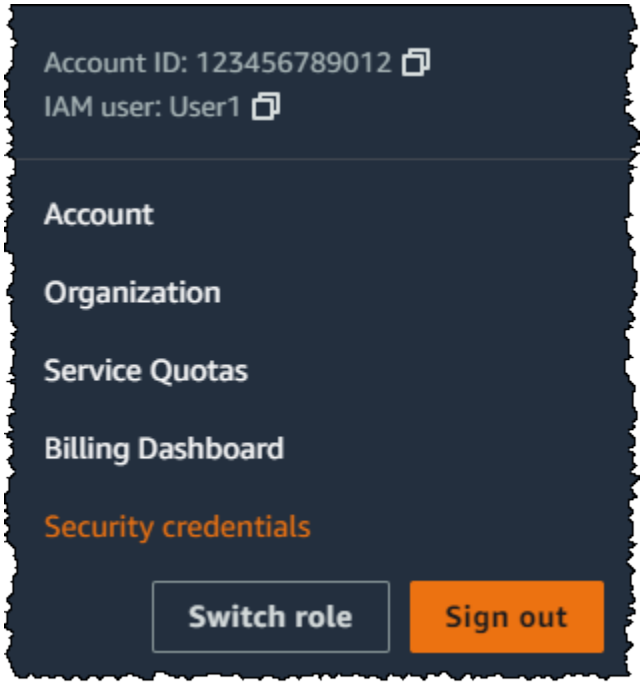
Em seguida, anexe duas políticas ao grupo de usuários de teste do IAM, que serão usadas para conceder as permissões protegidas por MFA.

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 Forçar 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 ter 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:

- Uso de autenticação multifator (MFA) na AWS
- Habilitar dispositivos com MFA para usuários na AWS
- Uso de dispositivos com MFA com sua página de login do IAM

Identidades do IAM (usuários, grupos de usuários e funções)

Tip

Está com problemas para fazer login na AWS? Certifique-se de estar na página de login correta.

- Para fazer login como Usuário raiz da conta da AWS (proprietário da conta), use as credenciais que você configurou ao criar a Conta da AWS.
- Para fazer login como usuário do IAM, use as credenciais concedidas pelo administrador da conta para fazer login no AWS.
- Para fazer login com o usuário do Centro de Identidade do IAM, utilize o URL de login enviado ao seu endereço de e-mail quando você criou o usuário do Centro do Usuário do IAM.

  Para obter ajuda com o login utilizando um usuário do Centro de Identidade do IAM, consulte Fazer login no portal de acesso da AWS portal de acesso da, no Início de Sessão da AWS Guia do usuário .

Para ver tutoriais de login, consulte Como fazer login na AWS no Guia do usuário do Início de Sessão da AWS.

Note

Se você precisar solicitar suporte, não use o link de Feedback nesta página. O feedback que você inserir será recebido pela equipe de AWS Documentation, não pelo AWS Support. Em vez disso, escolha o link Contact Us (Entrar em contato conosco) na parte superior desta página. Ali, encontram-se links para recursos que podem ajudar você a obter o suporte de que precisa.

O Usuário raiz da conta da AWS ou um usuário administrativo da conta pode criar identidades do IAM. Uma identidade do IAM fornece acesso a uma Conta da AWS. Um grupo de usuários do IAM é um conjunto de usuários do IAM gerenciados como uma unidade. Uma identidade do IAM representa um usuário humano ou uma workload programática e pode ser autenticada e autorizada

para executar ações na AWS. Cada identidade do IAM pode ser associada a uma ou mais políticas. As políticas determinam quais ações um usuário, uma função ou o membro de um grupo de usuários pode executar, em quais recursos da AWS e em quais condições.

Usuário raiz da Conta da AWS

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, denominada usuário raiz da Conta da AWS, e é acessada por login com o endereço de e-mail e a senha que você usou para criar a conta.

> **Important**
>
> É altamente recomendável não usar o usuário raiz para tarefas diárias. Proteja as credenciais do usuário raiz e use-as para executar as tarefas que somente ele pode executar. Para obter a lista completa das tarefas que exigem fazer login como usuário raiz, consulte Tarefas que exigem credenciais de usuário raiz.

Usuários do IAM

Um usuário do IAM é uma identidade dentro da Conta da AWS que tem permissões específicas para uma única pessoa ou aplicativo. 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. Antes de criar chaves de acesso, avalie as alternativas às chaves de acesso de longo prazo. Se você tiver casos de uso específicos que exijam chaves de acesso, recomendamos atualizar as chaves de acesso quando necessário. Para ter mais informações, consulte Atualize as chaves de acesso quando necessário para casos de uso que exijam credenciais de longo prazo. Para adicionar usuários do IAM à sua Conta da AWS, consulte Criar um usuário do IAM na sua Conta da AWS.

> **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).

Grupos de usuários do IAM

Um grupo do IAM é uma identidade que especifica uma coleção de usuários do IAM. Não é possível usar um grupo para fazer login. É possível usar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você poderia ter um grupo chamado IAMPublishers e oferecer a esse grupo os tipos de permissões necessárias para publicar workloads.

Perfis do IAM

Um perfil do IAM é uma identidade dentro da Conta da AWS que tem permissões específicas. Ele se assemelha a um usuário do IAM, entretanto, não está associado a uma pessoa específica. É possível assumir temporariamente um perfil do IAM no AWS Management Console alternando perfis. É possível assumir um perfil chamando uma operação de API da AWS CLI ou da AWS, ou usando um URL personalizado. Para obter mais informações sobre os métodos para o uso de perfis, consulte Uso de funções do IAM.

Perfis do IAM com credenciais temporárias são usados nas seguintes situações:

- Acesso de usuário federado: para atribuir permissões a identidades federadas, você pode criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas pelo mesmo. Para mais informações sobre perfis para federação, consulte Criar um perfil para um provedor de identidades de terceiros no Guia do usuário do IAM. Se você usar o IAM Identity Center, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o IAM Identity Center correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte Conjuntos de permissões no Guia do Usuário do AWS IAM Identity Center.
- Permissões temporárias para usuários do IAM: um usuário ou um perfil do IAM pode assumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso entre contas: é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, alguns Serviços da AWS permitem que você anexe uma política diretamente a um recurso (em vez de usar um perfil como proxy). Para conhecer a diferença entre usar perfis e políticas baseadas em recurso para acesso entre contas, consulte Acesso a recursos entre contas no IAM.

- Acesso entre serviços: alguns Serviços da AWS usam atributos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicações no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões da entidade principal de chamada, usando uma função de serviço ou uma função vinculada ao serviço.
  - Encaminhamento de sessões de acesso (FAS): qualquer pessoa que utilizar uma função ou usuário do IAM para realizar ações na AWS é considerada uma entidade principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O recurso FAS utiliza as permissões da entidade principal que chama um AWS service (Serviço da AWS), combinadas às permissões do AWS service (Serviço da AWS) solicitante, para realizar solicitações para serviços downstream. As solicitações de FAS só são feitas quando um serviço recebe uma solicitação que exige interações com outros Serviços da AWS ou com recursos para serem concluídas. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.
  - Perfil de serviço: um perfil de serviço é um perfil do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.
  - Função vinculada a serviço: uma função vinculada a serviço é um tipo de perfil de serviço vinculado a um AWS service (Serviço da AWS). O serviço pode assumir o perfil para executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para perfis vinculados ao serviço.
- Aplicações em execução no Amazon EC2: é possível usar um perfil do IAM para gerenciar credenciais temporárias para aplicações em execução em uma instância do EC2 e fazer solicitações da AWS CLI ou da AWS API. É preferível fazer isso a armazenar chaves de acesso na instância do EC2. Para atribuir um perfil da AWS a uma instância do EC2 e disponibilizá-la para todas as suas aplicações, crie um perfil de instância que esteja anexado a ela. Um perfil de instância contém o perfil e permite que os programas em execução na instância do EC2 obtenham credenciais temporárias. Para mais informações, consulte Usar um perfil do IAM para conceder permissões a aplicações em execução nas instâncias do Amazon EC2 no Guia do usuário do IAM.

Credenciais temporárias no IAM

Como prática recomendada, use credenciais temporárias com usuários humanos e workloads. As credenciais temporárias são usadas principalmente com as funções do IAM, mas também há outras utilidades. Você pode solicitar credenciais temporárias que tenham um conjunto mais restrito de permissões do que um usuário padrão do IAM. Isso evita que você execute acidentalmente tarefas que não são permitidas pelas credenciais mais restritas. Um benefício das credenciais temporárias é que elas expiram automaticamente após um período definido. Você tem o controle sobre a duração da validade das credenciais.

Quando usar perfis do Centro de Identidade do IAM?

Recomendamos que todos os usuários humanos usem o Centro de Identidade do IAM para acessar recursos da AWS. O Centro de Identidade do IAM possibilita melhorias significativas no acesso a recursos da AWS como usuário do IAM. O Centro de Identidade do IAM fornece:

- Um conjunto central de identidades e atribuições
- Acesso a contas em toda a organização da AWS
- Conexão com seu provedor de identidade atual
- Credenciais temporárias
- Autenticação multifator (MFA)
- Configuração de MFA de autoatendimento para usuários finais
- Aplicação administrativa do uso de MFA
- Autenticação única para todos os diretos de Conta da AWS

Para obter mais informações, consulte What is IAM Identity Center (O que é o Centro de Identidade do IAM?) no Guia do usuário do AWS IAM Identity Center.

Quando criar um usuário do IAM (em vez de uma função)

Recomendamos usar somente usuários do IAM para casos de uso sem suporte para usuários federados. 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 Uso do IAM com 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).

Quando criar uma função do IAM (em vez de um usuário)

Crie uma função do IAM nas seguintes situações:

Se você estiver criando uma aplicação que é executada em uma instância do Amazon Elastic Compute Cloud (Amazon EC2) e essa aplicação fizer solicitações à AWS.

> Não crie um usuário do IAM e passe as credenciais do usuário para a aplicação ou incorpore as credenciais na aplicação. Em vez disso, crie uma função do IAM que você possa anexar à instância do EC2 para oferecer credenciais de segurança temporárias às aplicações em execução na instância. Quando um aplicativo usa essas credenciais na AWS, ele pode executar todas as operações permitidas pelas políticas anexadas à função. Para obter detalhes, consulte Uso de uma função do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2.

Você está criando um aplicativo que é executado em um celular e que faz solicitações à AWS.

> Não crie um usuário do IAM para distribuir a chave de acesso do usuário com a aplicação. Em vez disso, use um provedor de identidade como o Login with Amazon, Amazon Cognito, Facebook ou Google para autenticar usuários e mapeá-los para uma função do IAM. O aplicativo pode usar a função para obter credenciais de segurança temporárias que têm as permissões especificadas pelas políticas anexadas à função. Para mais informações, consulte:
>
> - Guia do usuário do Amazon Cognito
> - Federação OIDC

Os usuários em sua empresa são autenticados em sua rede corporativa e desejam usar a AWS sem a necessidade de fazer login novamente, ou seja, você deseja permitir que os usuários se federem na AWS.

> Não crie usuários do IAM. Configure um relacionamento de federação entre o sistema de identidade corporativa e a AWS. Você pode fazer isso de duas maneiras:
>
> - Se o sistema de identidade da sua empresa for compatível com o SAML 2.0, estabeleça confiança entre o sistema de identidade da sua empresa e a AWS. Para ter mais informações, consulte Federação SAML 2.0.
> - Crie e use um servidor de proxy personalizado que converta identidades de usuários da empresa em funções do IAM que forneçam credenciais de segurança temporárias da AWS. Para ter mais informações, consulte Habilitar o acesso do agente de identidades personalizado ao console da AWS.

Comparar as credenciais do Usuário raiz da conta da AWS e as credenciais do usuário do IAM

O usuário raiz é o proprietário da conta e é criado quando a Conta da AWS é criada. Outros tipos de usuários, incluindo usuários do IAM e os usuários do AWS IAM Identity Center, são criados pelo usuário raiz ou pelo administrador da conta. Todos os usuários da AWS têm credenciais de segurança.

Credenciais do usuário raiz.

As credenciais do proprietário da conta permitem acesso total a todos os recursos da conta. Não é possível usar as políticas do IAM para negar ao usuário raiz acesso a recursos. Só é possível usar uma política de controle de serviços (SCP) do AWS Organizations para limitar as permissões do usuário raiz de uma conta-membro. Por isso, recomendamos criar um usuário administrativo no Centro de Identidade do IAM para usar nas tarefas diárias da AWS. Em seguida, proteja as credenciais do usuário raiz e use-as para realizar somente aquelas poucas tarefas de gerenciamento de contas e serviços que exigem fazer login como usuário raiz. Para obter a lista dessas tarefas, consulte Tarefas que exigem credenciais de usuário raiz. Para saber como configurar um administrador para uso diário no Centro de Identidade do IAM, consulte Introdução no Guia do usuário do Centro de Identidade do IAM.

Credenciais do IAM

Um usuário do IAM é uma entidade criada na AWS para representar a pessoa ou serviço que usa o IAM para interagir com recursos da AWS. Esses usuários são identidades dentro da sua Conta da AWS que têm permissões personalizadas específicas. Por exemplo, é possível criar usuários do IAM e fornecer a eles permissões para criar um diretório no Centro de Identidade do IAM. Os usuários do IAM têm credenciais de longo prazo que podem ser usadas para acessar a AWS via AWS Management Console, ou programaticamente via AWS CLI ou as APIs da AWS. Para obter instruções detalhadas sobre como os usuários do IAM fazem login no AWS Management Console, consulte Login no AWS Management Console como usuário do IAM no Guia do usuário de login na AWS.

Em geral, recomendamos que você evite criar usuários do IAM porque eles têm credenciais de longo prazo, como nome de usuário e senha. Em vez disso, exija que seus usuários humanos usem credenciais temporárias para acessar a AWS. É possível usar um provedor de identidade para que seus usuários humanos recebam acesso federado às Contas da AWS assumindo perfis do IAM que forneçam credenciais temporárias. Para gerenciamento de acesso centralizado, recomendamos

usar o Centro de Identidade do IAM para gerenciar o acesso às suas contas e as permissões nessas contas. Você pode 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 IAM? no Guia do usuário do Centro de Identidade 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 denominada usuário raiz da conta da AWS e é acessada pelo login com o endereço de e-mail e a senha usados para criar a conta.

> **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 pode executar. Para obter a lista completa das tarefas que exigem fazer login como usuário raiz, consulte Tarefas que exigem credenciais de usuário raiz.

Os tópicos a seguir detalham as tarefas de gerenciamento associadas ao usuário raiz.

Tarefas

- Habilitar um dispositivo com MFA virtual para o Usuário raiz da conta da AWS (console)
- Habilitar um token de hardware TOTP para o usuário raiz da Conta da AWS (console)
- Habilitar uma chave de segurança FIDO para o usuário raiz da Conta da AWS (console)
- Alterar a senha para o Usuário raiz da conta da AWS
- Redefinição de uma senha de usuário raiz perdida ou esquecida
- Criar chaves de acesso para o usuário raiz
- Excluir chaves de acesso do usuário raiz
- Tarefas que exigem credenciais de usuário raiz
- Solução de problemas com o usuário raiz
- Informações relacionadas

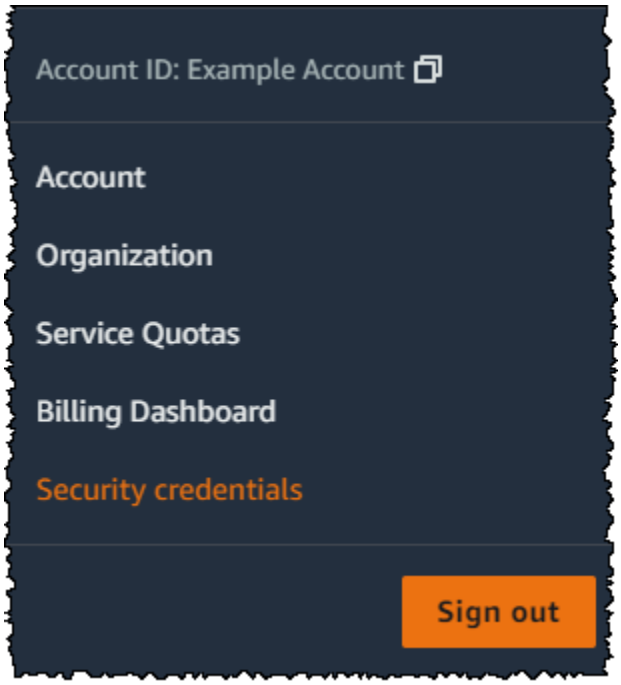
Habilitar um dispositivo com MFA virtual para o Usuário raiz da conta da AWS (console)

Você pode usar o AWS Management Console 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.

Antes de habilitar a MFA para seu usuário raiz, revise as configurações da sua conta e as informações de contato 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 esses fatores alternativos de autenticação, consulte O que acontece se um dispositivo com MFA for perdido ou parar de funcionar?.

Para configurar e habilitar um dispositivo com MFA virtual para uso com seu usuário raiz (console)

1. Faça login no AWS Management Console.
2. No lado direito da barra de navegação, escolha o nome de sua conta e escolha Security credentials (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]), 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. 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. 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 AWS Management Console, consulte Uso de dispositivos com MFA com sua página de login do IAM.

Habilitar um token de hardware TOTP para o usuário raiz da Conta da AWS (console)

Você só pode configurar e habilitar um dispositivo de MFA físico para o seu usuário raiz no AWS Management Console, não na AWS CLI e nem na API da AWS.

Se o seu dispositivo MFA for perdido, roubado ou não funcionar, você ainda poderá fazer login usando fatores alternativos de autenticação. Se você não puder fazer login com o seu dispositivo MFA, poderá fazer login verificando sua identidade usando o e-mail e telefone registrados na conta. Antes de habilitar a MFA para seu usuário raiz, revise as configurações da sua conta e as informações de contato para verificar se você tem acesso ao e-mail e ao número de telefone. Para saber como fazer login usando fatores alternativos de autenticação, consulte O que acontece se um dispositivo com MFA for perdido ou parar de funcionar?. Para desabilitar este recurso, entre em contato com AWS Support.

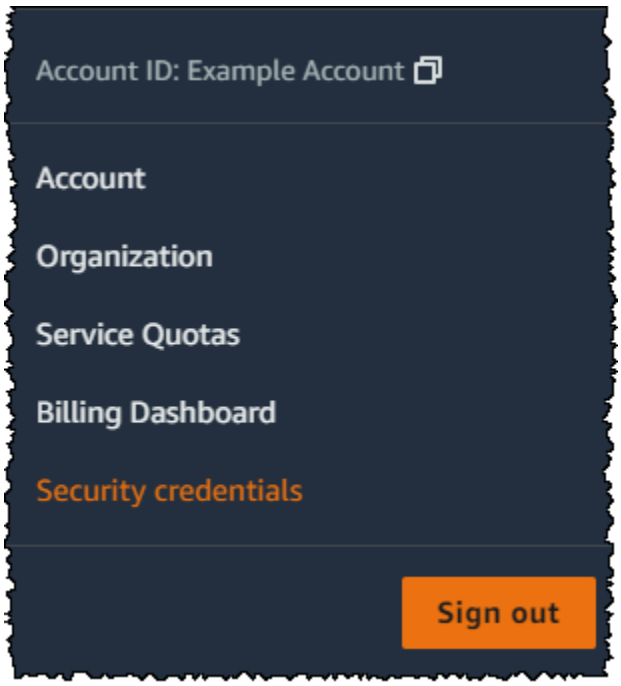
> **Note**
>
> Você pode ver um texto diferente, como Fazer login usando MFA e Solucionar problemas do dispositivo de autenticação. No entanto, os mesmos recursos são fornecidos. Em ambos os casos, se você não puder verificar o endereço de e-mail e o número de telefone de sua conta usando fatores alternativos de autenticação, entre em contato com o AWS Support para desativar sua configuração de MFA.

Para habilitar o dispositivo MFA para seu usuário raiz (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 AWS Management Console 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)).
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.

Habilitar uma chave de segurança FIDO para o usuário raiz da Conta da AWS (console)

Você só pode configurar e habilitar um dispositivo de MFA virtual para o seu usuário raiz no AWS Management Console, não na AWS CLI nem na API da AWS.

Se a chave de segurança FIDO for perdida, roubada ou não estiver funcionando, você ainda poderá fazer login usando outro dispositivo com MFA registrado no mesmo Usuário raiz da conta da AWS. Se você tiver um único dispositivo de MFA registrado, poderá fazer login usando fatores alternativos de identificação. Para saber como fazer login usando fatores alternativos de autenticação, consulte O que acontece se um dispositivo com MFA for perdido ou parar de funcionar?. Para desabilitar este recurso, entre em contato com AWS Support.

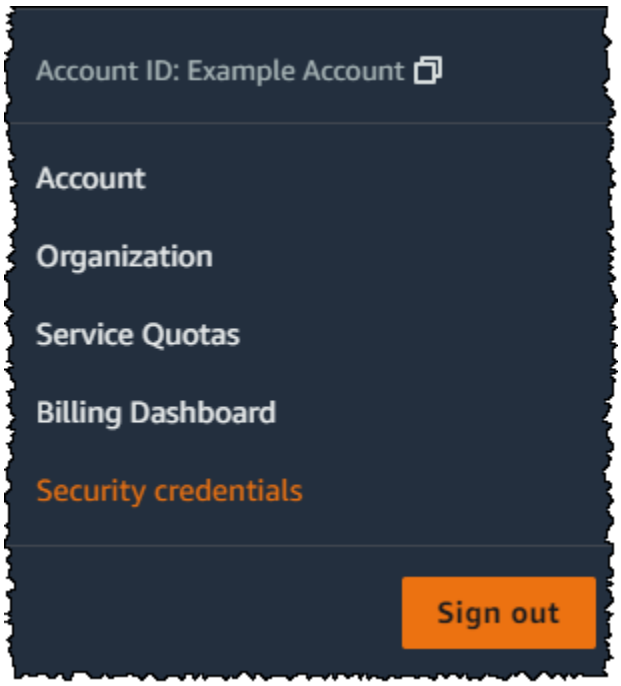
> **Note**
>
> Você não deve escolher nenhuma das opções disponíveis no pop-up do Goggle Chrome que solicite Verify your identity with amazon.com (Confirmar sua identidade na amazon.com). Você só precisa tocar na chave de segurança.

Para habilitar a chave FIDO para o usuário raiz (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 AWS Management Console como usuário raiz no Guia do usuário do Início de Sessão da AWS.

2. No lado direito da barra de navegação, selecione o nome da conta e escolha Security credentials (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)).
4. Escolha Assign MFA device (Atribuir dispositivo de MFA).
5. No assistente, digite um Device name (Nome de dispositivo), escolha Security Key (Chave de segurança) e escolha Next (Avançar).
6. Insira a chave de segurança FIDO na porta USB do computador.

7. Toque na chave de segurança FIDO.

A chave de segurança FIDO está pronta para ser usada com a AWS. A próxima vez que usar as credenciais de usuário raiz para fazer login, você deverá tocar na chave de segurança FIDO para concluir o processo de login.

Para obter ajuda na solução de problemas de chaves de segurança FIDO, consulte Solução de problemas de chaves de segurança FIDO.

Alterar a senha para o Usuário raiz da conta da AWS

Você pode 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 Redefinição de 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.

AWS Management Console

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. Use seu endereço de e-mail e a senha da Conta da AWS para entrar no AWS Management Console como seu Usuário raiz da conta da AWS.
2. No canto superior direito do console, selecione o nome ou número de sua conta e, em seguida, selecione Conta.
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.

   A AWS 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.

> **Note**
>
> A AWS está implementando melhorias no processo de login. Uma dessas melhorias é impor uma política de senha mais segura para sua conta. Se a AWS tiver atualizado sua conta, você deverá atender à política de senha descrita anteriormente. Se a AWS ainda não tiver atualizado sua conta, a AWS não aplicará essa política ainda. No entanto, é altamente recomendável seguir as diretrizes para criar uma senha mais segura.

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 AWS Management Console.

Redefinição de 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 AWS Management Console.

Para redefinir sua senha de usuário raiz:

1. Use o endereço de e-mail da sua Conta da AWS para começar a fazer login no AWS Management Console como usuário raiz e escolha Next (Próximo).

> **Note**
>
> Se você estiver conectado ao AWS Management Console 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 fizer login por meio do Portal de acesso da 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 AWS Management Console 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. Quando 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.

AWS Management Console

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. Use seu endereço de e-mail e a senha da Conta da AWS para entrar em Conceitos básicos do AWS Management Console como seu Usuário raiz da conta da AWS.
2. No canto superior direito do console, selecione o nome de sua conta e, em seguida, selecione Credenciais de segurança.
3. Na seção Access keys (Chaves de acesso), escolha Create access key (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. Escolha Done (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 AWS Management Console. 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 do usuário raiz

É possível usar o AWS Management Console, a AWS CLI ou a API da AWS para excluir as chaves de acesso do usuário raiz.

AWS Management Console

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. Use seu endereço de e-mail e a senha da Conta da AWS para entrar em Conceitos básicos do AWS Management Console como seu Usuário raiz da conta da AWS.
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

> **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 AWS Support. 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.

Recomendamos que configurar um usuário administrativo no AWS IAM Identity Center 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.

Tarefas de gerenciamento de contas

- Altere as configurações da conta. Isso inclui o nome da conta, endereço de e-mail, senha do usuário raiz e chaves de acesso do usuário raiz. Outras configurações de conta, como informações de contato, preferência de moeda de pagamento e Regiões da AWS, não exigem credenciais de usuário raiz.
- 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.
- Feche sua Conta da AWS.

  Para obter mais informações, consulte os tópicos a seguir.

  - Como atribuo a propriedade da minha Conta da AWS a outra entidade? .
  - Como faço para fechar minha Conta da AWS?
  - Fechar uma Conta da AWS independente

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).

Tarefas do AWS GovCloud (US)

- Cadastre-se na AWS GovCloud (US).

- Solicitar as chaves de acesso do usuário raiz da conta AWS GovCloud (US) ao AWS Support.
- Caso uma chave do AWS Key Management Service se torne incontrolável, será possível recuperá-la entrando em contato com o AWS Support como usuário raiz.

Tarefa do Amazon EC2

- Registrado como um vendedor no Marketplace de instâncias reservadas.

Tarefa do Amazon Mechanical Turk

- Vincule sua Conta da AWS à sua conta do MTurk Requester.

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.

Tarefa do Amazon Simple Queue Service

- Editar ou excluir uma política de recurso do Amazon SQS que negue todas as entidade principais.

Solução de 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.

Não consigo realizar as tarefas que espero poder realizar quando estou conectado como usuário raiz da conta

Se você não conseguir concluir tarefas quando estiver conectado como usuário raiz da conta, sua conta pode ser membro de uma organização em AWS Organizations. Nesse caso, e se o administrador da sua organização usou uma política de controle de serviços (SCP) para limitar as permissões da sua conta, então todos os usuários, incluindo o usuário raiz, serão afetados. Para obter mais informações, 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 Redefinição de 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, primeiro tente recuperar o acesso ao e-mail usando uma das seguintes opções:

- Se você for o proprietário do domínio do endereço de e-mail, poderá restaurar um endereço de e-mail excluído. Também é possível configurar um catch-all para sua conta de e-mail, que "capture todas" as mensagens enviadas para endereços de e-mail que não existam mais no servidor e as redirecione 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.

Informações relacionadas

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.

Um usuário do AWS Identity and Access Management (IAM) é uma entidade que você cria na AWS. O usuário do IAM representa o usuário humano ou a workload que utiliza o usuário do IAM para interagir com a AWS. Um usuário na AWS 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 AWS Management Console.
- 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 AWS Management Console. Desabilitar a senha (acesso ao console) de um usuário do IAM impede que ele faça login no AWS Management Console usando suas credenciais de login. Isso não altera as permissões do usuário nem o impede de acessar o console usando uma função assumida.
- 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 AWS Certificate Manager (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 AWS Certificate Manager na Referência geral da AWS.

Você pode escolher as credenciais certas para o seu usuário do IAM. Quando você usa o AWS Management Console 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 AWS Management Console. Definir uma política de senha para impor uma complexidade mínima para a senha. Permitir que os usuários troquem 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 precisam 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. As melhores práticas de segurança são remover senhas e chaves de acesso quando os usuários não precisam 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 os procedimentos sobre como criar usuários e conceder as credenciais e permissões iniciais, consulte Criar um usuário do IAM na sua Conta da AWS. Para obter os procedimentos sobre como alterar as permissões para usuários existentes, consulte Alteração de permissões de um usuário do IAM. Para obter os procedimentos sobre como alterar a senha ou chaves de acesso do usuário, consulte Gerenciar senhas de usuários na AWS e Gerenciamento de chaves de acesso de 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 IAM.

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.

Criar um usuário do IAM na sua Conta da AWS

Follow us on Twitter

> **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.

> **Note**
>
> Se você encontrou esta página porque está procurando informações sobre a API Product Advertising para vender produtos da Amazon em seu site, consulte a Documentação da Product Advertising API 5.0.
> Se você chegou a esta página do console do IAM, é possível que sua conta não inclua usuários do IAM, mesmo que você esteja conectado. Você pode estar conectado como o Usuário raiz da conta da AWS usando uma função ou conectado com credenciais temporárias. Para saber mais sobre essas identidades do IAM, consulte Identidades do IAM (usuários, grupos de usuários e funções).

O processo de criar um usuário e habilitá-lo para executar tarefas de trabalho consiste nas seguintes etapas:

1. Crie o usuário no AWS Management Console, na AWS CLI, no Tools for Windows PowerShell ou usando uma operação da API da AWS. Se você criar o usuário no AWS Management Console, as etapas de 1 a 4 serão tratadas automaticamente com base em suas opções. Se você criar os usuários de forma programática, você deverá executar cada uma dessas etapas individualmente.
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 AWS Management Console, crie uma senha para o usuário. Desabilitar o acesso ao console para um usuário impede que ele faça login no AWS Management Console 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 AWS Management Console, não crie chaves de acesso.

3. Forneça ao usuário permissões para executar as tarefas necessárias adicionando o usuário a um ou mais grupos. Você também pode conceder permissões associando políticas de permissões diretamente ao usuário. No entanto, recomendamos que você coloque seus usuários em grupos e gerencie permissões por meio de políticas anexadas a esses grupos. Você também pode usar um limite de permissões para limitar as permissões que um usuário pode ter, embora isso não seja comum.

4. (Opcional) Adicione metadados ao usuário anexando tags. Para obter mais informações sobre como usar etiquetas no IAM, consulte Recursos de etiquetas do IAM.
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 ter 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 AWS Management Console.
7. (Opcional) Conceda aos usuários permissões para gerenciar suas próprias credenciais de segurança. (Por padrão, os usuários não têm permissões para gerenciar suas próprias credenciais.) Para ter mais informações, consulte Permitir que os usuários do IAM alterem as próprias senhas.

Para obter informações sobre as permissões de que você precisa para criar um usuário, consulte Permissões necessárias para acessar recursos do IAM.

Tópicos

- Criação de usuários do IAM (console)
- Criação de usuários do IAM (AWS CLI)
- Criação de usuários do IAM (API da AWS)

Criação de usuários do IAM (console)

É possível usar o AWS Management Console para criar usuários do IAM.

Para criar um usuário 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, selecione o serviço do IAM.
3. No painel de navegação, selecione Usuários e Adicionar usuários.
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.

> **Note**
>
> 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. 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. Selecione Fornecer acesso do usuário ao AWS Management Console: opcional Isso produz credenciais de login no AWS Management Console para o novo usuário.

   Será exibida uma mensagem perguntado se você está fornecendo acesso ao console para uma pessoa. Recomendamos criar usuários no Centro de Identidade do IAM em vez de no IAM.

   - Para alternar para a criação do usuário no Centro de Identidade do IAM, selecione Especificar um usuário no Centro de Identidade.

     Caso não tenha habilitado o Centro de Identidade do IAM, ao selecionar essa opção, você será encaminhado à página de serviço no console para que possa habilitar o serviço. Para obter detalhes sobre esse procedimento, consulte Comece a usar tarefas comuns no Centro de Identidade do IAM no Guia de usuário do AWS IAM Identity Center

     Caso tenha habilitado o Centro de Identidade do IAM, ao selecionar essa opção, você será encaminhado à página Especificar detalhes do usuário no Centro de Identidade do IAM. Para obter detalhes sobre esse procedimento, consulte Adicionar usuários no Guia do usuário do AWS IAM Identity Center
   - Se não puder usar o Centro de Identidade do IAM, selecione Quero criar um usuário do IAM e continue seguindo esse procedimento.

   a. Em Senha do console, selecione uma das opções a seguir:

- Senha gerada automaticamente: o usuário obtém uma senha gerada de maneira aleatória que atende à política de senha da conta. É possível visualizar ou baixar a senha ao acessar a página Recuperar senha.
- Senha personalizada: o usuário recebe a senha que você inserir na caixa.

b. (Opcional) A opção Os usuários deverão criar uma senha no próximo login (recomendado) é selecionada por padrão para garantir que o usuário seja forçado a alterar a senha na primeira vez em que fizer login.

> **Note**
>
> Se um administrador tiver habilitado a configuração Permitir que os usuários alterem sua própria senha da política de senha da conta, essa caixa de seleção não terá nenhum efeito. Caso contrário, ele associa automaticamente uma política AWS gerenciada nomeada `IAMUserChangePassword` aos novos usuários. A política concede a eles permissão para alterar suas próprias senhas.

6. Escolha Próximo.
7. Na página Definir permissões, especifique como deseja atribuir permissões a esse novo usuário. Selecione uma das três opções a seguir:

   - Adicionar usuário ao grupo: selecione esta opção se você desejar atribuir o usuário a um ou mais grupos que já tenham políticas de permissões. O IAM exibe uma lista dos grupos em sua conta, junto com suas políticas anexadas. Você pode selecionar um ou mais grupos existentes ou selecionar Criar grupo para criar um novo grupo. Para ter mais informações, consulte Alteração de permissões de um usuário do IAM.
   - Copiar permissões: selecione esta opção para copiar todas as associações de grupo, políticas gerenciadas anexadas e políticas em linha incorporadas e limites de permissões de um usuário existente para o novo usuário. O IAM exibe uma lista dos usuários em sua conta. Selecione a opção cujas permissões atendam melhor às necessidades do novo usuário.
   - Anexar políticas diretamente: selecione esta opção para ver uma lista das políticas gerenciadas pela AWS e pelo cliente em sua conta. Selecione as políticas que deseja anexar ao usuário ou selecione Criar política para abrir uma nova aba no navegador e criar uma nova política. Para obter mais informações, consulte a etapa 4 no procedimento Criação de políticas do IAM. Depois de criar a política, feche essa guia e retorne à guia original para adicionar a política ao usuário.

> **Tip**
>
> Sempre que possível, anexe suas políticas a um grupo e torne os usuários membros dos grupos apropriados.

8. (Opcional) Defina um limite de permissões. Este é um recurso avançado.

   Abra a seção Limite de permissões e selecione Usar um limite de permissões para controlar o número máximo de permissões. O IAM exibe uma lista das políticas gerenciadas pela AWS e pelo cliente em sua conta. Selecione a política a ser usada para o limite de permissões ou selecione Criar política para abrir uma nova guia no navegador e criar uma nova política. Para obter mais informações, consulte a etapa 4 no procedimento Criação de políticas do IAM. Depois de criar a política, feche essa guia e retorne à guia original para selecionar a política a ser usada para o limite de permissões.

9. Escolha Próximo.

10. (Opcional) Na página Revisar e criar, em Etiquetas, selecione Adicionar nova etiqueta para adicionar metadados ao usuário anexando etiquetas aos pares de chave-valor. Para obter mais informações sobre como usar etiquetas no IAM, consulte Recursos de etiquetas do IAM.

11. Revise todas as escolhas feitas até esse ponto. Quando você estiver pronto para continuar, selecione Criar usuário.

12. Na página Recuperar senha, obtenha a senha atribuída ao usuário:

    - Selecione Exibir ao lado da senha para visualizar a senha do usuário e poder gravá-la manualmente.
    - Selecione Baixar .csv para baixar as credenciais de login do usuário como um arquivo. csv que você poderá salvar em um local seguro.

13. Selecione Instruções de login de e-mail. Seu cliente de e-mail local é aberto com um rascunho que você pode personalizar e enviar ao usuário. O modelo de e-mail inclui os seguintes detalhes de cada usuário:

    - Nome do usuário
    - URL para a página de login da conta. Use o exemplo a seguir, substituindo o número de ID ou alias da conta:

    ```
    https://AWS-account-ID or alias.signin.aws.amazon.com/console
    ```

> **Important**
>
> A senha do usuário não é incluída no e-mail gerado. É necessário fornecer a senha ao usuário de forma que esteja em conformidade com as diretrizes de segurança de sua organização.

14. Se o usuário também precisar de chaves de acesso para acesso programático, consulte Gerenciamento de chaves de acesso de usuários do IAM.

Criação de usuários do IAM (AWS CLI)

Você pode usar a AWS CLI para criar um usuário do IAM.

Para criar um usuário do IAM (AWS CLI)

1. Criar um usuário.

   - aws iam create-user

2. (Opcional) Forneça ao usuário acesso ao AWS Management Console. Isso requer uma senha. Você também deve oferecer ao usuário o URL da página de login da sua conta.

   - aws iam create-login-profile

3. (Opcional) Forneça ao usuário acesso programático. Isso requer chaves de acesso.

   - aws iam create-access-key
   - Tools for Windows PowerShell: New-IAMAccessKey
   - API do IAM: CreateAccessKey

   > **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.

4. Adicione o usuário a um ou mais grupos. Os grupos que você especificar devem ter políticas anexadas que concedam as permissões apropriadas para o usuário.

   - aws iam add-user-to-group

5. (Opcional) Anexe uma política ao usuário que defina as permissões do usuário. Observação: recomendamos que você gerencie as permissões de usuário ao adicionar o usuário a um grupo e anexando uma política ao grupo, em vez de anexar diretamente a um usuário.

   - aws iam attach-user-policy

6. (Opcional) Adicione atributos personalizados ao usuário anexando etiquetas. Para ter mais informações, consulte Gerenciamento de etiquetas em usuários do IAM (AWS CLI ou API da AWS).

7. (Opcional) Conceda ao usuário permissão para gerenciar suas próprias credenciais de segurança. Para ter mais informações, consulte AWS: permite que os usuários do IAM autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança.

Criação de usuários do IAM (API da AWS)

Você pode usar a API da AWS para criar um usuário do IAM.

Para criar um usuário do IAM na (API da AWS)

1. Criar um usuário.

   - CreateUser

2. (Opcional) Forneça ao usuário acesso ao AWS Management Console. Isso requer uma senha. Você também deve oferecer ao usuário o URL da página de login da sua conta.

   - CreateLoginProfile

3. (Opcional) Forneça ao usuário acesso programático. Isso requer chaves de acesso.

   - CreateAccessKey

   > **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.

4. Adicione o usuário a um ou mais grupos. Os grupos que você especificar devem ter políticas anexadas que concedam as permissões apropriadas para o usuário.

   - AddUserToGroup

5. (Opcional) Anexe uma política ao usuário que defina as permissões do usuário. Observação: recomendamos que você gerencie as permissões de usuário ao adicionar o usuário a um grupo e anexando uma política ao grupo, em vez de anexar diretamente a um usuário.

   - AttachUserPolicy

6. (Opcional) Adicione atributos personalizados ao usuário anexando etiquetas. Para ter mais informações, consulte Gerenciamento de etiquetas em usuários do IAM (AWS CLI ou API da AWS).

7. (Opcional) Conceda ao usuário permissão para gerenciar suas próprias credenciais de segurança. Para ter mais informações, consulte AWS: permite que os usuários do IAM autenticados por MFA gerenciem suas próprias credenciais na página Credenciais de segurança.

Controlar o acesso de usuário do IAM ao AWS Management Console

Os usuários do IAM com permissão que fizerem login na sua Conta da AWS por meio do AWS Management Console 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 AWS Management Console. 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 AWS Management Console

Você cria uma senha para cada usuário do IAM que precisa acessar o AWS Management Console. 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 Gerenciar senhas de usuários na AWS.

Você pode impedir que um usuário do IAM acesse o AWS Management Console removendo a senha dele. Isso impede que ele faça login no AWS Management Console 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 (usuários, grupos de usuários e funções). Para obter informações sobre o uso de políticas para definir permissões, consulte Gerenciamento de acesso para recursos da AWS.

Fóruns de discussão da AWS

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 AWS Management Console. As políticas para esse usuário (ou qualquer grupo ao qual o usuário pertence) controlam o que o usuário pode fazer no AWS Management Console. 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.

Como os usuários do IAM fazem login na AWS

Para fazer login no AWS Management Console 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 AWS Management Console, 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 AWS Management Console 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 agente 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.

Uso de dispositivos com MFA com sua página de login do IAM

Os usuários configurados com dispositivos de autenticação multifator (MFA) devem usar esses dispositivos para fazer login no AWS Management Console. Depois que o usuário insere suas credenciais de login, a AWS verifica a conta do usuário para ver se a MFA é necessária. Os tópicos a seguir fornecem informações sobre como os usuários concluem o login quando a MFA é necessária.

Tópicos

- Fazer login com vários dispositivos de MFA habilitados
- Fazer login com uma chave de segurança FIDO
- Fazer login com um dispositivo com MFA virtual
- Fazer login com um token de hardware TOTP

Fazer login com vários dispositivos de MFA habilitados

Ao fazer login no AWS Management Console 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.

Fazer login com uma 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 obter 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 segurança FIDO.

Fazer login com um 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 AWS Management Console. 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 AWS Management Console, 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 dispositivos com MFA virtuais e de hardware.

Fazer login com um token de hardware 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 AWS Management Console. 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 AWS Management Console, 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 dispositivos com MFA virtuais e de hardware.

Gerenciar usuários 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 AWS IAM Identity Center, consulte Manage identities in IAM Identity Center (Gerenciar identidades no Centro de Identidade do IAM) no Guia do usuário do AWS IAM Identity Center 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 em sua conta ou em um grupo de usuários, ou listar todos os grupos de usuários dos quais um usuário é membro. 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 Alteração de 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, Edição de políticas do IAM e Exclusão de 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 obter mais informações, consulte Políticas gerenciadas pela AWS. Para obter mais informações sobre políticas gerenciadas pela AWSque são projetadas para funções de trabalho específicas, consulte Políticas gerenciadas pela AWS para funções de trabalho.

Para saber como validar políticas do IAM, consulte Validação de políticas 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 Gerenciamento de senhas de usuários do IAM,

Tópicos

- Visualizar acesso do usuário
- Listagem de usuários do IAM
- Renomeação de um usuário do IAM
- Exclusão de um usuário do IAM
- Desativar um usuário do IAM

Visualizar acesso do usuário

Antes de excluir 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 as informações do último acesso.

Listagem de usuários do IAM

Você pode listar os usuários do IAM na sua Conta da AWS ou em um grupo de usuários do IAM específico e listar todos os grupos de usuários em que um usuário está. 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 na conta

- AWS Management Console: No painel de navegação, selecione Users (Usuários). O console exibe os usuários na sua Conta da AWS.
- AWS CLI: aws iam list-users
- API da AWS: ListUsers

Para listar os usuários em um grupo de usuários específico

- AWS Management Console: no painel de navegação, selecione User groups (Grupos de usuários), escolha o nome do grupo de usuários e, depois, escolha a guia Users (Usuários).
- AWS CLI: aws iam get-group
- API da AWS: GetGroup

Para listar todos os grupos de usuários em que um usuário se encontra

- AWS Management Console: No painel de navegação, selecione Usuários, escolha o nome do usuário e, então, selecione a guia Grupos.
- AWS CLI: aws iam list-groups-for-user
- API da AWS: ListGroupsForUser

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 permanecerá nos mesmos grupos de usuários 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

Exclusão de um usuário do IAM

Você pode excluir um usuário do IAM da Conta da AWS se alguém sair da empresa. Se o usuário se ausentar temporariamente, você poderá desativar seu acesso em vez de excluí-lo da conta, conforme descrito em Desativar um usuário do IAM.

Tópicos

- Exclusão de um usuário do IAM (console)
- Exclusão de um usuário do IAM (AWS CLI)

Exclusão de um usuário do IAM (console)

Ao usar o AWS Management Console para excluir um usuário do IAM, o IAM exclui automaticamente as seguintes informações para você:

- O usuário
- Quaisquer associações do grupo de usuários, ou seja, o usuário é removido de todos os grupos de usuários do IAM dos quais ele é membro
- Todas as senhas associadas ao usuário
- Todas as chaves de acesso pertencentes ao usuário
- Todas as políticas em linha incorporadas no usuário (políticas aplicadas a um usuário por meio de permissões do grupo de usuários não são afetadas)

> **Note**
>
> O IAM remove todas as políticas gerenciadas anexadas ao usuário quando você exclui o usuário, mas não exclui as políticas gerenciadas.

- Todos os dispositivos MFA associados

Para excluir um usuário do IAM (console)

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.

2. No painel de navegação esquerdo, escolha Users (Usuários) e marque a caixa de seleção ao lado do nome do usuário que você deseja excluir.
3. Na parte superior da página, escolha Delete (Excluir).
4. Na caixa de diálogo de confirmação, insira o nome de usuário no campo de entrada de texto para confirmar a exclusão do usuário. Escolha Delete (Excluir).

Exclusão de um usuário do IAM (AWS CLI)

Ao contrário do AWS Management Console, ao excluir um usuário com a AWS CLI, você tem que excluir os itens anexados ao usuário manualmente. Este procedimento ilustra o processo.

Para excluir um usuário da conta (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 de usuários.

   `aws iam list-groups-for-user` (para listar os grupos de usuários 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.

```
{
  "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).

```
{
  "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"}
          }
       }
   ]
}
```

Alteração de 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)
- Remoção de uma política de permissões de um usuário (console)
- Remoção do 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 as 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 Gerar políticas com base na atividade de acesso.

Adição de permissões a um usuário (console)

O IAM oferece três formas de adicionar políticas de permissões a um usuário:

- Adicionar usuário ao grupo: torne o usuário um membro de um grupo. As políticas do grupo são anexadas ao usuário.
- Copiar permissões de um usuário existente: copie todas as associações de grupo, políticas gerenciadas anexadas, políticas em linha e quaisquer limites de permissões existentes do usuário-fonte.
- Anexar políticas diretamente ao usuário: anexe uma política gerenciada diretamente ao usuário. Para facilitar o gerenciamento de permissões, anexe suas políticas a um grupo e torne os usuários membros dos grupos apropriados.

> **Important**
>
> Se o usuário tiver um limite de permissões, você não poderá adicionar mais permissões a um usuário além das permitidas pelo limite de permissões.

Adição de permissões incluindo o usuário em um grupo

A adição de usuário a um grupo afeta o usuário imediatamente.

Para adicionar permissões a um usuário incluindo-o em um grupo

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Analise as associações a grupos atuais para os usuários na coluna Grupos do console. Se necessário, adicione a coluna à tabela de usuários concluindo as etapas a seguir:
   1. Acima da tabela, no canto direito, selecione o símbolo de configurações ( ).
   2. Na caixa de diálogo Gerenciar colunas, selecione a coluna Grupos. Você também pode desmarcar a caixa de seleção para qualquer cabeçalhos de coluna que você não deseje exibir na tabela de usuários.
   3. Escolha Fechar para retornar à lista de usuários.

   A coluna Grupos mostra os grupos aos quais o usuário pertence. A coluna inclui os nomes de até dois grupos. Se o usuário for membro de três ou mais grupos, os primeiros dois grupos serão mostrados (classificados alfabeticamente) e o número de associações a grupos adicionais será incluído. Por exemplo, se o usuário pertencer aos grupos A, B, C e D, o campo conterá o valor Grupo A, grupo B+2. Para ver o número total de grupos aos quais o usuário pertence, você pode adicionar a coluna Contagem de grupos à tabela de usuários.
4. Escolha o nome do usuário cujas permissões você deseja modificar.
5. Selecione a guia Permissões e escolha Adicionar permissões. Escolha Add user to group.
6. 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.
7. (Opcional) Além de selecionar a partir de grupos existentes, você pode escolher Criar grupo para definir um novo grupo:
   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 Criação de políticas do IAM.

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.

8. Escolha Próximo para ver uma lista de associações de grupos a serem adicionadas ao usuário. Em seguida, selecione Adicionar permissões.

Adição de permissões por cópia de outro usuário

A cópia de permissões afeta o usuário imediatamente.

Para adicionar permissões a um usuário copiando permissões de outro usuário

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Usuários no painel de navegação, selecione o nome do usuário cujas permissões deseja modificar e escolha a guia Permissões.
3. Escolha Adicionar permissões e, em seguida, escolha Copiar permissões de um usuário existente. A lista exibe os usuários disponíveis junto com suas associações a grupos e políticas anexadas. Se a lista completa de grupos ou políticas não se encaixar em uma única linha, você poderá escolher o link para e ***n*** mais. Ao fazer isso, uma nova guia do navegador será aberta exibindo a lista completa de políticas (guia Permissões) e grupos (guia Grupos).

4. Selecione o botão ao lado do usuário cujas permissões você deseja copiar.
5. Escolha Próximo para ver uma lista de alterações a serem feitas para o usuário. Em seguida, selecione Adicionar permissões.

Adição de permissões anexando políticas diretamente ao usuário

A anexação de políticas afeta o usuário imediatamente.

Para adicionar permissões a um usuário anexando diretamente políticas gerenciadas

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Usuários no painel de navegação, selecione o nome do usuário cujas permissões deseja modificar e escolha a guia Permissões.
3. Escolha Adicionar permissões e depois escolha Anexar políticas diretamente.
4. Marque uma ou mais caixas de seleção para as políticas gerenciadas que você deseja anexar ao usuário. 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 marque a caixa de seleção da nova política para anexá-la ao usuário. Para obter mais informações, consulte Criação de políticas do IAM.
5. Escolha Próximo para ver a lista de políticas a serem anexadas ao usuário. Em seguida, selecione Adicionar permissões.

Definição do limite de permissões para um usuário

A definição de um limite de permissões afeta o usuário imediatamente.

Para definir o limite de permissões para um usuário

1. Faça login no AWS Management Console 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 cujo limite de permissões você deseja alterar.
4. Escolha a guia Permissions (Permissões). Se necessário, abra a seção Limite de permissões e escolha Definir limite de permissões.
5. Selecione a política que você deseja usar para o limite de permissões.

6. Escolha Definir limite.

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 de permissões afeta o usuário imediatamente.

Para editar as políticas gerenciadas anexadas a um usuário

1. Faça login no AWS Management Console 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 cuja política de permissões você deseja alterar.
4. Escolha a guia Permissions (Permissões). Se necessário, abra a seção Políticas de permissões.
5. Escolha o nome da política que deseja editar para visualizar os detalhes da política. Escolha a guia Utilização da política para visualizar outras entidades que poderão ser afetadas se você editar a política.
6. Escolha a guia Permissões e revise as permissões concedidas pela política. Em seguida, escolha Editar política.
7. Edite a política e resolva as recomendações de validação de política. Para obter mais informações, consulte Edição de políticas do IAM.
8. Escolha Revisar política, revise o resumo da política e escolha Salvar alterações.

Alteração do limite de permissões para um usuário

A alteração de um limite de permissões afeta o usuário imediatamente.

Para alterar a política usada para definir o limite de permissões para um usuário

1. Faça login no AWS Management Console 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 cujo limite de permissões você deseja alterar.
4. Escolha a guia Permissions (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 Definir limite.

Remoção de uma política de permissões de um usuário (console)

A remoção de uma política afeta o usuário imediatamente.

Para remover permissões para usuários do IAM

1. Faça login no AWS Management Console 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 cujo limite de permissões você deseja remover.
4. Escolha a guia Permissions (Permissões).
5. Se você deseja remover permissões removendo uma política existente, visualize Tipo para entender como o usuário está recebendo essa política antes de escolher Remover para removê-la:

   - 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 for uma política incorporada em linha, selecionar X removerá a política do IAM. As políticas em linha anexadas diretamente a um usuário existem somente para esse usuário.

Remoção do limite de permissões de um usuário (console)

A remoção de um limite de permissões afeta o usuário imediatamente.

Para remover o limite de permissões de um usuário

1. Faça login no AWS Management Console 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 cujo limite de permissões você deseja remover.
4. Escolha a guia Permissions (Permissões). Se necessário, abra a seção Limite de permissões e, em seguida, escolha Remover limite.
5. Escolha Remover limite para confirmar que deseja remover o limite de permissões.

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:

- Adicionar e remover usuários de um grupo de usuários do IAM
- Adicionar e remover permissões de identidade do IAM

Gerenciar senhas de usuários na AWS

Você pode gerenciar as senhas de usuários do IAM em sua conta. Usuários do IAM precisam de senhas para acessar o AWS Management Console. 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 Credenciais de segurança da AWS.

Índice

- Permitir que os usuários do IAM alterem as próprias senhas
- Como um usuário do IAM altera a própria senha

Definição de 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 ter 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
- Definir uma política de senhas (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 AWS Management Console, 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 Uso de autenticação multifator (MFA) na AWS.

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”.

```
{
    "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.
- Password strength (Nível de segurança da senha): selecione qualquer uma das seguintes opções para definir o nível de segurança de 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 AWS Management Console 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 AWS Management Console 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 AWS Management Console.
- 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.

Definir uma política de senhas (console)

Você pode usar o AWS Management Console para criar, alterar ou excluir uma política de senha.

Para criar uma política de senha personalizada (console)

1. Faça login no AWS Management Console 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 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).

Para alterar uma política de senha personalizada (console)

1. Faça login no AWS Management Console 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. Selecione as opções que você deseja aplicar à política de senha e escolha Save changes (Salvar alterações).
5. Confirme se deseja definir uma política de senha personalizada escolhendo Set custom (Definir personalizada).

Para excluir uma política de senhas personalizada (console)

1. Faça login no AWS Management Console 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 IAM default (Padrão do IAM) para excluir a política de senha personalizada e escolha Save changes (Salvar alterações).
5. Confirme se deseja definir a política de senha padrão do IAM escolhendo Set default (Definir padrão).

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`

Gerenciamento de senhas de usuários do IAM

Os usuários do IAM que usarem o AWS Management Console 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 AWS Management Console 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 AWS Management Console, 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 (usuários, grupos de usuários e funções). Para obter informações sobre o uso de políticas para definir permissões, consulte Alteração de permissões de um usuário do IAM.

É possível conceder aos usuários permissão para alterar as próprias senhas. Para ter 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 (AWS CLI)
- Criar, alterar ou excluir uma senha de usuário do IAM (API da AWS)

Criar, alterar ou excluir uma senha de usuário do IAM (console)

Você pode usar o AWS Management Console para gerenciar senhas de seus usuários do IAM.

Quando os usuários deixam sua organização ou não precisam mais de acesso à AWS, é importante localizar as credenciais que eles estavam utilizando e garantir que elas não estejam mais funcionando. 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 as credenciais para que os usuários antigos não possam mais ter acesso.

Para adicionar uma senha para um usuário do IAM (console)

1. Faça login no AWS Management Console 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 cuja senha você deseja criar.
4. Escolha a guia Credenciais de segurança e, em Login no console, escolha Habilitar acesso ao console.
5. No Habilitar o acesso ao console, no Senha do console, 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.

6. Para exigir que o usuário crie uma nova senha ao fazer login, escolha O usuário deverá criar uma senha no próximo login. Após, escolha Habilitar o acesso ao console.

> **Important**
>
> Se você selecionar a opção O usuário deverá criar uma senha no próximo login, certifique-se de que o usuário tenha permissão para alterar a senha. Para ter mais informações, consulte Permitir que os usuários do IAM alterem as próprias senhas.

7. Para visualizar a senha e compartilhá-la com o usuário, escolha Mostrar na caixa de diálogo Senha do console.

> **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.

Para alterar a senha para um usuário do IAM (console)

1. Faça login no AWS Management Console 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 cuja senha você deseja alterar.
4. Escolha a guia Credenciais de segurança e, em Login no console, escolha Gerenciar acesso ao console.
5. Em Gerenciar acesso ao console, escolha Redefinir senha se ainda não estiver selecionado. Se o acesso ao console estiver desativado, nenhuma senha será necessária.
6. Em Acesso ao console, 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, se houver uma definida.

7. Para exigir que o usuário crie uma nova senha ao fazer login, escolha O usuário deverá criar uma senha no próximo login.

> **Important**
>
> Se você selecionar a opção O usuário deverá criar uma senha no próximo login, certifique-se de que o usuário tenha permissão para alterar a senha. Para ter mais informações, consulte Permitir que os usuários do IAM alterem as próprias senhas.

8. Para revogar as sessões ativas do console do usuário, escolha Revogar sessões ativas do console. Em seguida, escolha Aplicar.

   Quando você revoga sessões ativas do console para um usuário, o IAM anexa uma nova política em linha ao usuário que nega todas as permissões para todas as ações. Ele incluirá uma condição aplicável às restrições somente se a sessão tiver sido criada antes do momento em que você revogar as permissões, bem como em aproximadamente 30 segundos no futuro. Se o usuário criar uma nova sessão depois que você revogar as permissões, a política de negação não se aplicará a esse usuário. Se um usuário revogar suas próprias sessões ativas do console usando esse método, ele será imediatamente desconectado do AWS Management Console.

> **Important**
>
> Para revogar com êxito as sessões ativas do console de um usuário, você deve ter a permissão `PutUserPolicy` do usuário. Isso permite que você anexe a política em linha `AWSRevokeOlderSessions` ao usuário.

9. Para visualizar a senha e compartilhá-la com o usuário, escolha Mostrar na caixa de diálogo Senha do console.

> **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.

Para excluir (desabilitar) a senha de um usuário do IAM (console)

1. Faça login no AWS Management Console 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 cuja senha você deseja excluir.
4. Escolha a guia Credenciais de segurança e, em Login no console, escolha Gerenciar acesso ao console.
5. Em Gerenciar acesso ao console, escolha Desabilitar o acesso ao console se ainda não estiver selecionado. Se o acesso ao console estiver desativado, nenhuma senha será necessária.
6. Para revogar as sessões ativas do console do usuário, escolha Revogar sessões ativas do console. Em seguida, escolha Desativar acesso.

> **Important**
>
> Para revogar com êxito as sessões ativas do console de um usuário, você deve ter a permissão `PutUserPolicy` do usuário. Isso permite que você anexe a política em linha `AWSRevokeOlderSessions` ao usuário.

Quando você revoga sessões ativas do console para um usuário, o IAM incorpora uma nova política em linha ao usuário do IAM que nega todas as permissões para todas as ações. Ele incluirá uma condição aplicável às restrições somente se a sessão tiver sido criada antes do momento em que você revogar as permissões, bem como em aproximadamente 30 segundos no futuro. Se o usuário criar uma nova sessão depois que você revogar as permissões, a política de negação não se aplicará a esse usuário. Se um usuário revogar suas próprias sessões ativas do console usando esse método, ele será imediatamente desconectado do AWS Management Console.

> **Important**
>
> Você pode impedir que um usuário do IAM acesse o AWS Management Console removendo a senha dele. Isso impede que ele faça login no AWS Management Console 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.

Criar, alterar ou excluir uma senha de usuário do IAM (AWS CLI)

Você pode usar a API da AWS CLI para gerenciar as 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 AWS Management Console. 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 ter 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.

   ```
   {
     "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 AWS Management Console. 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 ter 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.

   ```
   {
     "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 AWS Management Console. 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

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, clique em Account settings (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.

Para obter informações sobre a AWS CLI, o Tools for Windows PowerShell e comandos da API que você pode usar para alterar a política de senha da conta (que inclui permitir que todos os usuários alterem suas próprias senhas), consulte Definir uma política de senha (AWS CLI).

Para permitir que usuários selecionados do IAM alterem as próprias senhas

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, clique em Account settings (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 mais detalhes, consulte Criar um usuário do IAM na sua 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 mais detalhes, consulte Gerenciar grupos de usuários do IAM.
6. Atribua a política a seguir ao grupo. Para obter mais informações, consulte Gerenciamento de políticas do IAM.

```
{
  "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 garantir 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
- Gerenciar senhas de usuários na AWS
- Definição de uma política de senhas de contas para usuários do IAM
- Gerenciamento de 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 AWS Management Console 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)

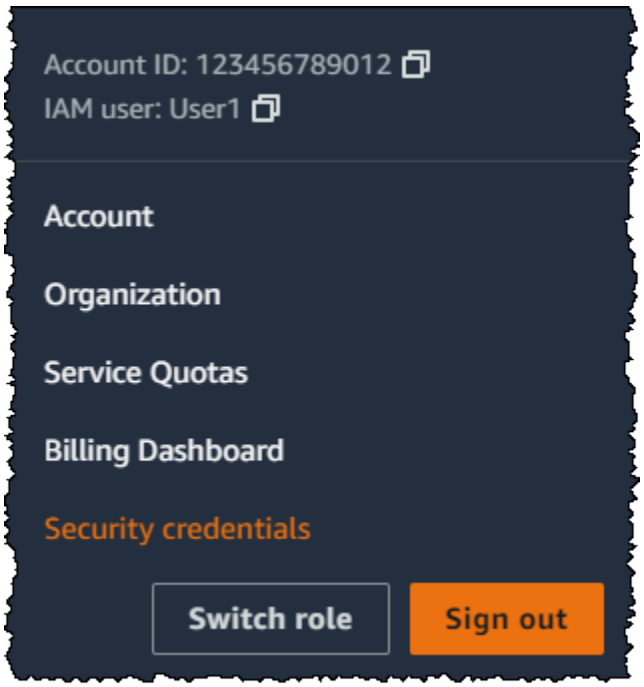
O procedimento a seguir descreve como os usuários do IAM podem usar o AWS Management Console para alterar sua própria senha.

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 ter mais informações, consulte Definição de 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`
- API da AWS: `ChangePassword`

Gerenciamento de chaves de acesso de usuários do IAM

Follow us on Twitter

> **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 ter mais informações, consulte Assinar solicitações de API do 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 só está disponível no momento em que é criada. Se tiver perdido a chave de acesso secreta, você deverá excluir a chave de acesso e criar uma nova. Para obter mais detalhes, consulte Redefinição de senhas perdidas ou esquecidas ou chaves de acesso para a AWS.

É possível ter um máximo de duas chaves de acesso por usuário.

> **Important**
>
> 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.

Os tópicos a seguir detalham as tarefas de gerenciamento associadas a chaves de acesso.

Tópicos

- Permissões necessárias para gerenciar chaves de acesso
- Gerenciar chaves de acesso (console)
- Gerenciar chaves de acesso (AWS CLI)
- Gerenciar chaves de acesso (API da AWS)
- Atualização de chaves de acesso
- Proteção de chaves de acesso
- Fazer auditoria de chaves de acesso

Permissões necessárias para gerenciar chaves de acesso

> **Note**
>
> `iam:TagUser` é uma permissão opcional para adicionar e editar descrições da chave de acesso. Para ter mais informações, consulte Etiquetar usuários do IAM.

Para criar chaves de acesso para seu próprio usuário do IAM, você deve ter as permissões na seguinte política:

```
{
    "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:

```
{
    "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}"
        }
    ]
}
```

Gerenciar chaves de acesso (console)

Você pode usar o AWS Management Console para gerenciar as chaves de acesso de um usuário do IAM.

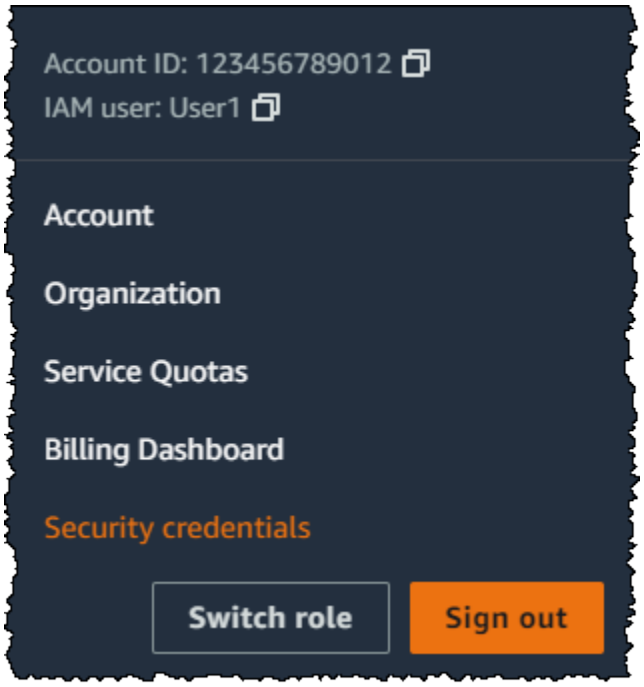
Para criar, modificar ou excluir suas próprias chaves de acesso (console)

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).



Execute um destes procedimentos:

Para criar uma chave de acesso

1. 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.
2. 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).
3. (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).

4. 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).

Para desativar uma chave de acesso

- 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.

Para ativar uma chave de acesso

- Na seção Access keys (Chaves de acesso), localize a chave que deseja ativar, escolha Actions (Ações) e escolha Activate (Ativar).

Para excluir uma chave de acesso quando não precisar mais dela

- 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.

Para criar, modificar ou excluir as chaves de acesso de outro usuário do IAM (console)

1. Faça login no AWS Management Console 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 cujas chaves de acesso você deseja gerenciar e a guia Security credentials (Credenciais de segurança).
4. Na seção Access Keys (Chaves de acesso), execute uma das seguintes ações:
   - Para criar uma chave de acesso, selecione Criar chave de acesso. Se esse botão estiver desativado, você deverá excluir uma das chaves de acesso existentes antes de criar outra. 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. Se você determinar que seu caso de uso ainda requer uma chave de acesso, escolha Other (Outro) e escolha Next (Avançar). Na página Retrieve access keys (Recuperar chaves de acesso), escolha Show (Exibir) para revelar o valor da chave de acesso secreta do usuário. 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). 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.

- Para desativar uma chave de acesso ativa, escolha Actions (Ações) e escolha Deactivate (Desativar).
- Para ativar uma chave de acesso inativa, escolha Actions (Ações) e escolha Activate (Ativar).
- Para excluir sua chave de acesso, escolha Actions (Ações) e escolha Delete (Excluir). Siga as instruções na caixa de diálogo para primeiro Deactivate (Desativar) e depois confirme a exclusão. A AWS recomenda que, antes de fazer isso, você primeiro desative a chave e teste se não está mais em uso. Ao usar o AWS Management Console, você deve desativar a chave antes de excluí-la.

Para listar as chaves de acesso de um usuário do IAM (console)

1. Faça login no AWS Management Console 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 desejado e, em seguida, selecione a guia Credenciais de segurança. Na seção Access keys (Chaves de acesso), você deverá ver as chaves de acesso do usuário e o status de cada chave de acesso.

> **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.

Para listar os IDs das chave de acesso de vários usuários do IAM (console)

1. Faça login no AWS Management Console 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 Access key ID 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 ID de chave de acesso.

   c. Escolha Fechar para retornar à lista de usuários.
4. A coluna Access key ID (ID de chave de acesso) exibe cada ID de chave de acesso, seguido por seu estado; por exemplo, 234782070278420732307623740 23 (Ativo) ou 220937402396702370248434203 27 (Inativo).

   Você pode usar essas informações para visualizar e copiar as chaves de acesso para os usuários com uma ou duas chaves de acesso. A coluna exibe Nenhum para usuários sem chaves de acesso.

   > **Note**
   >
   > Somente o ID de chave de acesso do usuário e o status são visíveis. A chave de acesso secreta só pode ser recuperada quando a chave é criada.

Para encontrar qual usuário do IAM tem uma determinada chave de acesso (console)

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Usuários.
3. Na caixa de pesquisa, digite ou cole o ID de chave de acesso do usuário que você deseja encontrar.
4. Se necessário, adicione a coluna Access key ID 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 ID de chave de acesso.

c. Selecione Fechar para retornar à lista de usuários e confirmar que o usuário filtrado tem a chave de acesso.

Gerenciar chaves de acesso (AWS CLI)

Para gerenciar as chaves de acesso de um usuário do IAM da AWS CLI, execute os comandos a seguir.

- Para criar uma chave de acesso: `aws iam create-access-key`
- Para ativar ou desativar uma chave de acesso: `aws iam update-access-key`
- Para listar as chaves de acesso de um usuário: `aws iam list-access-keys`
- Para determinar quando uma chave de acesso foi usada recentemente: `aws iam get-access-key-last-used`
- Para excluir uma chave de acesso: `aws iam delete-access-key`

Gerenciar chaves de acesso (API da AWS)

Para gerenciar as chaves de acesso de um usuário do IAM da API da AWS, chame as operações a seguir.

- Para criar uma chave de acesso: `CreateAccessKey`
- Para ativar ou desativar uma chave de acesso: `UpdateAccessKey`
- Para listar as chaves de acesso de um usuário: `ListAccessKeys`
- Para determinar quando uma chave de acesso foi usada recentemente: `GetAccessKeyLastUsed`
- Para excluir uma chave de acesso: `DeleteAccessKey`

Atualização de 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 ter mais informações, consulte Definição de uma política de senhas de contas para usuários do IAM.

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 AWS Management Console.

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 AWS Management Console 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 garantir que todos os aplicativos e as ferramentas sejam atualizadas, você poderá excluir a primeira chave de acesso:

   a. Faça login no AWS Management Console 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 AWS Management Console 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 garantir que todos os aplicativos e as ferramentas sejam atualizadas, 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 garantir que todos os aplicativos e as ferramentas sejam atualizadas, você poderá excluir a primeira chave de acesso chamando esta operação:

   - `DeleteAccessKey`

Proteção de 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 AWS IAM Identity Center 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 Atualização de 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 ter 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 ter 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 ter mais informações, consulte Habilitar o acesso do agente de identidades personalizado ao console da AWS.

> **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 for Java e Arquivos de configuração e credenciais no Guia do usuário da AWS Command Line Interface.

  Para armazenar credenciais para o AWS SDK for .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 for .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 Atualização de 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. 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. Para ter mais informações, consulte Configuração de acesso à API protegido por 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 for Java
- Utilização da SDK Store no Guia do desenvolvedor do AWS SDK for .NET
- Fornecimento de credenciais para o SDK no Guia do desenvolvedor do AWS SDK for 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 for .NET
- Configuração de perfis do IAM para o Amazon EC2 no AWS SDK for Java 2.x

Fazer auditoria de chaves de acesso

Você pode revisar as chaves de acesso da AWS em seu código para determinar se as chaves são de uma conta que você possui. Você pode transmitir um ID da chave de acesso usando o comando `aws sts get-access-key-info` da AWS CLI ou a operação de 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 ter 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.

Redefinição de senhas perdidas ou esquecidas ou chaves de acesso para a AWS

> **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 AWS Support. 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.

Na página principal de login, você deve inserir seu endereço de e-mail para fazer login como usuário raiz, ou insira o ID da conta para fazer login como usuário do IAM. Você pode fornecer sua senha apenas na página de login que corresponde ao seu tipo de usuário. Para obter mais informações, consulte Fazer login no AWS Management Console.

Se você estiver na página de login correta e perder ou esquecer suas senhas ou chaves de acesso, não poderá recuperá-las do IAM. Em vez disso, você pode redefini-las usando os seguintes métodos:

- Senha do Usuário raiz da conta da AWS: se você esquecer a senha de usuário raiz, poderá redefini-la no AWS Management Console. Para obter mais detalhes, consulte the section called "Redefinição de uma senha de usuário raiz perdida ou esquecida" mais adiante neste tópico.
- Chaves de acesso da Conta da AWS: se você esquecer as chaves de acesso à sua conta, poderá criar novas sem desabilitar as existentes. Se você não estiver usando as chaves existentes, poderá exclui-las. Para obter mais detalhes, consulte Criar chaves de acesso para o usuário raiz e Excluir chaves de acesso do usuário raiz.
- Senha de usuário do IAM: se você for um usuário do IAM e esquecer sua senha, deverá pedir ao administrador para redefinir sua senha. Para saber como um administrador pode gerenciar sua senha, consulte Gerenciamento de senhas de usuários do IAM.
- Chaves de acesso de usuário do IAM: se você for um usuário do IAM e esquecer suas chaves de acesso, precisará de novas chaves de acesso. Se você tem permissões para criar as próprias chaves de acesso, pode encontrar instruções para criar uma nova chave em Gerenciar chaves de acesso (console). Se você não tem as permissões necessárias, deve solicitar que o administrador crie novas chaves de acesso. Se você ainda estiver usando as chaves antigas, peça ao administrador que não as exclua. Para saber como um administrador pode gerenciar as chaves de acesso, consulte Gerenciamento de chaves de acesso de usuários do IAM.

Uso de autenticação multifator (MFA) na AWS

Follow us on Twitter

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 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. Os usuários do IAM na conta são identidades distintas com suas próprias credenciais, e cada identidade tem sua própria configuração de MFA. 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. Para obter mais informações sobre os tipos de MFA com suporte, consulte O que é MFA?. Com vários dispositivos com MFA, é necessário apenas um dispositivo com MFA para acessar o AWS Management Console ou criar uma sessão pela AWS CLI como esse usuário.

> **Note**
>
> Recomendamos exigir que seus usuários humanos usem credenciais temporárias ao acessar a AWS. Você já pensou em usar o AWS IAM Identity Center? 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 AWS IAM Identity Center.

O que é MFA?

A MFA aumenta a segurança porque, além das credenciais de login usuais, exige que os usuários forneçam autenticação exclusiva em um mecanismo de MFA compatível com a AWS quando acessam sites ou serviços da AWS. A AWS é compatível com os tipos de MFA a seguir.

Segurança FIDO

As chaves de segurança de hardware certificadas pela FIDO são oferecidas por fornecedores terceirizados.

A FIDO Alliance conta com uma lista de todos os produtos certificados pela FIDO compatíveis com as especificações da FIDO. Os padrões de autenticação FIDO são baseados em criptografia de chave pública, permitindo uma autenticação forte e resistente a phishing que é mais segura do que senhas. Chaves de segurança FIDO oferecem suporte a várias contas raiz e usuários do IAM usando uma única chave de segurança. Para obter mais informações sobre como habilitar as chaves de segurança FIDO, consulte Habilitar uma chave de segurança FIDO (console).

Dispositivos virtuais de MFA

Uma aplicação de autenticador virtual que é executada 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 em uma segunda página da web durante o login. Cada dispositivo MFA virtual atribuído a um usuário deve ser exclusivo. O usuário não pode

digitar um código no dispositivo de MFA de outro usuário para se autenticar. Como eles podem ser executados em dispositivos móveis não protegidos, a MFA virtual talvez não forneça o mesmo nível de segurança de chaves de segurança FIDO.

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 alguns aplicativos compatíveis que podem ser usados como dispositivos MFA virtuais, consulte a página Autenticação multifator. Para obter instruções sobre como configurar um dispositivo MFA virtual com a AWS, consulte Habilitar um dispositivo de autenticação multifator (MFA) virtual (console).

Token físico de TOTP

Um dispositivo físico que 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 MFA de hardware compatíveis, consulte Autenticação multifator. Para obter instruções sobre como configurar um token de hardware TOTP com a AWS, consulte Habilitar um token de hardware TOTP (console).

Recomendamos que você use chaves de segurança FIDO como alternativa aos dispositivos físicos de TOTP. As chaves de segurança FIDO oferecem a vantagem de não usar bateria, ter resistência a phishing e comportar vários usuários de IAM ou raízes em um único dispositivo para maior segurança.

> **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 usam MFA baseada em texto de SMS mudem para um dos seguintes métodos alternativos: chave de segurança FIDO, dispositivo de MFA virtual (baseado em software) 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.

Tópicos

Habilitar dispositivos com MFA para usuários na AWS

As etapas para configurar a MFA dependem do tipo de dispositivo MFA que você está usando.

Tópicos

Etapas gerais para a habilitação de dispositivos com MFA

O procedimento de visão geral a seguir descreve como configurar e usar a MFA e fornece links para informações relacionadas.

> **Observação**
>
> Você também pode assistir a este vídeo em inglês, Como configurar a autenticação multifator (MFA) da AWS e alertas do AWS Budget, para obter mais informações.

1. Obtenha um dispositivo MFA, como um dos seguintes. Você pode habilitar até oito dispositivos com MFA por Usuário raiz da conta da AWS ou usuário do IAM de qualquer combinação dos tipos abaixo.

- Um dispositivo MFA virtual, que é um app de software compatível com o RFC 6238, um algoritmo TOTP (senha única baseada em tempo) com base em padrões. Você pode instalar o aplicativo em um telefone ou outro dispositivo. Para obter uma lista de alguns aplicativos compatíveis que podem ser usados como dispositivos MFA virtuais, consulte a página Autenticação multifator.
- Uma chave de segurança FIDO com uma configuração com suporte da AWS. A FIDO Alliance conta com uma lista de todos os produtos certificados pela FIDO compatíveis com as especificações da FIDO.
- Um dispositivo físico de MFA de um provedor terceirizado, como um dispositivo de token. Esses tokens são usados exclusivamente com Contas da AWS. Para ter mais informações, consulte Habilitar um token de hardware TOTP (console). Você só pode usar tokens que compartilhem as sementes de token com a AWS de modo seguro. 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.

2. Habilite o dispositivo MFA.
   - Tokens virtuais ou de hardware TOTP: você pode usar os comandos da AWS CLI ou operações de API da AWS para habilitar um dispositivo de 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 AWS Management Console para habilitar um dispositivo com MFA para o usuário raiz.
   - Chaves de segurança FIDO: usuários raiz e usuários do IAM com chaves de segurança FIDO podem ser habilitados somente pelo AWS Management Console, não pela AWS CLI ou pela AWS.

   Para obter informações sobre como habilitar cada tipo de dispositivo MFA, consulte as seguintes páginas:
   - Dispositivo MFA virtual: Habilitar um dispositivo de autenticação multifator (MFA) virtual (console)
   - Chave de segurança FIDO: Habilitar uma chave de segurança FIDO (console)
   - Token de hardware TOTP: Habilitar um token de hardware TOTP (console)
3. Habilitar vários dispositivos de MFA (recomendado)
   - 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 AWS Management Console 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.

4. Use o dispositivo MFA ao fazer login ou acessar recursos da AWS. Observe o seguinte:
   - Chaves de segurança FIDO: para acessar um site da AWS, insira suas credenciais e toque na chave de segurança FIDO quando solicitado.
   - Dispositivos de MFA virtuais e tokens de hardware TOTP: para acessar um site da AWS, você precisa de um código de MFA do dispositivo, além do nome de usuário e senha.

     Para acessar as operações de API protegidas por MFA, você precisa do seguinte:
     - Um código MFA
     - O identificador do dispositivo de MFA (o número de série de um dispositivo físico ou o ARN de um dispositivo virtual definido na AWS)
     - O ID de chave de acesso e a chave de acesso secreta usuais

> **Observações**
>
> - Você não pode transmitir as informações de MFA de uma chave de segurança FIDO para operações de API do AWS STS para solicitar credenciais temporárias.

- 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 dispositivo raiz ou MFA do IAM.

Para ter mais informações, consulte Uso de dispositivos com MFA com sua página de login do IAM.

Habilitar um dispositivo de autenticação multifator (MFA) virtual (console)

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 eles podem ser executados em dispositivos móveis não protegidos, a MFA virtual talvez não forneça o mesmo nível de segurança de chaves de segurança FIDO. 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.

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 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 AWS Management Console ou criar uma sessão pela AWS CLI como esse usuário. Recomendamos que você registre vários dispositivos de MFA. 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.

Para obter uma lista das aplicações de MFA que você pode usar, consulte Autenticação multifator. A AWS exige uma aplicação de MFA que gere uma senha para uso único (OTP) com seis dígitos.

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 AWS Management Console 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 Habilitar e gerenciar dispositivos MFA virtuais (AWS CLI ou API da AWS). Para obter mais informações sobre recursos de marcação do IAM, consulte Recursos de etiquetas do IAM.

> **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 AWS Management Console 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. 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 , 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 (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 AWS Management Console, consulte Uso de dispositivos com MFA com sua página de login do IAM.

Substituir um dispositivo de MFA virtual

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. 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 associado no momento a outro usuário do IAM, consulte Desativar dispositivos 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 com MFA virtual para o Usuário raiz da conta da AWS (console).

Habilitar uma chave de segurança FIDO (console)

As chaves de segurança FIDO são um tipo de dispositivo de autenticação multifator (MFA) que você pode usar para proteger seus recursos da AWS. Você conecta a chave de segurança FIDO a uma porta USB do computador e a habilita usando as instruções a seguir. Depois de habilitá-la, você tocará nela quando solicitado 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. Para obter as especificações e informações de compra, consulte Autenticação multifator.

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 pela 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 toca na chave de segurança FIDO, que responde ao desafia de autenticação emitido pela AWS. Para saber mais sobre o padrão FIDO2, consulte Projeto FIDO2.

Você pode registrar até oito dispositivos de 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 AWS Management Console 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 segurança FIDO para seu próprio usuário do IAM (console)

- Habilitar uma chave de segurança FIDO para outro usuário do IAM (console)
- Substituir uma chave de segurança FIDO
- Configurações compatíveis com o uso de chaves de segurança FIDO

Permissões obrigatórias

Para gerenciar uma chave de segurança 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.

```
{
    "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 segurança FIDO para seu próprio usuário do IAM (console)

Você só pode habilitar uma chave de segurança FIDO para seu próprio usuário do IAM no AWS Management Console, não na AWS CLI ou na API da AWS.

> **Note**
>
> Para poder habilitar uma chave de segurança FIDO, você deve ter acesso físico ao dispositivo.

> **Note**
>
> Você não deve escolher nenhuma das opções disponíveis no pop-up do Goggle Chrome que solicite Verify your identity with amazon.com (Confirmar sua identidade na amazon.com). Você só precisa tocar na chave de segurança.

Para habilitar uma chave de segurança FIDO para seu próprio usuário do IAM (console)

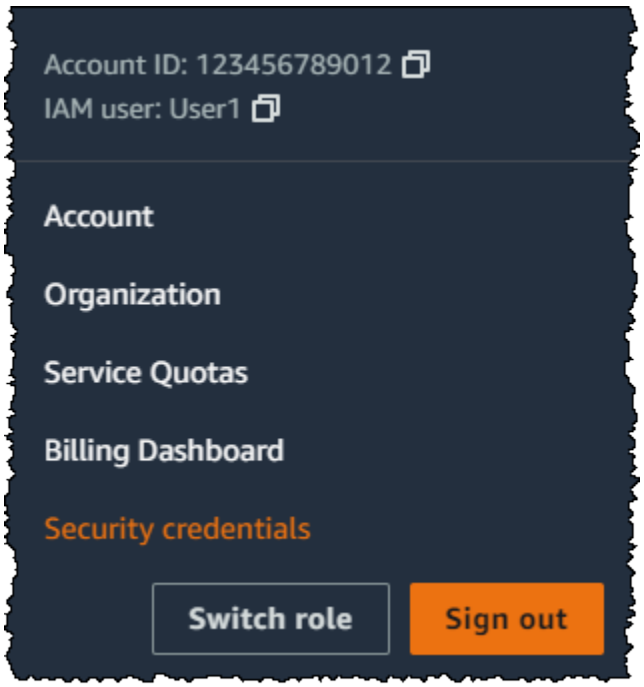
1. Use o ID ou o alias da conta da AWS, o nome de usuário do IAM e a senha para fazer login no console do IAM.

   > **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 Security Key (Chave de segurança) e escolha Next (Avançar).
5. Insira a chave de segurança FIDO na porta USB do computador.

6. Toque na chave de segurança FIDO.

A chave de segurança FIDO está pronta para ser usada com a AWS. Para obter informações sobre como usar a MFA com o AWS Management Console, consulte Uso de dispositivos com MFA com sua página de login do IAM.

Habilitar uma chave de segurança FIDO para outro usuário do IAM (console)

Você só pode habilitar uma chave de segurança FIDO para outro usuário do IAM no AWS Management Console, não na AWS CLI ou na API da AWS.

Para habilitar uma chave de segurança FIDO para outro usuário do IAM (console)

1. Faça login no AWS Management Console 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 Security Key (Chave de segurança) e escolha Next (Avançar).
6. Insira a chave de segurança FIDO na porta USB do computador.

7. Toque na chave de segurança FIDO.

A chave de segurança FIDO está pronta para ser usada com a AWS. Para obter informações sobre como usar a MFA com o AWS Management Console, consulte Uso de dispositivos com MFA com sua página de login do IAM.

Substituir uma chave de segurança FIDO

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 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 dispositivos MFA.
- Para adicionar uma nova chave de segurança FIDO para um usuário do IAM, consulte Habilitar uma chave de segurança FIDO para seu próprio usuário do IAM (console).

Se você não tiver acesso a uma nova chave de segurança FIDO, poderá ativar um novo dispositivo de MFA virtual ou um token de hardware TOTP. Consulte um dos tópicos a seguir para obter instruções:

- Habilitar um dispositivo de autenticação multifator (MFA) virtual (console)
- Habilitar um token de hardware TOTP (console)

Configurações compatíveis com o uso de chaves de segurança FIDO

Você pode usar as chaves de segurança FIDO2 como um método 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 recursos 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.

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. Não há compatibilidade com os autenticadores de plataforma, como TouchID, FaceID ou Windows Hello.

> **Note**
>
> A AWS precisa ter acesso à porta USB física no computador para verificar o dispositivo U2F. As chaves de segurança FIDO2 não funcionam 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 FIDO2:

	macOS 10.15+	Windows 10	Linux	iOS 14.5+	Android 7+
Chrome	Sim	Sim	Sim	Sim	Não
Safari	Sim	Não	Não	Sim	Não
Borda	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 Solução de problemas de chaves de segurança FIDO.

Para obter mais informações sobre a compatibilidade do navegador com um dispositivo FIDO2 certificado, 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 FIDO. 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 FIDO mudar, isso não será refletido automaticamente nas etiquetas do dispositivo. Para atualizar as informações de certificação de um dispositivo, registre-o novamente para buscar as informações de certificação atualizadas.

A AWS 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 especificar o registro de autenticadores específicos em suas políticas do IAM, com base no tipo e nível de certificação de sua preferência. 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

```
{
    "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

```
{
    "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

```
{
    "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

```
{
    "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:FIPS-140-2-certification": "L2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:EnableMFADevice",
            "Resource": "*",
            "Condition": {
```

```
                "Null": {
                    "iam:RegisterSecurityKey": "true"
                }
            }
        }
    ]
}
```

AWS CLI e API da AWS

A AWS só permite o uso de chaves de segurança FIDO2 no AWS Management Console. Não é possível usar de chaves de segurança FIDO2 para MFA na AWS CLI e na API da AWS, nem para acessar operações de API protegidas por MFA.

Recursos adicionais do

- Para obter mais informações sobre o uso de chaves de segurança FIDO2 na AWS, consulte Habilitar uma chave de segurança FIDO (console).
- Para obter ajuda na solução de problemas de chaves de segurança FIDO2 na AWS, consulte Solução de problemas de chaves de segurança FIDO.
- Para obter informações gerais do setor sobre compatibilidade com FIDO2, consulte FIDO2 Project.

Habilitar um token de hardware TOTP (console)

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 AWS Management Console, 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 da Conta da AWS (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 AWS Management Console 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:

```
{
    "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 AWS Management Console.

> **Note**
>
> Para poder habilitar um token de hardware TOTP, é necessário ter acesso físico ao dispositivo.

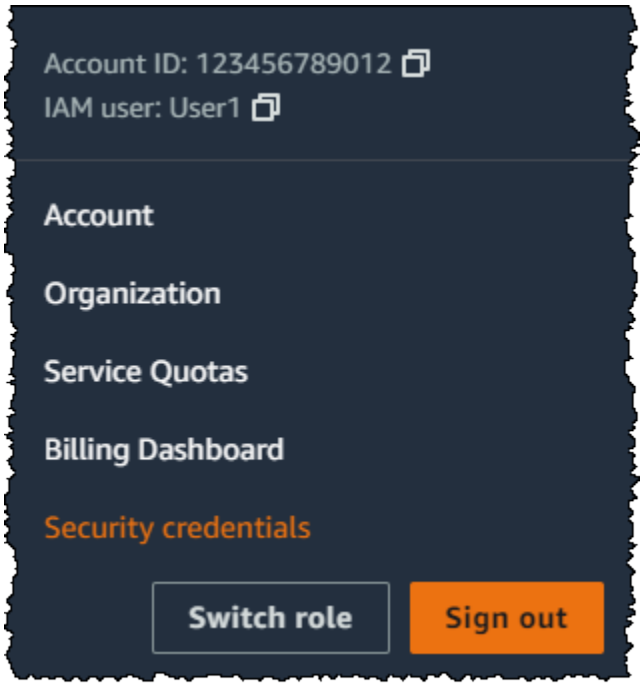
Para habilitar um token de hardware TOTP para seu próprio usuário do IAM (console)

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 AWS Management Console, consulte Uso de dispositivos com MFA com sua página de login do IAM.

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 AWS Management Console.

Para habilitar um token de hardware TOTP para outro usuário do IAM (console)

1. Faça login no AWS Management Console 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 AWS Management Console, consulte Uso de dispositivos com MFA com sua página de login do IAM.

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 dispositivos 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 da Conta da AWS (console) anterior neste tópico.

Habilitar e gerenciar dispositivos MFA virtuais (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 AWS Management Console para habilitar um dispositivo com MFA para o usuário raiz.

Quando você habilita um dispositivo MFA do AWS Management Console, 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`
- API da AWS: `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 (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`
- API da AWS: `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`
- API da AWS: `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`
- API da AWS: `ListVirtualMFADevices`

Marcar um dispositivo MFA virtual

Use estes comandos para marcar um dispositivo MFA virtual.

- AWS CLI: `aws iam tag-mfa-device`
- API da AWS: `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`
- API da AWS: `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`
- API da AWS: `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`

- API da AWS: `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`
- API da AWS: `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 O que acontece se um dispositivo com MFA for perdido ou parar de funcionar?.

Verificação do 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 AWS Management Console 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 com MFA virtual para o Usuário raiz da conta da AWS (console)
- Habilitar uma chave de segurança FIDO para o usuário raiz da Conta da AWS (console)

- Habilitar um token de hardware TOTP para o usuário raiz da Conta da AWS (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 (Usuários).
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:

- Habilitar um dispositivo de autenticação multifator (MFA) virtual (console)
- Habilitar uma chave de segurança FIDO (console)
- Habilitar um token de hardware TOTP (console)

Sincronizar novamente dispositivos com 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 dispositivos MFA e Habilitar dispositivos com MFA para usuários na AWS.

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.

```
{
    "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.

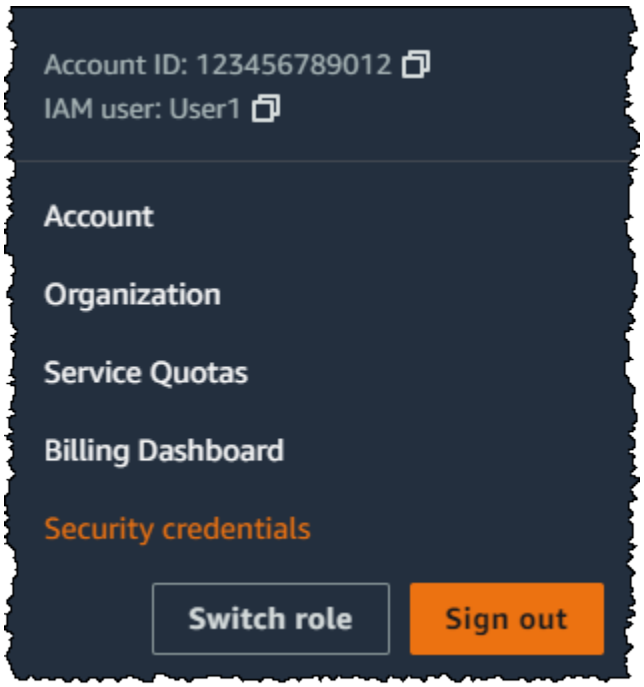
Para sincronizar novamente um dispositivo com MFA virtual ou de hardware para seu próprio usuário do IAM (console)

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 AWS Management Console 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**
>
> Você pode ver um texto diferente, como Fazer login usando MFA e Solucionar problemas do dispositivo de autenticação. No entanto, os mesmos recursos são fornecidos.

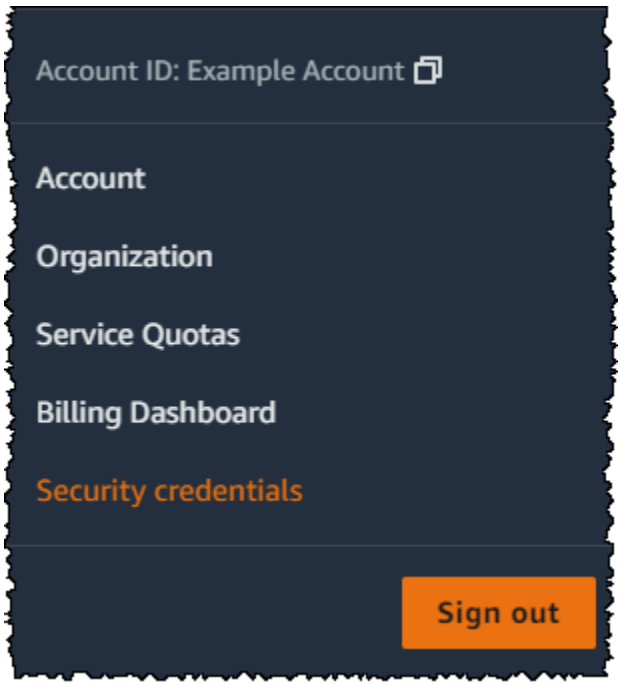
2. Na seção Re-Sync With Our Servers (Sincronizar novamente com nossos servidores), digite os próximos dois códigos gerados sequencialmente no dispositivo em MFA code 1 (Código MFA 1) e MFA code 2 (Código MFA 2). Em seguida, escolha Sincronizar novamente dispositivo de autenticação.
3. Se necessário, 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 AWS Management Console 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 dispositivos 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 the section called “Habilitação de dispositivos com MFA”.

> **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 exclusão de usuários, consulte Gerenciar usuários 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 AWS Management Console 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.

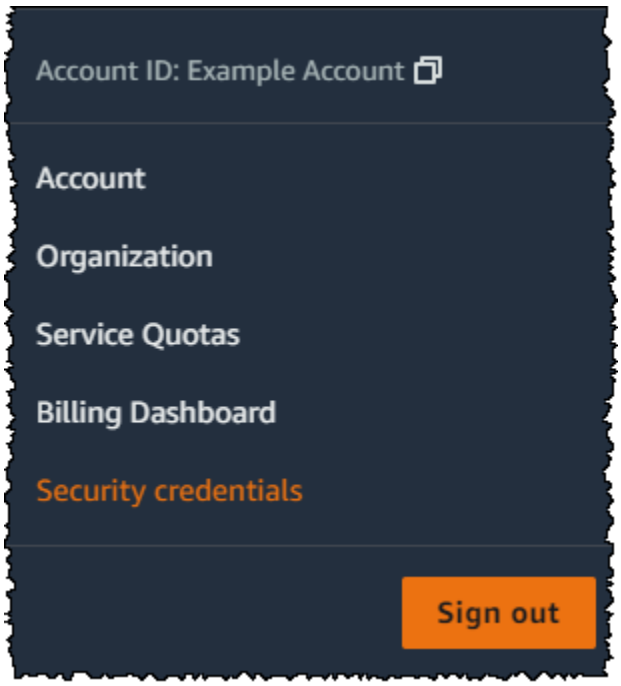
   O dispositivo é removido da AWS. Ele não poderá ser usado para fazer login nem autenticar as solicitações até ser reativado e associado a um usuário da AWS ou ao Usuário raiz da conta da AWS.

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 AWS Management Console 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`.

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`

O que acontece se um dispositivo com MFA for perdido ou parar de funcionar?

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 dispositivos com MFA virtuais e de hardware. As chaves de segurança FIDO não perdem a sincronia.

Se o dispositivo com autenticação multifator (MFA) do Usuário raiz da conta da AWS for perdido, danificado ou não funcionar, você poderá recuperar o acesso à conta. Os usuários do IAM devem entrar em contato com um administrador para desativar o dispositivo.

> **Important**
>
> Recomendamos habilitar vários dispositivos de MFA para que seus usuários do IAM garantam acesso contínuo à conta, caso um dispositivo de MFA seja perdido ou fique inacessível. Você pode registrar até oito dispositivos de MFA de qualquer combinação dos tipos de MFA atualmente compatíveis com seu usuário raiz da Conta da AWS e usuários do IAM.

Recuperar um dispositivo com MFA de usuário raiz

Se o dispositivo com autenticação multifator (MFA) do Usuário raiz da conta da AWS for perdido, danificado ou não estiver funcionando, você poderá fazer login usando outro dispositivo com MFA registrado no mesmo Usuário raiz da conta da AWS. Se o usuário raiz tiver apenas um dispositivo de MFA habilitado, será possível usar métodos alternativos de autenticação. Isso significa que, se você não puder fazer login com o seu dispositivo de MFA, poderá fazer login comprovando sua identidade usando o e-mail e o telefone registrados na sua conta.

Antes de fazer login como usuário raiz usando fatores de autenticação alternativos, verifique se tem acesso ao e-mail e ao telefone de contato associados à conta. Se você precisar atualizar o telefone de contato principal, poderá fazer 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 Account Management.

Para fazer login usando fatores alternativos de autenticação como um Usuário raiz da conta da AWS

1. Faça login no AWS Management Console como o proprietário da conta ao escolher a opção Root user (Usuário raiz) e inserir o endereço de e-mail da Conta da AWS. Na próxima página, insira sua 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 AWS Security Credentials (Credenciais de segurança da AWS) e exclua a entidade do dispositivo com MFA virtual antiga antes de criar uma nova.
    - Para obter uma chave de segurança FIDO, acesse a página Credenciais de segurança da AWS e desative a chave 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 com MFA de hardware, acesse a página Credenciais de segurança da AWS e exclua a entidade de dispositivo com MFA de hardware antiga antes de criar um nova.

> **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 de MFA virtual ou um token de hardware TOTP até receber uma nova chave de segurança FIDO.

> **Important**
>
> Se seu dispositivo de MFA estiver ausente ou tiver sido roubado, depois de iniciar sessão usando fatores alternativos de autenticação e estabelecer seu dispositivo de MFA substituto, altere a senha de usuário raiz, para proteção caso um invasor tenha roubado o dispositivo de autenticação e também possa ter sua senha atual. Para obter mais informações, consulte Alterar a senha do Usuário raiz da conta da AWS no Guia de referência do AWS Account Management.

Recuperar um dispositivo com MFA de usuário do IAM

Se você for um usuário do IAM e seu dispositivo for perdido ou parar de funcionar, não será possível recuperá-lo sozinho. 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 dispositivos 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 Habilitar um dispositivo de autenticação multifator (MFA) virtual (console).
   - 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 Habilitar uma chave de segurança FIDO (console).
   - 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 Habilitar um token de hardware TOTP (console).

   > **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 de 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 Gerenciamento de senhas de usuários do IAM.

Configuração de acesso à API protegido por MFA

Com as políticas do IAM, você pode especificar quais operações de API um usuário tem permissão para chamar. Em alguns casos, você pode desejar obter segurança adicional para exigir que os usuários sejam autenticados com a Multi-Factor Authentication (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. 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. Esse processo é descrito em Habilitar dispositivos com MFA para usuários na AWS.
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 suporte os parâmetros de MFA AssumeRole ou GetSessionToken, dependendo do cenário de proteção de MFA, conforme

explicado adiante. 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.

```
{
  "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 Chaves de contexto de condição globais da AWS, 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.

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.

```
{
  "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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TableActions",
            "Effect": "Allow",
            "Action": "dynamodb:*",
            "Resource": "arn:aws:dynamodb:*:ACCOUNT-A-ID: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.

   ```
   {
     "Version": "2012-10-17",
     "Statement": [{
       "Effect": "Allow",
       "Action": ["sts:AssumeRole"],
       "Resource": ["arn:aws:iam::ACCOUNT-A-ID: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.

```
{
  "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.

```
{
  "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 códigos a seguir mostram como usar `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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetSessionToken na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetSessionToken na Referência de Cmdlet do AWS Tools for PowerShell.

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 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)
```

- Para obter detalhes da API, consulte GetSessionToken na Referência da API do 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 códigos a seguir mostram como usar `AssumeRole`.

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja 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>

        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);

            // 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}");
        }
    }
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API do AWS SDK for .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
        ;;
      \?)
        ech o"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
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência de comandos da AWS CLI.

C++

SDK for 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();
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API do AWS SDK for 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte 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, rn: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);
        }
    }
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API do 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);
  }
};
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API do AWS SDK for 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);
    }
  });
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API do AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte AssumeRole na Referência de Cmdlet do AWS Tools for PowerShell.

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 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)
```

- Para obter detalhes da API, consulte AssumeRole 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.

```
 # 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
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API do AWS SDK for Ruby.

Rust

SDK for 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),
    }
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência do AWS SDK para API Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWSCode Examples Repository.

```
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 {
        throw error
    }
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência do AWS SDK para API Swift.

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 AWS Management Console 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 AWS Management Console 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 AWS Management Console 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 AWS Management Console 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 AWS Management Console 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 AWS Management Console 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).

Obter relatórios de credenciais da 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 AWS Management Console, dos SDKs da AWS e das Ferramentas da linha de comando ou da API do IAM.

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

- 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:

user

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, ele será `FALSE`. O valor para o Usuário raiz da conta da AWS é sempre `not_supported`.

password_last_used

A data e a hora em que a senha do Usuário raiz da conta da AWS ou do usuário 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 AWS Management Console, 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). O valor para a Conta da AWS (raiz) é sempre `not_supported`.

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`.

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).

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.

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.

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.

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`.

> **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 Atualização de 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).

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.

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. 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. 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).

Obter relatórios de credenciais (console)

Você pode usar o AWS Management Console 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 AWS Management Console 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`

Uso do IAM com 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 sua 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 sua 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 Provedores de identidade e federação e Fornecer 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.

Uso do 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 AWS Management Console 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 Connecting programmatically to Amazon Keyspaces (Conexão programática com o Amazon Keyspaces) no Amazon Keyspaces (para Apache Cassandra) Developer Guide (Guia do desenvolvedor do Amazon Keyspaces [for Apache Cassandra]).

> **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 AWS Management Console 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 AWS Management Console 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

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 AWS Certificate Manager (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 AWS Certificate Manager na Referência geral da AWS.

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 AWS Certificate Manager. 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 AWS Certificate Manager. Para obter mais informações sobre a importação de certificados de terceiros para o ACM, consulte Importação de certificados no Manual do usuário do AWS Certificate Manager.

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.

Índice

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 Solução de problemas.
- 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 Solução de problemas.

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.

Recuperar um certificado de servidor (API da AWS)

Para usar a API do IAM para recuperar um certificado, envie uma solicitação GetServerCertificate. O exemplo a seguir mostra como fazer isso com a AWS CLI. Substitua *ExampleCertificate* pelo nome do certificado a ser recuperado.

```
aws iam get-server-certificate --server-certificate-name ExampleCertificate
```

Quando o comando anterior for executado com êxito, ele retornará 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.

Para usar o AWS Tools for Windows PowerShell para recuperar um certificado, use Get-IAMServerCertificate.

Listar certificados de servidor (API da AWS)

Para usar a API do IAM para listar seus certificados de servidor carregados, envie uma solicitação ListServerCertificates. O exemplo a seguir mostra como fazer isso com a AWS CLI.

```
aws iam list-server-certificates
```

Quando o comando anterior for executado com êxito, ele retornará uma lista com metadados sobre cada certificado.

Para usar o AWS Tools for Windows PowerShell para listar seus certificados de servidor carregados, use Get-IAMServerCertificates.

Marcar e desmarcar certificados de servidor (API da AWS)

Você pode associar etiquetas aos seus recursos do IAM para organizar e controlar o acesso a eles. Para usar a API do IAM para etiquetar um certificado de servidor existente, envie uma solicitação TagServerCertificate. O exemplo a seguir mostra como fazer isso com a AWS CLI.

```
aws iam tag-server-certificate --server-certificate-name ExampleCertificate
                                --tags '{"Key": "ExampleKey", "Value":
 "ExampleValue"}'
```

Quando o comando anterior for bem-sucedido, nenhuma saída será retornada.

Para usar a API do IAM para desetiquetar um certificado de servidor, envie uma solicitação UntagServerCertificate. O exemplo a seguir mostra como fazer isso com a AWS CLI.

```
aws iam untag-server-certificate --server-certificate-name ExampleCertificate
                                --tag-keys ExampleKeyName
```

Quando o comando anterior for bem-sucedido, nenhuma saída será retornada.

Renomear um certificado de servidor ou atualizar seu caminho (API da AWS)

Para usar a API do IAM para renomear um certificado de servidor ou atualizar seu caminho, envie uma solicitação UpdateServerCertificate. 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/
```

Quando o comando anterior for executado com êxito, ele não retornará nenhuma saída.

Para usar o AWS Tools for Windows PowerShell para renomear um certificado de servidor ou atualizar seu caminho, use Update-IAMServerCertificate.

Excluir um certificado de servidor (API da AWS)

Para usar a API do IAM para excluir um certificado de servidor, envie uma solicitação DeleteServerCertificate. O exemplo a seguir mostra como fazer isso com a AWS CLI.

Para usar o seguinte comando de exemplo, substitua *ExampleCertificate* pelo nome do certificado a ser excluído.

```
aws iam delete-server-certificate --server-certificate-name ExampleCertificate
```

Quando o comando anterior for executado com êxito, ele não retornará nenhuma saída.

Para usar o AWS Tools for Windows PowerShell para excluir um certificado de servidor, use Remove-IAMServerCertificate.

Solução de problemas

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 uma pessoa mudar de cargo na organização, em vez de editar as permissões de usuário, você poderá removê-las dos grupos antigos do usuário e adicioná-las aos novos grupos do usuário apropriados.

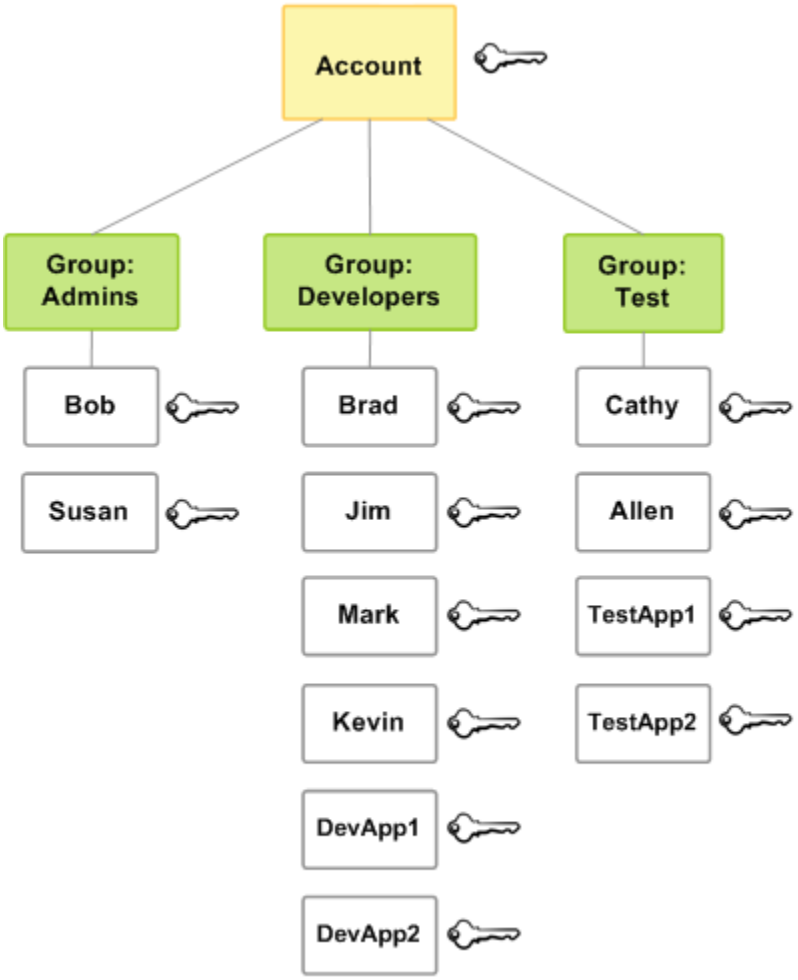
Você pode anexar uma política baseada em identidade a um grupo de usuários para que todos os usuários do grupo de usuários recebem as permissões da política. Não é possível identificar um grupo de usuários como `Principal` em uma política (como uma política baseada em recursos) porque os grupos são relacionados 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.

Veja a seguir algumas características importantes de grupos de usuários:

- 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; eles podem conter apenas usuários, não outros grupos de usuários.
- 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 de usuários consiste em usuários (humanos e aplicações) que interagem com a AWS (Jim, Brad, DevApp1 e assim por diante). 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 de usuários.



Criação de grupos de usuários 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 AWS IAM Identity Center, consulte Manage identities in IAM Identity Center (Gerenciar

identidades no Centro de Identidade do IAM) no Guia do usuário do AWS IAM Identity Center para obter informações sobre a criação de usuários e grupos no Centro de Identidade do IAM.

Para configurar um grupo de usuários, você precisa criar o grupo. Em seguida, conceda ao grupo permissões com base no tipo de trabalho que você espera que os usuários no grupo façam. Por fim, adicione os usuários ao grupo.

Para obter informações sobre as permissões necessárias para criar um grupo de usuários, consulte Permissões necessárias para acessar recursos do IAM.

Para criar um grupo de usuários do IAM e anexar políticas (console)

1. Faça login no AWS Management Console 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. Eles não são diferenciados por letras 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 Create group (Criar grupo).

Para criar grupos de usuários do IAM (AWS CLI ou API da AWS)

Use uma das seguintes opções:

- AWS CLI: aws iam create-group
- API da AWS: CreateGroup

Gerenciar grupos de usuários do IAM

A Amazon Web Services oferece várias ferramentas para gerenciar grupos de usuários do IAM. Para obter informações sobre as permissões necessárias para adicionar e remover usuários de um grupo de usuários, consulte Permissões necessárias para acessar recursos do IAM.

Tópicos

- Listar grupos de usuários do IAM
- Adicionar e remover usuários de um grupo de usuários do IAM
- Anexar uma política a um grupo de usuários do IAM
- Renomeação de um grupo de usuários do IAM
- Exclusão de um grupo de usuários do IAM

Listar grupos de usuários do IAM

Você pode listar todos os grupos de usuários em sua conta, listar os usuários em um grupo de usuários e listar os grupos de usuários aos quais um usuário pertence. Se você usar a AWS CLI ou a API da AWS, poderá listar todos os grupos de usuários com um prefixo de caminho específico.

Para listar todos os grupos de usuários em sua conta

Faça o seguinte:

- AWS Management Console: no painel de navegação, escolha User groups (Grupos de usuários).
- AWS CLI: aws iam list-groups
- API da AWS: ListGroups

Para listar os usuários em um grupo de usuários específico

Faça o seguinte:

- AWS Management Console: no painel de navegação, escolha User groups (Grupos de usuários), escolha o nome do grupo e a guia Users (Usuários).
- AWS CLI: aws iam get-group
- API da AWS: GetGroup

Para listar todos os grupos de usuários em que um usuário se encontra

Faça o seguinte:

- AWS Management Console: No painel de navegação, selecione Usuários, escolha o nome do usuário e, então, selecione a guia Grupos.
- AWS CLI: aws iam list-groups-for-user
- API da AWS: ListGroupsForUser

Adicionar e remover usuários de um grupo de usuários do IAM

Use grupos de usuários para aplicar as mesmas políticas de permissões em vários usuários de uma só vez. Você pode adicionar ou remover usuários de um grupo de usuários do IAM. Isso é útil à medida que pessoas entram e saem de sua organização.

Visualizar acesso à política

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 as informações do último acesso.

Adicionar ou remover um usuário de um grupo de usuários (console)

Você pode usar o AWS Management Console para adicionar ou remover um usuário de um grupo de usuários.

Para adicionar um usuário a um grupo de usuários do IAM (console)

1. Faça login no AWS Management Console 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).

Para remover um usuário de um grupo do IAM (console)

1. Faça login no AWS Management Console 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).

Adicione ou remova um usuário de um grupo de usuários (AWS CLI)

Você pode usar o AWS CLI para adicionar ou remover um usuário de um grupo de usuários.

Para adicionar um usuário a um grupo de usuários do IAM (AWS CLI)

- Use o seguinte comando :
  - aws iam add-user-to-group

Para remover um usuário de um grupo de usuários do IAM (AWS CLI)

- Use o seguinte comando :
  - aws iam remove-user-from-group

Adicionar ou remover um usuário de um grupo de usuários (API da AWS)

Você pode usar a API da AWS para adicionar ou remover um usuário de um grupo de usuários.

Para adicionar um usuário a um grupo do IAM (API da AWS)

- Conclua a seguinte operação:

- AddUserToGroup

Para remover um usuário de um grupo de usuários do IAM (API da AWS)

- Conclua 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 Criação de políticas do IAM.

Para obter informações sobre permissões e políticas, consulte Gerenciamento de acesso para recursos da AWS.

Para anexar uma política a um grupo de usuários (console)

1. Faça login no AWS Management Console 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 de usuários do IAM e escolha Anexar políticas.

Para anexar uma política a um grupo de usuários (AWS CLI ou API da AWS)

Realize um dos procedimentos a seguir:

- AWS CLI: aws iam attach-group-policy
- API da AWS: AttachGroupPolicy

Renomeação de 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 ao grupo de usuários 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

Faça o seguinte:

- AWS Management Console: no painel de navegação, escolha User groups (Grupos de usuários) e, em seguida, selecione o nome do grupo. Escolha Editar. Digite o novo nome do grupo de usuários e escolha Save changes (Salvar alterações).
- AWS CLI: aws iam update-group
- API da AWS: UpdateGroup

Exclusão de um grupo de usuários do IAM

Quando você exclui um grupo de usuários do AWS Management Console, o console remove automaticamente todos os membros do grupo, desanexa todas as políticas gerenciadas anexadas e exclui todas as políticas em linha. No entanto, como o IAM não exclui automaticamente as políticas que se referem ao grupo de usuários como um recurso, você deve ter cuidado ao excluir um grupo de usuários. Antes de excluir seu grupo de usuários, você deve verificar manualmente todas 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.

Para encontrar as políticas que se referem ao grupo de usuários 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 excluída.
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 Excluir para excluir a política.
8. Digite o nome da política para confirmar a exclusão da política e escolha Excluir.

Por outro lado, ao usar a AWS CLI, o Tools for Windows PowerShell ou a API da AWS para excluir um grupo de usuários, você deve primeiro remover os usuários do grupo. Em seguida, exclua todas as políticas em linha incorporadas ao grupo de usuários. Em seguida, desanexe todas as políticas gerenciadas anexadas ao grupo. Só então você pode excluir o próprio grupo de usuários.

Exclusão de um grupo de usuários do IAM (console)

Você pode excluir um grupo de usuários do IAM do AWS Management Console.

Para excluir um grupo de usuários do IAM (console)

1. Faça login no AWS Management Console 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 de usuários, marque a caixa de seleção ao lado dos nomes dos grupos de usuários a serem excluídos. Você pode usar a caixa de pesquisa para filtrar a lista de grupos de usuários por tipo, permissões e nome de grupo de usuários.
4. Escolha Delete (Excluir).
5. Na caixa de confirmação, se quiser excluir um único grupo de usuários, digite o nome do grupo de usuários e escolha Delete (Excluir). Se você quiser excluir vários grupos de usuários, digite o número de grupos de usuários a serem excluídos seguido por **`user groups`** e escolha Delete (Excluir). Por exemplo, se você excluir três grupos de usuários, digite **`3 user groups`**.

Exclusão de um grupo de usuários do IAM (AWS CLI)

Você pode excluir um grupo de usuários do IAM do AWS CLI.

Para excluir um grupo de usuários do IAM (AWS CLI)

1. Remova todos os usuários do grupo de usuários.
   - aws iam get-group (para obter a lista de usuários no grupo de usuários) e aws iam remover-user-from-group (para remover um usuário do grupo de usuários)
2. Exclua todas as políticas em linha incorporadas no grupo de usuários.
   - aws iam list-group-policies (para obter uma lista das políticas em linha do grupo de usuários) e aws iam delete-group-policy (para excluir as políticas em linha do grupo de usuários)
3. Desanexe todas as políticas gerenciadas anexadas ao grupo de usuários.

- aws iam list-attached-group-policies (para obter uma lista das políticas gerenciadas anexadas ao grupo de usuários) e aws iam detach-group-policy (para desanexar uma política gerenciada do grupo de usuários)

4. Exclua o grupo de usuários.

   - aws iam delete-group

Exclusão de um grupo de usuários do IAM (API da AWS)

Você pode usar a API da AWS para excluir um grupo de usuários do IAM.

Para excluir um grupo de usuários do IAM (API da AWS)

1. Remova todos os usuários do grupo de usuários.

   - GetGroup (para obter a lista de usuários no grupo de usuários) e RemoveUserFromGroup (para remover um usuário do grupo de usuários)

2. Exclua todas as políticas em linha incorporadas no grupo de usuários.

   - ListGroupPolicies (para obter uma lista das políticas em linha do grupo de usuários) e DeleteGroupPolicy (para excluir as políticas em linha do grupo de usuários)

3. Desanexe todas as políticas gerenciadas anexadas ao grupo de usuários.

   - ListAttachedGroupPolicies (para obter uma lista das políticas gerenciadas anexadas ao grupo de usuários) e DetachGroupPolicy (para desanexar uma política gerenciada do grupo de usuários)

4. Exclua o grupo de usuários.

   - 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.

Você pode usar funções para delegar acesso a usuários, aplicativos 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.

> **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 ao serviço é um tipo de perfil de serviço vinculado a um AWS service (Serviço da AWS). O serviço pode assumir a função de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode 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 Serviços da AWS 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 a esse serviço.

Tópicos

Termos e conceitos das funções

Veja a seguir alguns termos básicos para começar a usar as funções.

Função

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.

As funções podem ser usadas pelas seguintes entidades:

- Um usuário do IAM na mesma Conta da AWS que o perfil
- Um usuário do IAM em uma Conta da AWS diferente do perfil
- Um serviço da Web oferecido pela AWS, como o Amazon Elastic Compute Cloud (Amazon EC2)
- Um usuário externo autenticado por um serviço de provedor de identidade (IdP) compatível com SAML 2.0 ou OpenID Connect ou um identity broker personalizado.

Função de serviço da AWS

O perfil de serviço é um perfil do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do usuário do IAM.

AWS Função de serviço da para uma instância do EC2

Um tipo especial de função de serviço que uma aplicação em execução em uma instância do Amazon EC2 pode assumir para executar ações em sua conta. Essa função é atribuída à instância do EC2 ao ser executada. Os aplicativos em execução nessa instância podem recuperar credenciais de segurança temporárias e realizar ações que a função permitir. Para obter detalhes sobre o uso de uma função de serviço para uma instância do EC2, consulte Uso de uma função do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2.

Função vinculada ao serviço da AWS

Um perfil vinculado ao serviço é um tipo de perfil de serviço vinculado a um AWS service (Serviço da AWS). O serviço pode assumir a função de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode 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 ter 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 Serviços da AWS 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 a esse serviço. Para ter mais informações, consulte Usar funções vinculadas ao serviço.

Encadeamento de funções

O encadeamento de funções ocorre quando você usa uma função para assumir uma segunda função por meio da AWS CLI ou da API. Por exemplo, suponha que o `RoleA` tenha permissão para assumir o `RoleB`. Você pode 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 funções limita a sessão da função da AWS CLI ou da API da AWS em um máximo de uma hora. 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`. Você pode especificar um valor de parâmetro de até 43200 segundos (12 horas), dependendo da configuração da duração máxima da sessão para sua função. 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.

A AWS não trata o uso de funções para conceder permissões a aplicativos executados em instâncias do EC2 como encadeamento de funções.

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 uma função 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.

Federação

A criação de uma relação de confiança entre um provedor de identidade externo e a AWS. Os usuários podem fazer login em um provedor OIDC, como Login with Amazon, Facebook, Google ou qualquer IdP compatível com o OpenID Connect (OIDC). Os usuários também podem fazer login em um sistema de identidade empresarial que seja compatível com Security Assertion Markup Language (SAML) 2.0, como o Microsoft Active Directory Federation Services. Quando você usa o OIDC e o SAML 2.0 para configurar uma relação de confiança entre esses provedores de identidade externos e a AWS, o usuário é atribuído a uma função do IAM. O usuário também recebe credenciais temporárias que permitem que ele acesse seus recursos da AWS.

Usuário federado

Em vez de criar um usuário do IAM, você pode usar identidades existentes do AWS Directory Service, do diretório de usuário da sua empresa ou de um provedor OIDC. Eles são conhecidos como usuários federados. A AWS atribui uma função a um usuário federado quando o acesso é solicitado por meio de um provedor de identidades. Para obter mais informações sobre usuários federados, consulte Usuários federados e funções.

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.

Política de permissões

Um documento de permissões no formato JSON no qual você define quais ações e recursos a função pode usar. O documento é criado de acordo com as regras da linguagem da política do IAM.

Limite de permissões

Um recurso avançado no qual 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 ter mais informações, consulte Limites de permissões para entidades do IAM.

Principal

Entidade na AWS que pode executar ações e acessar recursos. Uma entidade principal pode ser um Usuário raiz da conta da AWS, um usuário do IAM ou um perfil. Você pode conceder permissões para acessar um recurso de uma das seguintes formas:

- Você pode anexar uma política de permissões a um usuário (diretamente ou indiretamente por meio de um grupo) ou a uma função.
- Para os serviços que dão suporte a políticas baseadas em recursos, você pode identificar as entidades principais no elemento `Principal` de uma política anexada ao recurso.

Se você faz referência a uma Conta da AWS como entidade principal, geralmente significa qualquer entidade principal definida nessa conta.

> **Note**
>
> Não é possível usar um curinga (*) para fazer a correspondência de parte de um nome de entidade principal ou ARN em uma política de confiança de função. Para obter detalhes, consulte Elementos da política JSON da AWS:Principal.

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 S3 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 Serviços da AWS que funcionam com o IAM. Para obter mais informações sobre políticas baseadas em recursos, consulte Acesso a recursos entre contas no IAM.

Cenários comuns para funções: usuários, aplicações e serviços

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)

Fornecer 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 Fornecer acesso a 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 AWS Management Console 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 Configuração de acesso à API protegido por MFA.

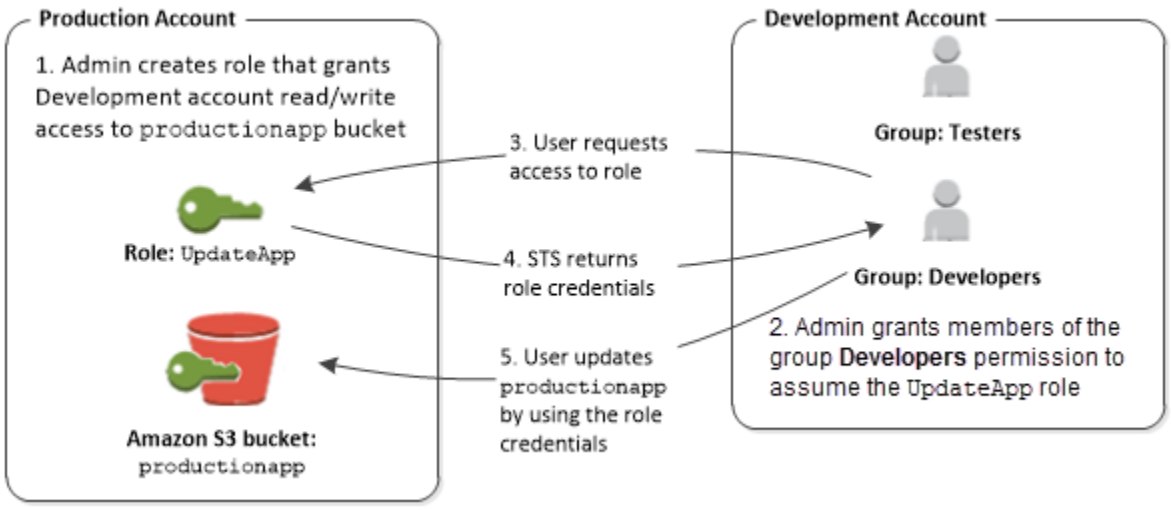
Recomendamos usar essa abordagem para aplicar o princípio do privilégio mínimo. Isso significa restringir o uso de permissões elevadas para apenas quando elas forem necessárias para tarefas específicas. Com as funções, você pode ajudar a evitar alterações acidentais em ambientes confidenciais, especialmente se você combiná-las com uma auditoria para ajudar a garantir que as funções sejam usadas 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 Configuração de acesso à API protegido por 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 `AssumeRole` do AWS Security Token Service (AWS STS) 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 a função:
   - 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 clicar em 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. O AWS STS retorna credenciais temporárias:
   - Console da AWS: o AWS STS verifica a solicitação com a política de confiança da função para garantir 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 da função para garantir 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.

Mais informações

Para obter mais informações, consulte as informações a seguir.

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

Como fornecer acesso a workloads que não são 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 datacenter ou em outra infraestrutura fora da AWS que precise 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, você configura uma CA usando o AWS Private Certificate Authority ou usa uma CA de sua própria infraestrutura de PKI. Depois de configurar uma CA, você cria um objeto no IAM Roles Anywhere chamado de âncora de confiança para estabelecer confiança entre o IAM Roles Anywhere e sua CA para autenticação. Em seguida, você pode configurar seus perfis existentes do IAM ou criar novos perfis que confiam no serviço do IAM Roles Anywhere. Quando suas workloads que não são da AWS são autenticadas com o IAM Roles Anywhere usando a âncora de confiança, elas podem obter credenciais temporárias para seus perfis do IAM a fim de acessar seus recursos 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.

Fornecer acesso a 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 à função. 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. Para obter mais informações sobre o ID externo, consulte Como usar um ID externo ao conceder acesso aos seus recursos da AWS a terceiros.
- 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.

Como usar um ID externo ao conceder acesso aos seus recursos da AWS a terceiros

Às vezes, é necessário oferecer acesso a terceiros para os recursos da AWS (delegar acesso). Um aspecto importante desse cenário é o ID externo, informação opcional que você usa em uma política de confiança da função do IAM para designar quem pode assumir a função.

> **Important**
>
> A AWS 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.

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.

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 Solicitação de credenciais de segurança temporárias.

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.

Por que usar um ID externo?

Em termos abstratos, o ID externo permite que o usuário que está assumindo a função 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 de "confused deputy".

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.

Fornecimento de acesso a um produto 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 de serviço para instâncias do EC2 ou 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 uma função para delegar permissões a um serviço da AWS.

O problema de "confused deputy"

O problema do substituto confuso 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).

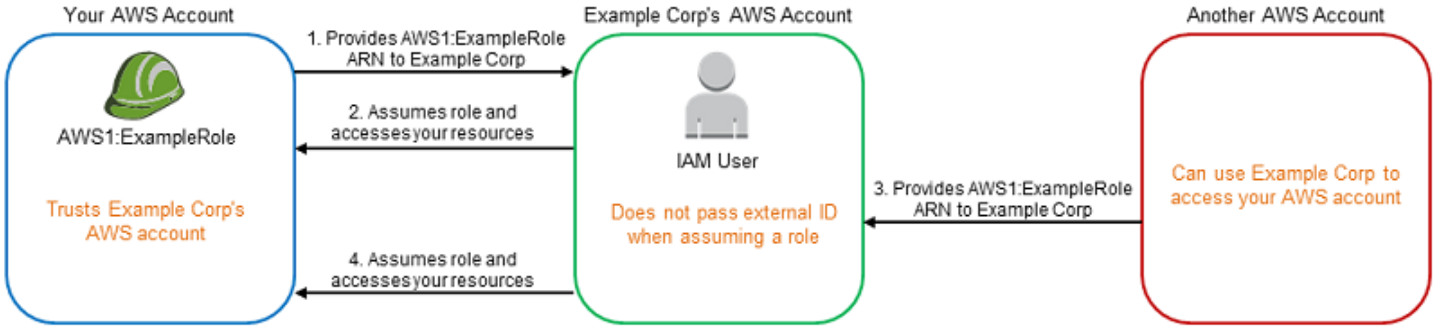
Às vezes, poderá ser necessário conceder acesso aos recursos da AWS a terceiros (delegar acesso). Por exemplo, suponha que você decida contratar uma empresa terceirizada chamada Example Corp para monitorar sua Conta da AWS e ajudar a otimizar os custos. Para rastrear seus gastos diários, a Example Corp precisa de acesso aos seus recursos da AWS. A Example Corp também monitora muitas outras Contas da AWS para outros clientes. Você pode 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, uma informação opcional que você pode

usar em uma política de confiança de função do IAM para designar quem pode assumir a função. A função principal do ID externo é abordar e impedir o problema "confused deputy".

Na AWS, a personificação entre serviços pode resultar no problema do substituto confuso. A imitação entre serviços pode ocorrer quando um serviço (o serviço de chamada) chama outro serviço (o serviço chamado). O serviço de chamada pode ser manipulado de modo a usar suas permissões para atuar nos recursos de outro cliente de uma forma na qual ele não deveria ter permissão para acessar.

Prevenção de substituto confuso entre contas

O diagrama a seguir ilustra o problema do substituto confuso 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 do substituto confuso 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 substituto confuso e conceda acesso a recursos da AWS de outra conta.

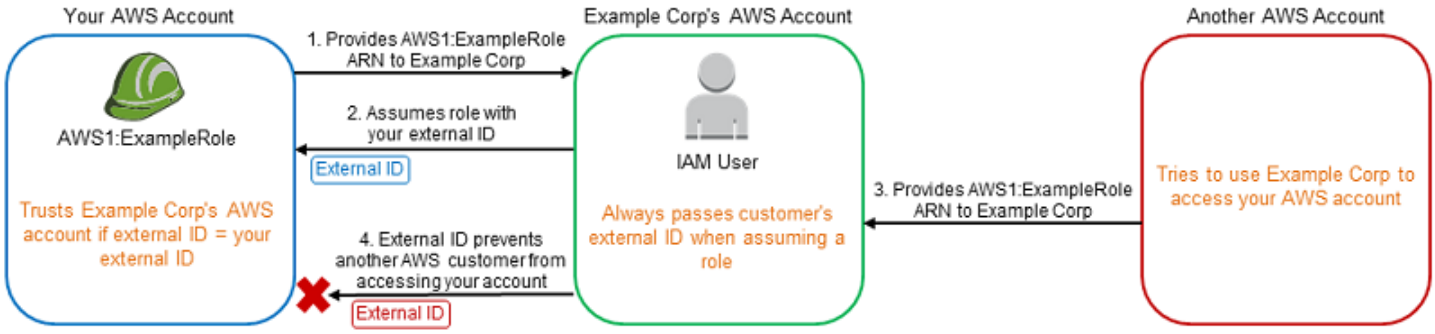
Em nosso cenário, imagine que seu identificador exclusivo da Example Corp seja 12345 e que o identificador do outro cliente seja 67890. Esses identificadores são simplificados para esse cenário. Em geral, esses identificadores são GUIDs. Supondo que esses identificadores sejam exclusivos entre os clientes 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 da função que exige que o valor do `sts:ExternalId` seja 12345, como:

```
{
  "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 do substituto confuso entre serviços

Recomendamos o uso das chaves de contexto de condições globais `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths` em políticas baseadas em recursos para limitar as permissões de um serviço para um determinado recurso. Use `aws:SourceArn` se quiser associar apenas um recurso ao acesso entre serviços. Use `aws:SourceAccount` se quiser permitir que qualquer recurso nessa conta seja associado ao uso entre serviços. Use `aws:SourceOrgID` se quiser permitir que qualquer recurso de qualquer conta de uma organização seja associado ao uso entre serviços. Use `aws:SourceOrgPaths` se quiser associar qualquer recurso das contas em um caminho do AWS Organizations seja associado ao uso entre serviços. Para obter mais informações sobre como usar e entender os caminhos, consulte Entender o caminho da entidade do AWS Organizations.

A maneira mais granular de se proteger contra o problema de "confused deputy" é usar a chave de contexto de condição global `aws:SourceArn` com o ARN completo do recurso nas políticas baseadas em recursos. Se você não souber o ARN completo do recurso ou estiver especificando vários recursos, use a chave de condição de contexto global `aws:SourceArn` com curingas (`*`) para as partes desconhecidas do ARN. Por exemplo, `arn:aws:servicename:*:123456789012:*`.

Se o valor do `aws:SourceArn` não contiver o ID da conta, como um ARN de bucket do Amazon S3, você deverá usar ambos, a `aws:SourceAccount` e o `aws:SourceArn` para limitar as permissões.

Para se proteger do problema de "confused deputy" em grande escala, use a chave de contexto de condição global `aws:SourceOrgID` ou `aws:SourceOrgPaths` com o ID ou o caminho da organização do recurso nas políticas baseadas em recursos. As políticas que incluem a chave `aws:SourceOrgID` ou `aws:SourceOrgPaths` incluem automaticamente as contas corretas e você não tem que atualizar manualmente as políticas quando adiciona, remove ou move contas na organização.

Para políticas de confiança de função não vinculadas a serviços, cada serviço na política de confiança executou a ação `iam:PassRole` para verificar se a função está na mesma conta do serviço chamador. Por isso, não é necessário usar `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths` com essas políticas de confiança. O uso de `aws:SourceArn` em uma política de confiança permite especificar recursos dos quais uma função pode ser assumida, como o ARN de uma função do Lambda. Alguns Serviços da AWS usam `aws:SourceAccount` e `aws:SourceArn` em políticas de confiança para perfis recém-criados, mas o uso das chaves não é necessário para os perfis já existentes na conta.

> **Note**
>
> Os Serviços da AWS que se integram com o AWS Key Management Service usando concessões de chaves do KMS não são compatíveis com as chaves de condições `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths`. O uso dessas chaves de condições em uma política de chave do KMS causará um comportamento inesperado se a chave também for usada pelos Serviços da AWS por meio de concessões de chaves do KMS.

Prevenção do problema do substituto confuso entre serviços para AWS Security Token Service

Muitos serviços da AWS exigem que você use funções para permitir que o serviço acesse recursos de outro serviço em seu nome. A função de serviço é uma função que um serviço assume para realizar ações em seu nome. Uma função requer duas políticas: uma política de confiança de função que especifica qual entidade principal tem permissão para assumir a função e uma política de permissões que especifica o que pode ser feito com a função. Uma política de confiança de função é o único tipo de política baseada em recursos no IAM. Outros Serviços da AWS têm políticas baseadas em recursos, como uma política de bucket do Amazon S3.

Quando um serviço assume uma função em seu nome, a entidade principal de serviço deve ter permissão para executar a ação `sts:AssumeRole` na política de confiança de função. Quando um serviço chama `sts:AssumeRole`, o AWS STS retorna um conjunto de credenciais temporárias de segurança usado pela entidade principal de serviço para acessar os recursos permitidos pela política de permissões da função. Quando um serviço assume um perfil na sua conta, você pode incluir as chaves de contexto de condições globais `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths` na política de confiança do perfil para limitar o acesso ao perfil apenas às solicitações que forem geradas pelos recursos esperados.

Por exemplo, em AWS Systems Manager Incident Manager, você deve escolher um perfil que permita que o Incident Manager execute um documento de automação do Systems Manager em seu nome. O documento de automação pode incluir planos de resposta automatizados para incidentes iniciados por alarmes do CloudWatch ou eventos EventBridge. No exemplo de política de confiança de função a seguir, você pode usar a chave de condição `aws:SourceArn` para restringir o acesso à função de serviço com base no ARN do registro de incidente. Somente registros de incidentes criados com base no recurso do plano de resposta `myresponseplan` são capazes de usar essa função.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "ssm-incidents.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:ssm-incidents:*:111122223333:incident-
record/myresponseplan/*"
      }
    }
  }
}
```

> **Note**
>
> Nem todas as integrações de serviços com o AWS STS são compatíveis com as chaves de condições `aws:SourceArn`, `aws:SourceAccount`, `aws:SourceOrgID` ou `aws:SourceOrgPaths`. O uso dessas chaves nas políticas de confiança do IAM com integrações que não são compatíveis pode resultar em um comportamento inesperado.

Fornecer 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).

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 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 ter mais informações, consulte Usar o 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 AWS Management Console 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

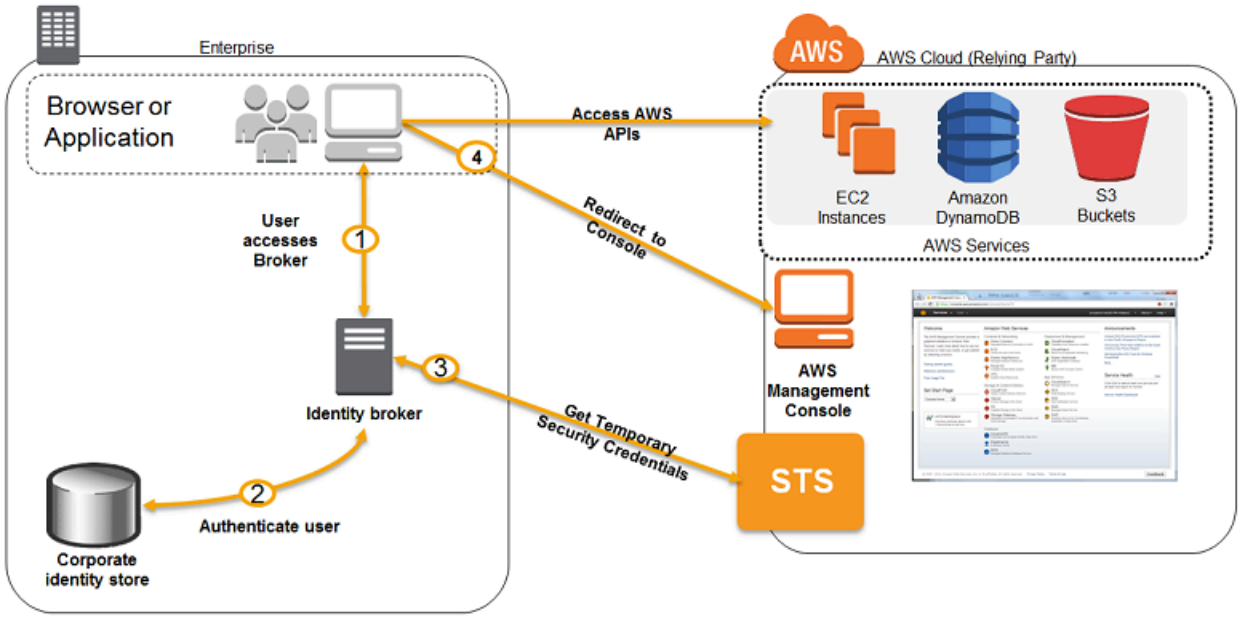
Se o armazenamento de identidades não for compatível com o SAML 2.0, você poderá criar um aplicativo identity broker personalizado para executar uma função semelhante. O aplicativo agente

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 Controle de permissões para 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 agente 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 Solicitação de credenciais de segurança temporárias. Para obter mais informações sobre como os usuários federados obtêm acesso ao Console de Gerenciamento da AWS, consulte Habilitar o acesso de usuários federados SAML 2.0 ao AWS Management Console.

Usar funções vinculadas 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. A função de serviço é uma função do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do usuário do IAM. Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um AWS service (Serviço da AWS). O serviço pode assumir a função de executar uma ação em seu nome. Os perfis vinculados ao serviço aparecem em sua Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para funções vinculadas 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, e essa política não pode ser anexada a nenhuma outra entidade do IAM.

Para poder excluir as funções, você deve primeiroexcluir os recursos relacionados a eles. Isso protege seus recursos do , pois você não pode remover por engano as permissões para acessar os recursos.

> **Tip**
>
> Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte Serviços da AWS 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 da função vinculada a esse serviço.

Permissões de função vinculada 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 da função vinculada 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.

```
{
    "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 Conceder permissões a um usuário para passar uma função 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 uma função vinculada ao serviço é usada por um serviço da AWS, essa função vinculada 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 sua função vinculada ao serviço para acessar esses recursos.

Criar uma função vinculada 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 Serviços da AWS 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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Roles (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 Next (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 Next (Avançar).

   > **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 Role name (Nome da função), o grau de personalização do nome da função 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 da AWS. Eles não são diferenciados por letras maiúsculas e minúsculas. Por exemplo, não é possível criar funções chamadas **`<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 etiquetas no IAM, consulte Recursos de etiquetas do IAM.
9. Revise a função e escolha Create role (Criar função).

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`.

Editar uma função vinculada 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 Serviços da AWS 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 Roles (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 Save (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

Excluir uma função vinculada ao serviço

O método que você usa para criar uma função vinculada 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 AWS CLI do serviço.

Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte Serviços da AWS 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 exclusão a função vinculada ao serviço, escolha o link Sim para visualizar a documentação da função vinculada desse serviço.

Se o serviço não oferecer suporte à exclusão da função, você poderá excluir a função vinculada ao serviço do console, da API ou da AWS CLI do IAM. Se você não precisar mais usar um recurso ou serviço que requer uma função vinculada a serviço, é recomendável excluí-la. Dessa forma, você não tem uma entidade não utilizada que não seja monitorada ativamente ou mantida. No entanto, você deve limpar sua função vinculada ao serviço antes de excluí-la.

Limpar uma função vinculada ao serviço

Antes de você poder usar o IAM para excluir uma função vinculada ao serviço, você deve primeiro confirmar que a função não tem sessões ativas e remover quaisquer recursos usados pela função.

Para verificar se a função vinculada ao serviço tem uma sessão ativa no console do IAM

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Roles (Perfis). Então, escolha o nome (não a caixa de marcação) da função vinculada ao serviço.
3. Na página Resumo para a função selecionada, escolha a guia Consultor de Acesso.
4. Na guia Consultor de Acesso, revise a atividade recente para a função vinculada ao serviço.

> **Note**
>
> Se você não tem certeza se o serviço está usando a função vinculada ao serviço, pode tentar excluir a função. Se o serviço está usando a função, a exclusão falha e você pode visualizar as regiões em que a função está sendo usada. Se a função está sendo usada, você deve aguardar a sessão final antes de excluir a função. Você não pode revogar a sessão para uma função vinculada a serviço.

Para remover os recursos usados por uma função vinculada ao serviço

Para obter informações sobre quais serviços oferecem suporte a funções vinculadas a serviços, consulte Serviços da AWS 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 exclusão a função vinculada ao serviço, escolha o link Sim para visualizar a documentação da função vinculada desse serviço. Consulte a documentação daquele serviço para saber como remover usado pela sua função vinculada a esse serviço.

Excluir uma função vinculada ao serviço (console)

É possível usar o console do IAM para excluir uma função vinculada ao serviço.

Para excluir uma função vinculada ao serviço (console)

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Roles. Selecione a caixa de marcação ao lado do nome da função que você deseja excluir, não o nome ou a linha em si.
3. Em Role actions (Ações da função) 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 ativo no momento. Se você deseja continuar, escolha Sim, excluir para enviar a função vinculada ao serviço para exclusão.
5. Observe as notificações do console do IAM para monitorar o progresso da exclusão da função vinculada ao serviço. Como a exclusão da função vinculada ao serviço do IAM é assíncrona, depois de enviar a função para exclusão, a tarefa de exclusão pode ou não ser bem-sucedida.

   - Se a tarefa for bem-sucedida, a função será removida da lista e uma notificação de sucesso será exibida na parte superior da página.
   - Se a tarefa falhar, você pode escolher Visualizar detalhes ou Exibir recursos a partir das notificações para saber por que a exclusão falhou. Se a exclusão falhar porque a função está usando os recursos do serviço, a notificação inclui uma lista de recursos, se o serviço retorna essas informações. Você pode então limpar os recursos e submeter novamente a exclusão.

     > **Note**
     >
     > Você pode repetir esse processo várias vezes, de acordo com as informações que o serviço retorna. Por exemplo, a função vinculada ao serviço pode usar seis recursos e seu serviço pode retornar informações sobre cinco deles. Se você limpar cinco recursos e enviar a função para exclusão novamente, a deleção falha e o serviço reporta o recurso remanescente. Um serviço pode retornar todos os recursos, alguns deles, ou pode não reportar nenhum recurso.

   - Se a tarefa falhar e a notificação não inclui uma lista de recursos, o serviço pode não retornar essas informações. Para saber como limpar os recursos para esse serviço, consulte Serviços da AWS que funcionam com o IAM. Encontre o serviço na tabela e escolha o link Sim para visualizar a documentação da função vinculada desse serviço.

Excluir uma função vinculada ao serviço (AWS CLI)

Você pode usar comandos do IAM na AWS CLI para excluir uma função vinculada ao serviço.

Para excluir uma função vinculada ao serviço (AWS CLI)

1. Se você souber o nome da função vinculada ao serviço que deseja excluir, insira o seguinte comando para listar a função na conta:

```
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. Como uma função vinculada ao serviço não podem ser excluída se estiver sendo usada ou tem recursos associados, você deve enviar uma solicitação de exclusão. Essa solicitação pode ser negada se essas condições não forem atendidas. Você deve capturar o `deletion-task-id` da resposta para verificar o estado da tarefa de exclusão. Insira o seguinte comando para enviar uma solicitação de exclusão de função vinculada ao serviço:

```
aws iam delete-service-linked-role --role-name role-name
```

3. Insira o seguinte comando para conferir o estado da tarefa de exclusão:

```
aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
```

O status da tarefa de exclusão pode ser `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`, ou `FAILED`. Se a exclusão falhar, a chamada retorna o motivo de falha para que você possa solucionar problemas. Se a exclusão falhar porque a função está usando os recursos do serviço, a notificação inclui uma lista de recursos, se o serviço retorna essas informações. Você pode então limpar os recursos e submeter novamente a exclusão.

> **Note**
>
> Você pode repetir esse processo várias vezes, de acordo com as informações que o serviço retorna. Por exemplo, a função vinculada ao serviço pode usar seis recursos e seu serviço pode retornar informações sobre cinco deles. Se você limpar cinco recursos e enviar a função para exclusão novamente, a deleção falha e o serviço reporta o recurso remanescente. Um serviço pode retornar todos os recursos, alguns deles, ou pode não reportar nenhum recurso. Para saber como limpar os recursos para um serviço que não reporta nenhum recurso, consulte Serviços da AWS que funcionam com o IAM. Encontre o serviço na tabela e escolha o link Sim para visualizar a documentação da função vinculada desse serviço.

Excluir uma função vinculada ao serviço (API da AWS)

É possível usar a API do AWS para excluir uma função vinculada ao serviço.

Para excluir uma função vinculada ao serviço (AWS API)

1. Para enviar uma solicitação de exclusão de uma função vinculada ao serviço, chame DeleteServiceLinkedRole. Na solicitação, especifique o nome da função.

   Como uma função vinculada ao serviço não podem ser excluída se estiver sendo usada ou tem recursos associados, você deve enviar uma solicitação de exclusão. Essa solicitação pode ser negada se essas condições não forem atendidas. Você deve capturar o `DeletionTaskId` da resposta para verificar o estado da tarefa de exclusão.
2. Para verificar o status da exclusão, chame GetServiceLinkedRoleDeletionStatus. Na solicitação, especifique o `DeletionTaskId`.

   O status da tarefa de exclusão pode ser `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`, ou `FAILED`. Se a exclusão falhar, a chamada retorna o motivo de falha para que você possa solucionar problemas. Se a exclusão falhar porque a função está usando os recursos do serviço, a notificação inclui uma lista de recursos, se o serviço retorna essas informações. Você pode então limpar os recursos e submeter novamente a exclusão.

   > **Note**
   >
   > Você pode repetir esse processo várias vezes, de acordo com as informações que o serviço retorna. Por exemplo, a função vinculada ao serviço pode usar seis recursos e seu serviço pode retornar informações sobre cinco deles. Se você limpar cinco recursos e enviar a função para exclusão novamente, a deleção falha e o serviço reporta o recurso remanescente. Um serviço pode retornar todos os recursos, alguns deles, ou pode não reportar nenhum recurso. Para saber como limpar os recursos para um serviço que não reporta nenhum recurso, consulte Serviços da AWS que funcionam com o IAM. Encontre o serviço na tabela e escolha o link Sim para visualizar a documentação da função vinculada desse serviço.

Criação de funções do IAM

Para criar uma função você pode usar o AWS Management Console, a AWS CLI, o Tools for Windows PowerShell ou a API do IAM.

Se você usar o AWS Management Console, um assistente orienta você durante as etapas para a criação de uma função. O assistente tem etapas ligeiramente diferentes, dependendo de você estar criando um perfil para um serviço da AWS, para uma Conta da AWS ou para um usuário federado.

Tópicos

Criação de uma função para delegar permissões a um usuário do IAM

Você pode usar funções 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 `sts:ExternalID` correto. Para ter mais informações, consulte Como usar um ID externo ao conceder acesso aos seus recursos da AWS a 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 uma função para delegar permissões a um serviço da AWS.

Criação de uma função do IAM (console)

Você pode usar o AWS Management Console 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 (console)

1. Faça login no AWS Management Console 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 `sts:ExternalID` correto. Para ter mais informações, consulte Como usar um ID externo ao conceder acesso aos seus recursos da AWS a 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 AWS Management Console 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 Uso de autenticação multifator (MFA) na AWS
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 ter 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) Para 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 etiquetas no IAM, consulte Recursos de etiquetas do IAM.
15. Revise a função 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 Concessão de permissões a um usuário para alternar funções.

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 ter 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 Uso de autenticação multifator (MFA) na AWS.

```
{
  "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.

```
{
  "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 Concessão de permissões a um usuário para alternar funções.

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 ter mais informações, consulte Alternância para uma função 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 Concessão de permissões a um usuário para alternar funções.
3. (Opcional) Adicione atributos personalizados ao usuário anexando tags: TagRole

Para ter mais informações, consulte Gerenciamento de etiquetas em usuários 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 Alternância para uma função 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 uma função 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 uma função atende a uma finalidade especial de um serviço, ela é categorizada como uma função de serviço para instâncias do EC2 (por exemplo) ou uma função vinculada 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 Serviços da AWS 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 da função vinculada 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.

```
{
    "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

A AWS 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.

```
{
    "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

A AWS 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)

Você pode usar o AWS Management Console para criar uma função 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. Você pode saber como atribuir as políticas de confiança e as permissões necessárias à função para que o serviço possa assumir a função 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.

Para criar uma função para um AWS service (Serviço da AWS) (console do IAM)

1. Faça login no AWS Management Console 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 política de permissão alguma, 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 Marcar recursos do IAM no Guia do usuário do IAM.
13. Reveja a função e escolha Create role (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 ter 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 de instância pode conter somente uma função do e esse limite não pode ser aumentado. Se você criar a função usando o AWS Management Console, 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 uma função, consulte Controlar o acesso aos recursos do Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux.

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.

```
{
  "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.

```
{
  "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 usuário anexando tags: TagRole

   Para ter mais informações, consulte Gerenciamento de etiquetas em usuários 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 AWS Management Console, 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 uma função, consulte Controlar o acesso aos recursos do Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux.

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 uma função para um provedor de identidade de terceiros (federação)

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 Provedores de identidade e federação.

Criar uma função para usuários federados (console)

Os procedimentos para criar um perfil para usuários federados 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 uma função para a federação de identidades (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 ter 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 Configurando 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.

```
{
    "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.

```
{
  "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 uma função para a federação de identidades (API da 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 usuário anexando tags: TagRole

   Para ter mais informações, consulte Gerenciamento de etiquetas em usuários 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)

Você pode 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 Provedores de identidade e federação.

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.) Você pode configurar vários

aplicativos 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 ter 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 AWS Management Console. 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.

```
{
    "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 uma 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 a URL totalmente qualificada 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.

```
{
      "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.

```
{
      "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.

```
{
      "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.

```
{
      "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. O procedimento a seguir descreve como criar o perfil para federação OIDC no AWS Management Console. Para criar uma

função na AWS CLI ou na API da AWS, consulte os procedimentos em Criar uma função para um provedor de identidade de terceiros (federação).

> **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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Funções e Criar função.
3. Selecione o tipo de perfil OIDC.
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 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"). 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.

5. 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:
  - Para Audience (Público), escolha `sts.amazonaws.com`.
  - Em Organização do GitHub, insira o nome da sua 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 a 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 (*).

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 ter mais informações, consulte Modificação de uma política de confiança de função (console).
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 ter 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 etiquetas no IAM, consulte Recursos de etiquetas do IAM.
15. Revise a função 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.

```
{
  "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 Provedores de identidade e federação.

> **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 ter 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 o usuário federado 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.
- No elemento `Condition`, use uma condição `StringEquals` para verificar se o atributo `saml:aud` da resposta de SAML corresponde ao endpoint da federação do SAML para a AWS.

A política de confiança de exemplo a seguir foi projetada para um usuário federado do SAML:

```
{
    "Version": "2012-10-17",
    "Statement": {
      "Effect": "Allow",
      "Action": "sts:AssumeRoleWithSAML",
      "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/PROVIDER-
NAME"},
      "Condition": {"StringEquals": {"SAML:aud": "https://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 AWS Management Console 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 AWS Management Console access (Permitir acesso programático e do console) para criar uma função que possa ser assumida de forma programática e no AWS Management Console.

   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 login da AWS para SAML (https://signin.aws.amazon.com/saml).
6. 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 uma função para acesso programático e do console, o atributo `SAML:aud` será adicionado automaticamente e definido como o URL do endpoint da SAML AWS (https://signin.aws.amazon.com/saml).
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 ter 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 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.
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 funções chamadas **`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 etiquetas no IAM, consulte Recursos de etiquetas do IAM.
17. Revise a função 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 usem. Isso é chamado de configuração da confiança da parte confiável entre seu IdP e a AWS. Para ter mais informações, consulte Configurar o IdP SAML 2.0 com objeto de confiança de terceira parte confiável e adição de declarações.

Criar uma função usando políticas de confiança personalizadas (console)

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 AWS Management Console 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 AWS Management Console 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. Marque a caixa de seleção ao lado da política de confiança personalizada que você criou.
7. (Opcional) Defina um limite de permissões. Esse é um recurso avançado que está disponível para funções de serviço, mas não para funções vinculadas 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.
8. Escolha Next (Próximo).
9. Para Role name (Nome da função), o grau de personalização do nome da função 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 funções chamadas **`PRODROLE`** e **`prodrole`**. Como outros recursos da AWS podem fazer referência à função, não é possível editar o nome da função depois de ela ser criada.
10. (Opcional) Em Description (Descrição), insira uma descrição para a nova função.
11. Escolha Edit (Editar) nas seções Etapa 1: selecionar entidades confiáveis ou na Etapa 2: adicionar permissões para editar a politica personalizada e as permissões para a função.
12. (Opcional) Adicione metadados à função anexando tags como pares de chave-valor. Para obter mais informações sobre como usar etiquetas no IAM, consulte Recursos de etiquetas do IAM.
13. Revise a função e escolha Create role (Criar função).

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

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`.

```
{
  "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 chama-se mybucket, e o número de conta da conta B é 111122223333. Ele não especifica quaisquer usuários individuais ou grupos na conta B, apenas a conta em si.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBAccess1",
    "Effect": "Allow",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": [
```

```
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }
}
```

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 objetos e buckets do S3, acesse 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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:List*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }
}
```

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.

```
{
  "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).

```
{
  "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 chamado de mybucket, e o número da conta B é 1111-2222-3333. A conta A usa o Amazon S3 para implementar esta política.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "AccountBDeny",
    "Effect": "Deny",
    "Principal": {"AWS": "111122223333"},
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::mybucket/*"
  }
}
```

Esta negação explícita substitui todas as políticas na conta B que concedem permissão para acessar o bucket do S3 na conta A.

Uso de funções do IAM

Antes que um usuário, uma aplicação ou um serviço possa usar um perfil que você criou, você deve 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 AWS Management Console, 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 AWS Management Console, o `Path` e o `RoleName` combinados não podem exceder 64 caracteres.

Você pode mudar funções no AWS Management Console. Você pode assumir uma função chamando uma operação da AWS CLI ou da API, ou usando um URL personalizado. O método que você usa 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.

Comparação de métodos para usar funções

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)
AWS Management Console	Usuário (mudando perfis)	Maximum session duration (Duração máxima da sessão) na página Resumo Role (Função)	15 min|Configuração de duração máxima da sessão²|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)
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
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

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 `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 uma função para assumir uma função diferente 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.

[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 Modificar uma função. 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 Visualizar a configuração de duração máxima da sessão para uma função 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 às durações máximas de sessão configuradas 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 Modificação de uma política de confiança de função (console).

Tópicos

Visualizar a configuração de duração máxima da sessão para uma função

Você pode especificar a duração máxima da sessão para uma função usando o AWS Management Console, a AWS CLI ou a API da AWS. Quando usa uma operação da AWS CLI ou da API para assumir uma função, você pode especificar um valor para o parâmetro `DurationSeconds`. Você pode usar este parâmetro para especificar a duração da sessão de função, de 900 segundos (15 minutos) até a configuração de duração máxima da sessão para a função. Antes de especificar o

parâmetro, você deve verificar essa configuração para sua função. Se você especificar um valor para o parâmetro `DurationSeconds` que seja maior do que o valor máximo, a operação falhará.

Para visualizar a duração máxima da sessão de uma função (console)

1. No painel de navegação do console do IAM, escolha Perfis.
2. Escolha o nome da função que você deseja visualizar.
3. Ao lado de Maximum session duration (Duração máxima da sessão), visualize a duração máxima da sessão concedida para a função. Esta é a duração máxima da sessão que você pode especificar na sua AWS CLI ou na operação da API.

Para visualizar a configuração de duração máxima da sessão de uma função (AWS CLI)

1. Se você não souber o nome da função que deseja assumir, execute o seguinte comando para listar as funções em sua conta:
   - aws iam list-roles
2. Para visualizar a duração máxima da sessão da função, execute o comando a seguir. Em seguida, verifique o parâmetro de duração máxima da sessão.
   - aws iam get-role

Para visualizar a configuração de duração máxima da sessão de uma função (API da AWS)

1. Se você não souber o nome da função que deseja assumir, chame a seguinte operação para listar as funções em sua conta:
   - ListRoles
2. Para visualizar a duração máxima da sessão da função, execute a operação a seguir. Em seguida, verifique o parâmetro de duração máxima da sessão.
   - GetRole

Concessão de permissões a um usuário para alternar funções

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 mais 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 mais 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 no AWS Management Console para uma função 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 AWS Management Console, 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

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`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::account-id: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 Alternância para uma função (console) para orientá-los durante o processo. Para solucionar problemas comuns que você pode encontrar ao assumir uma função, 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 AWS Management Console. 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 AWS Management Console, 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.

Conceder permissões a um usuário para passar uma função 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 Serviços da AWS 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.

```
{
    "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.

```
{
    "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-`:

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "iam:GetRole",
            "iam:PassRole"
        ],
        "Resource": "arn:aws:iam::account-id: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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Selecione Funções e, em seguida, Criar função.
3. Escolha o tipo de perfil 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) Para Descrição da função, insira uma descrição para a nova função.
8. (Opcional) Adicione metadados ao usuário anexando tags como pares de chave-valor. Para obter mais informações sobre como usar etiquetas no IAM, consulte Recursos de etiquetas do IAM.
9. Revise a função 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 com 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.

Alternância para uma função (console)

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 criá-los, consulte Perfis do IAMe Criação de funções do IAM.

> **Important**
>
> As permissões do seu usuário do e quaisquer funções para as quais você mudar 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.

Quando você muda de funções no AWS Management Console, o console sempre usa suas credenciais originais para autorizar a mudança. Isso será aplicável se você 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. Por exemplo, se você alternar para FunçãoA, o IAM usará seu usuário original ou as credenciais de função federada para determinar se você tem permissão para assumir a FunçãoA. Se você tentar mudar para a RoleB enquanto estiver usando a RoleA, o AWS ainda usará o usuário original ou as credenciais da função federada para autorizar a mudança, e não as credenciais da RoleA.

O que é preciso saber sobre como alternar funções no console

Esta seção fornece informações adicionais sobre como usar o console do IAM para alternar para uma função.

> **Observações:**
>
> - Você não pode mudar de funções caso se conecte como o Usuário raiz da conta da AWS. 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.

- Não é possível alternar no AWS Management Console para uma função que exija um valor ExternalId. Você só pode alternar para tal perfil chamando a API `AssumeRole`, que é compatível com o parâmetro `ExternalId`.

- Se o seu administrador fornecer um link, escolha o link e passe para a etapa Step 5 do procedimento a seguir. O link direciona você para a página da Web apropriada e preenche o ID da conta (ou o alias) e o nome da função.
- Você pode criar manualmente o link e, em seguida, pular para a etapa Step 5 no procedimento a seguir. Para criar o link, use o seguinte formato:

  `https://signin.aws.amazon.com/switchrole?account=`*`account_id_number`*`&roleName=`*`role_name`*`&displayName=`*`text_to_display`*

  Se você substituir o seguinte texto:

  - *`account_id_number`*: o identificador de conta de 12 dígitos fornecido pelo seu administrador. Como alternativa, o administrador pode criar um alias de conta, para que o URL inclua o nome de sua conta, em vez de um ID de conta. Para obter mais informações, consulte Tipos de usuário, no Guia do usuário do Início de Sessão da AWS.
  - *`role_name`*: o nome da função que você deseja assumir. Você pode obter isso no final do ARN de função. Por exemplo, forneça o nome da função `TestRole` da seguinte função de ARN: `arn:aws:iam::123456789012:role/TestRole`.
  - (Opcional) *`text_to_display`*: o texto que você deseja que seja exibido na barra de navegação no lugar de seu nome de usuário quando esta função estiver ativa.
- Você pode alternar manualmente as funções utilizando as informações fornecidas pelo administrador, usando os procedimentos a seguir.

Por padrão, quando você alterna funções, a sessão do AWS Management Console dura uma hora. 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. Por exemplo, suponha que uma duração máxima de sessão de dez horas seja definida para uma função. Um usuário do IAM esteve conectado ao console por 8 horas quando decide alternar para a função. Há 4 horas restantes na sessão do usuário, portanto, a duração

permitida da sessão de perfil é de 4 horas. A tabela a seguir mostra como determinar a duração da sessão de um usuário do IAM ao alternar funções no console.

Duração da sessão da função do console dos usuários do IAM

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)		

> **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.

Para solucionar problemas comuns que você pode encontrar ao assumir uma função, consulte Não consigo assumir uma função.

Para mudar para uma função (console)

1. Faça login no AWS Management Console como um usuário do IAM 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***.
3. Selecione Switch Role (Mudar de função). Se esta for a primeira vez que esta opção é selecionada, uma página será exibida com mais informações. Depois de ler, escolha Switch Role (Mudar de função). Se você limpar seus cookies do navegador, esta página poderá ser exibida novamente.
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.

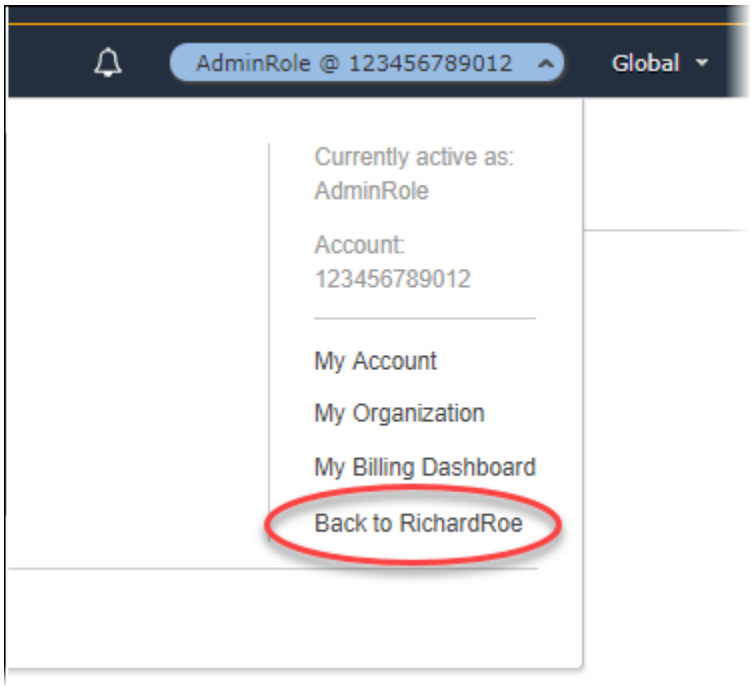
5. (Opcional) Escolha um Display name (Nome de exibição). Digite o texto que deseja que seja exibido na barra de navegação no lugar de seu nome de usuário quando esta função estiver ativa. Um nome é sugerido, baseado nas informações da conta e da função, mas você poderá alterá-lo para outro com significado para você. Você também pode selecionar 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 Switch Role (Mudar de função). 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.

> **Dica**
>
> As últimas funções que você usou aparecem no menu. Na próxima vez que você precisar mudar para uma dessas funções, bastará escolher a função desejada. Você só precisa digitar as informações da conta e da função manualmente se a função não for exibida no menu.

Para parar de usar uma função (console)

1. No console do IAM, escolha o Display Name (Nome de exibição) da sua função na barra de navegação no canto superior direito. Geralmente, ele é assim: ***rolename***@***account_ID_number_or_alias***.
2. Selecione Back to (Voltar para) ***username***. A função e suas permissões são desativadas e as permissões associadas ao seu usuário e grupos do IAM são restauradas automaticamente.

   Por exemplo, suponha que você tenha feito login à conta número `123456789012` usando o nome de usuário `RichardRoe`. Depois de usar a função `AdminRole`, você deseja interromper o uso da função e retornar para suas permissões originais. Para interromper o uso de uma função, escolha AdminRole @ 123456789012 e, depois, Back to RichardRoe (Voltar para RichardRoe).



Alternância para uma função 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 IAMe Criação de funções do IAM. Para saber mais sobre os diferentes métodos que você pode usar para assumir uma função, consulte Uso de funções do IAM.

> **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 uma função, perde temporariamente as permissões de usuário ou função anteriores e trabalha com as permissões atribuídas à função. 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 uma função para executar um comando da AWS CLI quando estiver conectado como um usuário autenticado externamente (SAML ou OIDC) que já esteja usando uma função. 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 uma função 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 a função. 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 Visualizar a configuração de duração máxima da sessão para uma função.

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 um usuário federado, ele poderá ser o par de chaves de acesso para a função atribuída inicialmente 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 `my-bucket-1` 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

```
{
    "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:::my-bucket-1/*",
                "arn:aws:s3:::my-bucket-1"
            ]
        },
        {
            "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 `my-bucket-2` 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

```
{
    "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:::my-bucket-2/*",
                "arn:aws:s3:::my-bucket-2"
            ]
        }
    ]
}
```

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

```
{
    "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 my-bucket-2 --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.

Alternância para uma função do IAM (Tools for 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 IAMe Criação de funções 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 tiver feito login como um usuário federado, elas poderão ser o par de chaves de acesso para a função atribuída inicialmente 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 AWS Tools for Windows 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.

Alternância para uma função do IAM (API da AWS)

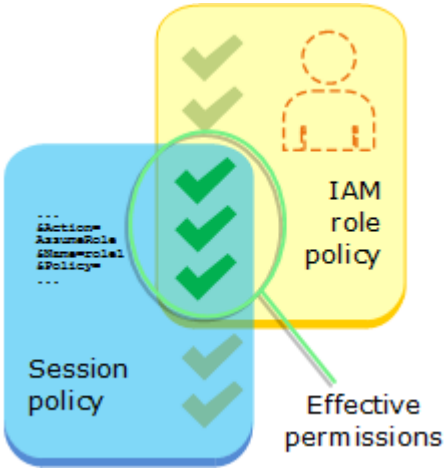
Uma função especifica um conjunto de permissões que você pode usar para acessar os recursos da AWS. Nesse sentido, ela é semelhante a um usuário do IAM. Um principal (pessoa ou aplicativo) assume uma função para receber permissões temporárias para executar as tarefas necessárias e interagir com recursos 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 Criação de funções do IAM. Para saber mais sobre os diferentes métodos que você pode usar para assumir uma função, consulte Uso de funções do IAM.

> **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 uma função, perde temporariamente as permissões de usuário ou função anteriores e trabalha com as permissões atribuídas à função. 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 temporária para uma função ou usuário federado. Você pode passar um único documento de política JSON de sessão em linha usando o parâmetro `Policy`. Você pode 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 estiver conectado como um usuário do IAM ou como um usuário autenticado externamente (SAML ou OIDC) que já esteja usando uma função. Você também pode usar o encadeamento de funções, que é o uso de uma função para assumir uma segunda função. Você não pode assumir uma função quando está conectado como o Usuário raiz da conta da AWS.

Por padrão, a sessão da função 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 a função. Para saber como visualizar o valor máximo para sua função, consulte Visualizar a configuração de duração máxima da sessão para uma função.

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 AWS IAM Identity Center.

- 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

AWS SDK for .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>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <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 group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
        return response.Group;
    }


    /// <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 group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        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>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</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> PutGroupPolicyAsync(string groupName, string
 policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <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 do AWS SDK for .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 WS 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 WS 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 WS 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 WS 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 WS 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
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência de comandos da AWS CLI.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

C++

SDK for C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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 do AWS SDK for 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.

```
// 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() {
 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()
 accessKey := scenario.CreateAccessKey(user)
 role := scenario.CreateRoleAndPolicies(user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(accessKey)
 scenario.ListBucketsWithAssumedRole(noPermsConfig, role)
 scenario.Cleanup(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() *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(userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(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(user *types.User)
 *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(*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(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(scenario.helper.GetName(), *user.Arn)
```

```
 if err != nil {panic(err)}
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  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(*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(*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(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(context.TODO(),
  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(context.TODO(), &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(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(context.TODO(),
 &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(context.TODO(),
  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(context.TODO(), &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(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(*role.RoleName)
   if err != nil {panic(err)}
   for _, policy := range policies {
    err = scenario.roleWrapper.DetachRolePolicy(*role.RoleName,
 *policy.PolicyArn)
    if err != nil {panic(err)}
    err = scenario.policyWrapper.DeletePolicy(*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(*role.RoleName)
   if err != nil {panic(err)}
   log.Printf("Deleted role %v.\n", *role.RoleName)

   userPols, err := scenario.userWrapper.ListUserPolicies(*user.UserName)
   if err != nil {panic(err)}
   for _, userPol := range userPols {
    err = scenario.userWrapper.DeleteUserPolicy(*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(*user.UserName)
   if err != nil {panic(err)}
   for _, key := range keys {
    err = scenario.userWrapper.DeleteAccessKey(*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(*user.UserName)
   if err != nil {panic(err)}
   log.Printf("Deleted user %v.\n", *user.UserName)
   log.Println(strings.Repeat("-", 88))
 }

}
```

Defina um struct que encapsule as ações de conta.

```
// 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() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &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() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &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.

```
// 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"`
}



// 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(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &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
}



// 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(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(context.TODO(),
 &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(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &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(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &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.

```
// 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(maxRoles int32) ([]types.Role, error) {
```

```
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &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(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(context.TODO(),
 &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(roleName string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &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(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &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(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &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(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &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(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &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(roleName string, policyArn string)
 error {
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &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(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &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(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &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.

```
// 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(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &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(userName string) (*types.User, error) {
```

```
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &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(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &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(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(context.TODO(),
 &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(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &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(userName string, policyName string)
 error {
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &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(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &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(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &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(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &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(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &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 do AWS SDK for 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 veja como configurar e executar no repositório de exemplos de código da AWS.

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 do 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 intervalos para a conta.

```
import {
  CreateUserCommand,
  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";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "test_name";
const policyName = "test_policy";
const roleName = "test_role";

export const main = async () => {
  // Create a user. The user has no permissions by default.
  const { User } = await iamClient.send(
    new CreateUserCommand({ UserName: userName }),
  );

  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: 60 }, () =>
  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 do AWS SDK for JavaScript.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole
  - DeleteUser
  - DeleteUserPolicy
  - DetachRolePolicy
  - PutUserPolicy

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

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: String = "{" +
        "  \"Version\": \"2012-10-17\"," +
        "  \"Statement\": [" +
        "    {" +
        "        \"Effect\": \"Allow\"," +
        "        \"Action\": [" +
        "            \"s3:*\"" +
        "       ]," +
        "       \"Resource\": \"*\"" +
        "    }" +
        "   ]" +
        "}"

    val request = CreatePolicyRequest {
        policyName = policyNameVal
        policyDocument = policyDocumentValue
    }

    IamClient { 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 { 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 {
        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 {
        credentialsProvider = staticCredentials
        region = "us-east-1"
    }

    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 { 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 { 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 do 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 do AWS SDK for 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 intervalos 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 do 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 intervalos 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 do AWS SDK for Ruby.

- AttachRolePolicy
- CreateAccessKey
- CreatePolicy
- CreateRole
- CreateUser
- DeleteAccessKey
- DeletePolicy
- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Rust

SDK for 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 do AWS SDK para Rust.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy

- DeleteRole
- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Uso de uma função 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 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.

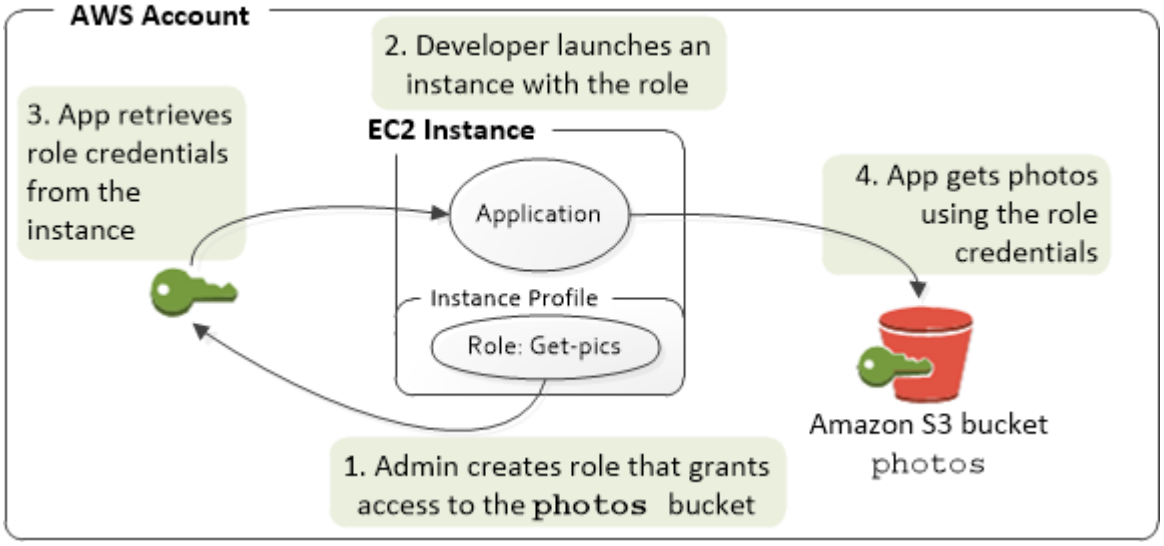
Usar funções dessa forma tem vários benefícios. Como as credenciais de perfil são temporárias e atualizadas automaticamente, não é necessário gerenciar credenciais, e você não precisa se preocupar com os riscos de segurança de longo prazo. Além disso, se usar uma única função para várias instâncias, 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?

Na figura a seguir, um desenvolvedor executa um aplicativo em uma instância do Amazon EC2 que requer acesso ao bucket do S3 denominado `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 `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 AWS Management Console 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

```
{
    "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.

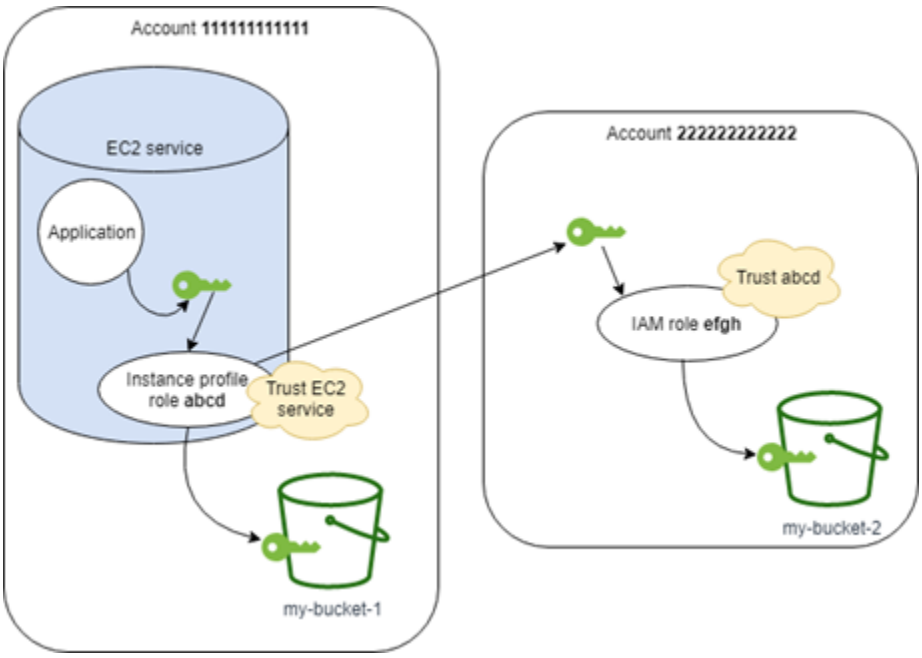
```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::account-id: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 `my-bucket-1` dentro da mesma conta `111111111111`. No entanto, a aplicação também deve ter permissão para assumir a função `efgh` entre contas para acessar o bucket `my-bucket-2` 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 `my-bucket-1` do Amazon S3:

Política de permissões da função ***abcd*** da conta 111111111111

```
{
    "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:::my-bucket-1/*",
                "arn:aws:s3:::my-bucket-1"
            ]
        },
        {
            "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

```
{
    "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 `my-bucket-2` 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

```
{
    "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:::my-bucket-2/*",
                "arn:aws:s3:::my-bucket-2"
            ]
        }
    ]
}
```

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

```
{
    "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 for 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 for .NET
  - Criar uma instância do Amazon EC2 com o SDK for Ruby no Guia do desenvolvedor do AWS SDK for 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 funções do IAM com instâncias do Amazon EC2, consulte a Guia do usuário do Amazon EC2 para instâncias do Linux.
- Para criar uma função, consulte Criação de funções 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 as credenciais de segurança temporárias das funções nos metadados da instância, consulte Recuperação de credenciais de segurança dos metadados da instância no Guia do usuário do Amazon EC2 para instâncias do Linux.

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 Funções do IAM para o Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux.

Gerenciar perfis de instância (console)

Se você usar o AWS Management Console 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 AWS Management Console 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, consulteExcluir funções ou perfis de instância.

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 EC2: `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 EC2: `DescribeIamInstanceProfileAssociations`
- Desanexar um perfil de instância com uma função de uma instância do EC2 em execução ou interrompida: `DisassociateIamInstanceProfile`

Revogação das credenciais de segurança temporárias da função 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 AWS Management Console 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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Roles (Funções) 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. A AWS pede que você confirme a ação. Marque a caixa de seleção I acknowledge that I am revoking all active sessions for this role. (Confirmo que estou revogando todas as sessões ativas para essa função) e escolha Revoke active sessions (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.

```
{
  "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
```

Gerenciamento de funções do IAM

Ocasionalmente, você precisa modificar ou excluir as funções que você criou. Para alterar uma função, você pode fazer o seguinte:

- Modificar as políticas associadas à função
- Alterar quem pode acessar a função
- Editar as permissões que a função concede aos usuários
- Altere a configuração de duração máxima da sessão para funções que são assumidas usando o AWS Management Console, a AWS CLI ou a API

Você também pode excluir funções que não são mais necessárias. Você pode gerenciar suas funções a partir do AWS Management Console, do AWS CLI e da API.

Tópicos

- Modificar uma função

- Excluir funções ou perfis de instância

Modificar uma função

É possível usar o AWS Management Console, a AWS CLI ou a API do IAM para fazer alterações em uma função.

Tópicos

- Visualizar acesso à função
- Gerar uma política com base em informações de acesso
- Modificar uma função (console)
- Modificar uma função (AWS CLI)
- Modificar uma função (API da AWS)

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 as 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 Gerar políticas com base na atividade de acesso.

Modificar uma função (console)

Você pode usar a AWS Management Console para modificar uma função. Para alterar o conjunto de tags em uma função, consulte Gerenciamento de etiquetas em funções do IAM (console).

Tópicos

Modificação de uma política de confiança de função (console)

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.

Como modificar a política de confiança de uma função (console)

1. Faça login em AWS Management Console 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 Trust relationships (Relacionamentos de confiança) e, em seguida, escolha Edit trust policy (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 Concessão de permissões a um usuário para alternar funções.

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 IAM e Referência de elementos 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 Concessão de permissões a um usuário para alternar funções.

1. Faz login na Conta da AWS externa confiável.
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.

Modificar a política de permissões de uma função (console)

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 Serviços da AWS 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 a esse serviço.

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 gerenciada pela AWS. A política gerenciada pela AWS é exibida 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).

Modificar a descrição de uma função (console)

Para alterar a descrição da função, modifique o texto da descrição.

Para alterar a descrição de uma função (console)

1. Faça login em AWS Management Console 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).

Modificar a duração máxima da sessão de uma função (console)

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.

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 em AWS Management Console 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 Revogação das credenciais de segurança temporárias da função do IAM.

No AWS Management Console, 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 Alternância para uma função do IAM (AWS CLI).
- Para especificar a duração de uma sessão usando a API da AWS, use o parâmetro `DurationSeconds`. Para saber mais, consulte Alternância para uma função do IAM (API da AWS).

Modificar o limite de permissões de uma função (console)

Para alterar o número máximo de permissões permitidas para uma função, modifique o limite de permissões da função.

Para alterar a política usada para definir o limite de permissões para uma função

1. Faça login no AWS Management Console 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.

Modificar uma função (AWS CLI)

Você pode usar a AWS Command Line Interface para modificar uma função. Para alterar o conjunto de tags em uma função, consulte Gerenciar etiquetas em funções do IAM (AWS CLI ou API da AWS).

Tópicos

- Modificar uma política de confiança de função (AWS CLI)
- Modificar uma política de permissões de função (AWS CLI)
- Modificar a descrição de uma função (AWS CLI)
- Modificar a duração máxima da sessão de uma função (AWS CLI)
- Modificar o limite de permissões de uma função (AWS CLI)

Modificar uma política de confiança de função (AWS CLI)

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. 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 Modificação de uma política de confiança de função (console).

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.

```
{
    "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 Concessão de permissões a um usuário para alternar funções.

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 Concessão de permissões a um usuário para alternar funções.

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:

```
{
```

```
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE: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

Modificar uma política de permissões de função (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 Serviços da AWS 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 da função vinculada a 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

Modificar a descrição de uma função (AWS CLI)

Para alterar a descrição da função, modifique o texto da descrição.

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

Modificar a duração máxima da sessão de uma função (AWS CLI)

Para especificar a configuração de duração máxima da sessão para funções que são assumidas usando a AWS CLI ou a API, 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.

> **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 Revogação das credenciais de segurança temporárias da função do IAM.

Modificar o limite de permissões de uma função (AWS CLI)

Para alterar o número máximo de permissões permitidas para uma função, modifique o limite de permissões da função.

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.

Modificar uma função (API da AWS)

Você pode usar a API da AWS para modificar uma função. Para alterar o conjunto de tags em uma função, consulte Gerenciar etiquetas em funções do IAM (AWS CLI ou API da AWS).

Tópicos

- Modificar a política de confiança de uma função (API da AWS)
- Modificar a política de permissões de uma função (API da AWS)
- Modificar a descrição de uma função (API da AWS)
- Modificar a duração máxima da sessão de uma função (API da AWS)
- Modificar o limite de permissões de uma função (API da AWS)

Modificar a política de confiança de uma função (API da AWS)

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. 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 Modificação de uma política de confiança de função (console).

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.

   ```
   {
   ```

```
    "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 Concessão de permissões a um usuário para alternar funções.

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 Concessão de permissões a um usuário para alternar funções.

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:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::ACCOUNT-ID-THAT-CONTAINS-ROLE: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

Modificar a política de permissões de uma função (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 Serviços da AWS 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 da função vinculada a 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

Modificar a descrição de uma função (API da AWS)

Para alterar a descrição da função, modifique o texto da descrição.

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

Modificar a duração máxima da sessão de uma função (API da AWS)

Para especificar a configuração de duração máxima da sessão para funções que são assumidas usando a AWS CLI ou a API, 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.

> **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-sessionduration` 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 Revogação das credenciais de segurança temporárias da função do IAM.

Modificar o limite de permissões de uma função (API da AWS)

Para alterar o número máximo de permissões permitidas para uma função, modifique o limite de permissões da função.

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.

Excluir funções ou perfis de instância

Se você não precisar mais de uma função, recomendamos excluir a função e suas permissões associadas. Dessa forma, você não tem uma entidade não utilizada que não seja monitorada ativamente 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 Revogação das credenciais de segurança temporárias da função do IAM.

Tópicos

- Visualizar acesso à função
- Excluir uma função vinculada 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 à função

Antes de excluir uma função, recomendamos que você revise quando a função foi usada pela última vez. Você pode fazer isso usando o AWS Management Console, 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 da última atividade da função pode não corresponder à última data relatada na guia Access Advisor (Consultor de acesso). A guia Access Advisor (Consultor de acesso) relata a atividade somente para serviços permitidos pelas políticas de permissões da função. A data da última atividade da função inclui a última tentativa de acessar qualquer produto na AWS.

> **Note**
>
> O período de rastreamento da última atividade de uma função e os dados do Consultor de acesso são 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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Roles.
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 uma função vinculada ao serviço

Se a função for uma função vinculada ao serviço, consulte a documentação do serviço vinculado para saber como excluir a função. 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. Para obter mais informações, consulte Excluir uma função vinculada ao serviço.

Exclusão de uma função do IAM (console)

Ao usar o AWS Management Console 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 AWS Management Console 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 AWS Management Console 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 Usar funções vinculadas ao serviço.

Provedores de identidade e federação

Se você já gerencia identidades de usuários fora da AWS, poderá usar provedores de identidades em vez de criar usuários do IAM na sua Conta da AWS. 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.

Um IdP externo fornece informações de identidade à AWS usando OIDC (OpenID Connect) ou 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.

> **Note**
>
> Por segurança, recomendamos que você gerencie o acesso de usuários humanos no IAM Identity Center com um provedor de identidades SAML externo em vez de usar federação SAML no 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).

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.

Este guia aborda a federação do IAM. Seu caso de uso pode ser mais bem atendido pelo IAM Identity Center ou pelo 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.

	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 AWS Management Console 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 usuários federados 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 usuário federados 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 back-end. 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 obter 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.

Cenários comuns

> **Note**
>
> Recomendamos exigir que seus usuários humanos usem credenciais temporárias ao acessar a AWS. Você já pensou em usar o AWS IAM Identity Center? 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 AWS IAM Identity Center.

É possível usar um provedor de identidades (IdP) externo para gerenciar identidades de usuários fora da AWS. 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

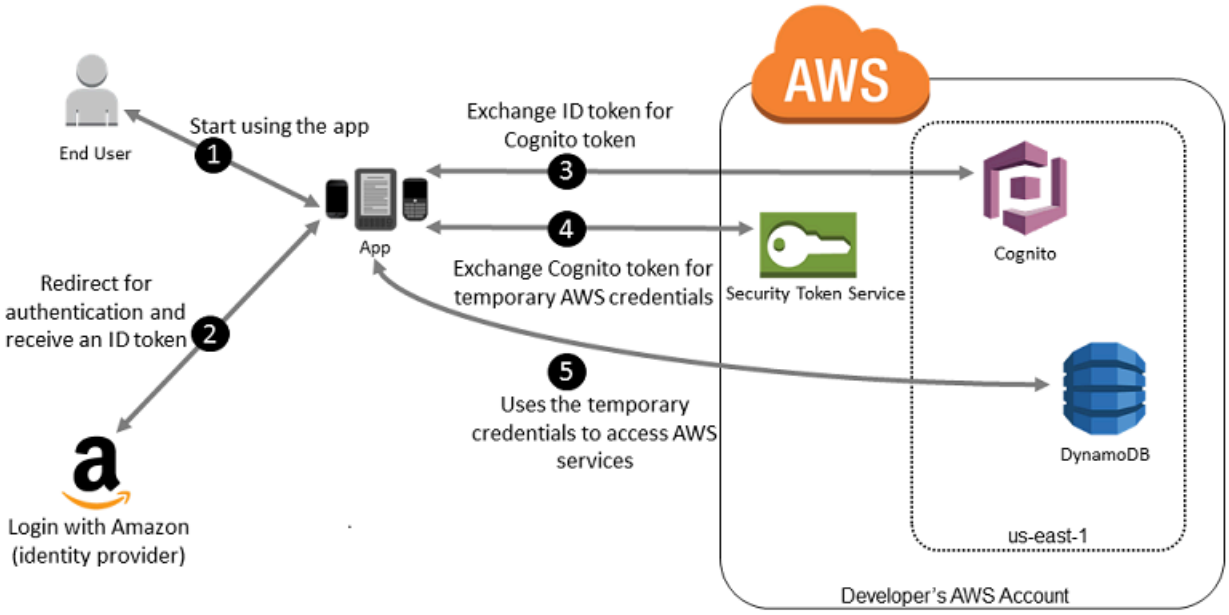
Usar o Amazon Cognito para aplicativos móveis

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 constrói o jogo para que os usuários possam fazer login usando uma identidade que eles já estabeleceram com um provedor de identidade (IdP) externo 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 AWS Management Console. 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 AWS Management Console 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 obter mais informações, consulte:

- Faça login (Android) na documentação da estrutura do AWS Amplify.
- Faça login (iOS) na documentação da estrutura do AWS Amplify.

Usar operações de API de federação OIDC para aplicações 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 noAWS Management Console. 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 uma função para um provedor de identidade de terceiros (federação).

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 distinguir 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 Identificar usuários com a federação OIDC mais adiante neste tópico.

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.

Para a maioria dos cenários, recomendamos que você use o Amazon Cognito porque ele atua como um agente de identidades e faz grande parte do trabalho de federação para você. Para obter detalhes, consulte a seção a seguir, Usar o 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

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 do IAM OIDC usando oAWS Management Console, 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, uma função é atribuída dinamicamente a um usuário federado que é autenticado 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 à função determinam o que os usuários federados podem fazer na AWS. Para criar uma função para um provedor de identidade de terceiros, consulte Criar uma função para um provedor de identidade de terceiros (federação).

> **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.

Tópicos

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.
- 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 usuários federados. 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

> **Note**
>
> Você pode incluir declerações adicionais, como custom 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",
    "custom"
  ],
  "response_types_supported": [
    "id_token"
  ],
  "id_token_signing_alg_values_supported": [
    "RS256"
  ],
  "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 AWS Management Console.

> **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**
>
> A AWS protege a comunicação com alguns provedores de identidade (IdPs) OIDC por meio de nossa biblioteca de autoridades de certificação (CAs) raiz confiáveis, em vez de usar uma impressão digital do certificado para verificar o certificado do servidor IdP. Nesses casos, sua impressão digital herdada permanece em sua configuração, mas não é mais usada para validação. Esses IdPs do OIDC incluem Auth0, GitHub, GitLab, Google e aqueles que usam um bucket do Amazon S3 para hospedar um endpoint do JSON Web Key Set (JWKS).

2. Abra o console 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.

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 identidade 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 uma função para um provedor de identidade de terceiros (federação).

   > **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**
>
> A AWS protege a comunicação com alguns provedores de identidade (IdPs) OIDC por meio de nossa biblioteca de autoridades de certificação (CAs) raiz confiáveis, em vez de usar uma

impressão digital do certificado para verificar o certificado do servidor IdP. Nesses casos, sua impressão digital herdada permanece em sua configuração, mas não é mais usada para validação. Esses IdPs do OIDC incluem Auth0, GitHub, GitLab, Google e aqueles que usam um bucket do Amazon S3 para hospedar um endpoint do JSON Web Key Set (JWKS).

1. Abra o console 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. Na seção Thumbprints (Impressões digitais), escolha Manage (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 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 AWS Certificate Manager. 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.

Você pode 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**
>
> A AWS protege a comunicação com alguns provedores de identidade (IdPs) OIDC por meio de nossa biblioteca de autoridades de certificação (CAs) raiz confiáveis, em vez de usar uma impressão digital do certificado para verificar o certificado do servidor IdP. Nesses casos, sua impressão digital herdada permanece em sua configuração, mas não é mais usada para validação. Esses IdPs do OIDC incluem Auth0, GitHub, GitLab, Google e aqueles que usam um bucket do Amazon S3 para hospedar um endpoint do JSON Web Key Set (JWKS).

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. Você pode 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 da 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",
   ...
   ```
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, 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.

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
```

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. Você pode 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. Você pode 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. Você pode 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.

Identificar usuários com a federação OIDC

Quando você cria políticas de acesso no IAM, geralmente é útil poder especificar permissões com base em aplicações configuradas e no ID de usuários que se autenticaram usando um provedor de identidade (IdP) externo. Por exemplo, sua aplicação móvel que usa a federação OIDC pode manter as informações no Amazon S3 usando uma estrutura como esta:

```
myBucket/app1/user1
myBucket/app1/user2
myBucket/app1/user3
...
myBucket/app2/user1
myBucket/app2/user2
myBucket/app2/user3
...
```

Além disso, você pode distinguir esses caminhos por provedor. Neste caso, a estrutura pode se parecer com a seguinte (apenas dois provedores são listados para economizar espaço):

```
myBucket/Amazon/app1/user1
myBucket/Amazon/app1/user2
myBucket/Amazon/app1/user3
...
myBucket/Amazon/app2/user1
myBucket/Amazon/app2/user2
myBucket/Amazon/app2/user3

myBucket/Facebook/app1/user1
myBucket/Facebook/app1/user2
myBucket/Facebook/app1/user3
...
myBucket/Facebook/app2/user1
myBucket/Facebook/app2/user2
myBucket/Facebook/app2/user3
...
```

Para essas estruturas, `app1` e `app2` representam aplicativos diferentes, como jogos diferentes e cada usuário do aplicativo tem uma pasta diferente. Os valores para `app1` e `app2` podem ser nomes acessíveis que você atribui (por exemplo, `mynumbersgame`) ou podem ser os IDs de aplicativos que os provedores atribuem quando você configura o aplicativo. Se você incluir nomes de provedor no caminho, eles também poderão ser nomes acessíveis, como `Cognito`, `Amazon`, `Facebooke Google`.

Normalmente, você pode criar as pastas para `app1` e `app2` por meio do AWS Management Console, já que os nomes de aplicativos são valores estáticos. Isso também se aplica se você incluir o nome do provedor no caminho, já que o nome do provedor também é um valor estático. Em contraste, as pastas específicas do usuário (*`user1`*, *`user2`*, *`user3`* etc.) devem ser criadas em tempo de execução do aplicativo, usando o ID do usuário disponível no valor `SubjectFromWebIdentityToken` retornado pela solicitação para `AssumeRoleWithWebIdentity`.

Para gravar políticas que permitam acesso exclusivo aos recursos para usuários individuais, você pode combinar o nome da pasta, incluindo o nome do aplicativo e o nome do provedor, se estiver usando-o. Inclua as seguintes chaves de contexto específicas do provedor que fazem referência ao ID do usuário que o provedor retorna:

- `cognito-identity.amazonaws.com:sub`
- `www.amazon.com:user_id`
- `graph.facebook.com:id`
- `accounts.google.com:sub`

Para provedores OIDC, use a URL totalmente qualificada do provedor OIDC com a chave de subcontexto, como o seguinte exemplo:

- *`server.example.com`*`:sub`

O exemplo a seguir mostra uma política de permissões que concede acesso a um bucket no Amazon S3 somente se o prefixo para o bucket corresponder à string:

`myBucket/Amazon/mynumbersgame/`*`user1`*

O exemplo supõe que o usuário faz login usando o Login with Amazon e que o usuário usa uma aplicação chamada `mynumbersgame`. O ID exclusivo do usuário é apresentado como um atributo chamado `user_id`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::myBucket"],
      "Condition": {"StringLike": {"s3:prefix": ["Amazon/mynumbersgame/
${www.amazon.com:user_id}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": [
        "arn:aws:s3:::myBucket/amazon/mynumbersgame/${www.amazon.com:user_id}",
        "arn:aws:s3:::myBucket/amazon/mynumbersgame/${www.amazon.com:user_id}/*"
      ]
    }
  ]
}
```

Você criaria políticas semelhantes para usuários que fazem login usando o Amazon Cognito, Facebook, Google ou outro IdP compatível com OpenID Connect. Essas políticas usariam um nome de provedor diferente como parte do caminho, bem como IDs de aplicativo diferentes.

Para obter mais informações sobre chaves de federação OIDC disponíveis para verificação de condições nas políticas, consulte Chaves disponíveis para federação OIDC da AWS.

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.

- Automatizar perfis de identidade Web do AWS IAM baseadas no OpenID Connect com o Microsoft Entra ID no blog da AWS Partner Network (APN) explica como autenticar processos automatizados em segundo plano ou aplicações executadas fora da AWS usando autorização OIDC máquina a máquina.
- O artigo Federação de identidades da Web com aplicações móveis discute a federação OIDC e mostra um exemplo de como usar a federação OIDC para obter acesso ao conteúdo no Amazon S3.

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 AWS Management Console 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.

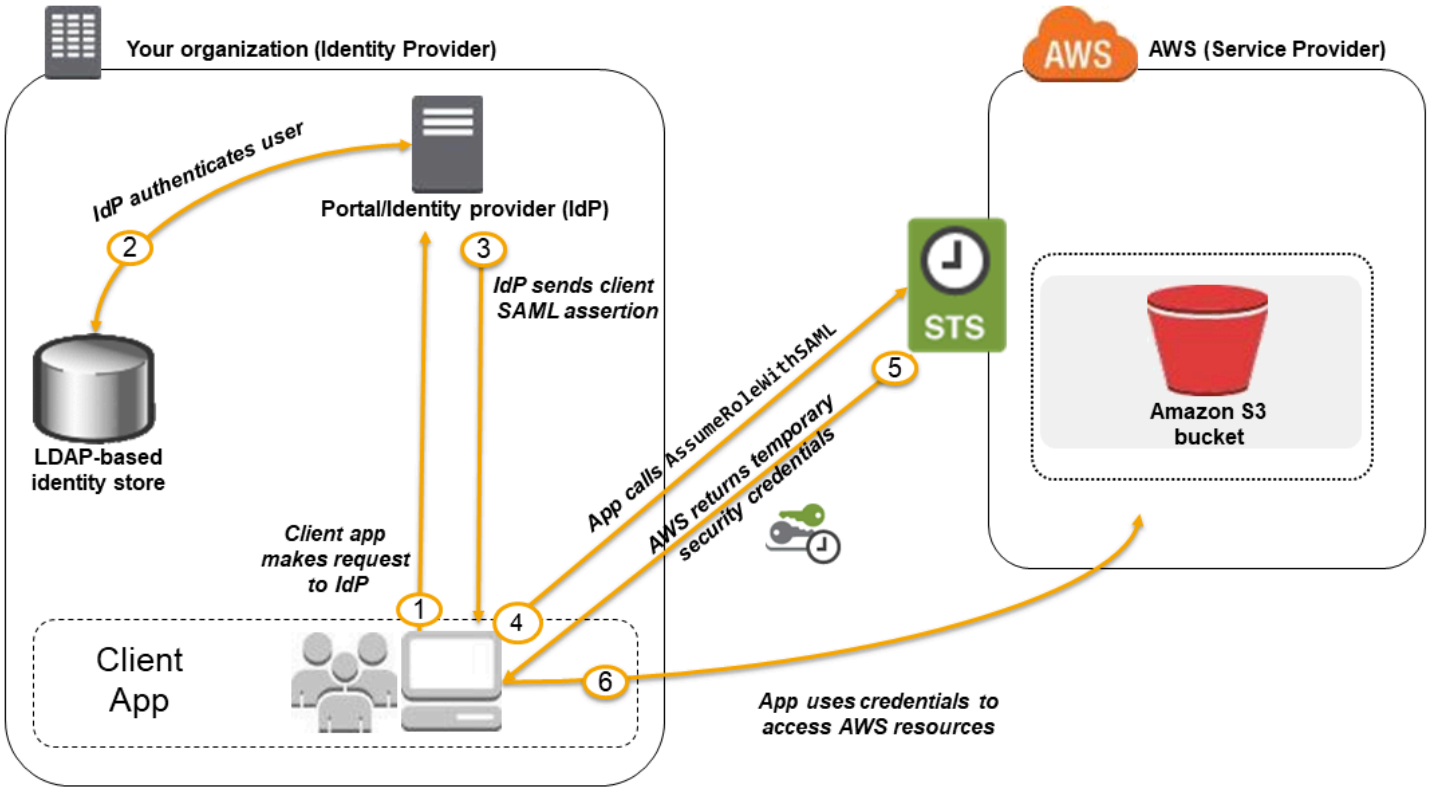
A federação do IAM é compatível com estes casos de uso:

- Acesso federado para permitir que um usuário ou aplicativo na sua organização chame operações de API da AWS. 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 Solicitação de 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. Esse é o cenário discutido nesse tópico.
- Logon único (SSO) baseado na web para o AWS Management Console 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 agente de identidades personalizado ao console da AWS.

Tópicos

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.
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.
5. A resposta da API para o aplicativo do cliente inclui credenciais de segurança temporárias.
6. 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*, consulte a coluna Região em Endpoints de login da AWS.

   Ou você também pode usar o documento de metadados SAML em `https://signin.aws.amazon.com/static/saml-metadata.xml` .
2. Usando o IdP da sua organização, você gera um arquivo XML de metadados equivalente que pode descrever seu 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.
3. No console do IAM, crie um provedor de identidade SAML. Como parte do processo, carregue o documento de metadados SAML que foi gerada pelo IdP na sua organização em Step 2. Para ter 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 ter mais informações, consulte Criar uma função para um provedor de identidade de terceiros (federação).

> **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 ter 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 ter mais informações, consulte Habilitar o acesso de usuários federados SAML 2.0 ao AWS Management Console.

6. No aplicativo que você estiver criando, chame a API `AssumeRoleWithSAML` do AWS Security Token Service, 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

A função ou as funções que você cria no IAM definem o que os usuários federados 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:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {
      "StringEquals": {
        "saml:aud": "https://signin.aws.amazon.com/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.

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.

Você pode incluir 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 login da).

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.

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:

```
myBucket/app1/user1
myBucket/app1/user2
myBucket/app1/user3
```

Você pode criar o bucket (`myBucket`) 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 do `NameQualifier` e do `Subject` pode ser usada para identificar exclusivamente um usuário federado. 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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "s3:GetObject",
      "s3:PutObject",
      "s3:DeleteObject"
    ],
    "Resource": [
      "arn:aws:s3:::exampleorgBucket/backup/${saml:namequalifier}/${saml:sub}",
      "arn:aws:s3:::exampleorgBucket/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 os usuários em sua organização 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 AWS Management Console 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, uma função é atribuída dinamicamente a um usuário federado que é autenticado 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 à função determinam o que os usuários federados podem fazer na AWS. Para criar uma função para a federação do SAML, consulte Criar uma função para um provedor de identidade de terceiros (federação).

Por fim, depois de criar a função, você conclui a confiança de SAML configurando o IdP com informações sobre a AWS e as funções que seus usuários federados 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)
- 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)

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” (Não foi possível analisar metadados).
> Conforme definido pelo Perfil de Interoperabilidade de Metadados SAML V2.0 Versão 1.0, o IAM não avalia nem toma medidas em relação à expiração do certificado X.509 do documento de metadados.

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 AWS Management Console 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 AWS Management Console 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”.
6. (Opcional) Para adicionar tags, você pode adicionar pares de chave-valor a fim de ajudá-lo a identificar e organizar 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 Etiquetamento de provedores de identidade SAML do IAM.

   Escolha Adicionar Tag. Insira valores para cada par de chave-valor de tag.
7. Verifique as informações fornecidas. Quando terminar, escolha Add provider (Adicionar provedor).
8. 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 uma função para um provedor de identidade de terceiros (federação).

   > **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 AWS Management Console 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.

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)

- 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)

- 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`

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 `https://`*`region-code`*`.signin.aws.amazon.com/static/saml-metadata.xml` ou `https://signin.aws.amazon.com/static/saml-metadata.xml`. Para obter uma lista dos possíveis valores de *`region-code`*, consulte a coluna Região em Endpoints de login da AWS.

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 usuários federados. Para obter mais informações, consulte os tópicos a seguir.

- 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 usuários federados 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 AWS IAM Identity Center? 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 AWS IAM Identity Center.

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.

> **Tip**
>
> Os engenheiros do AWS 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 AWS Management Console e inclui um exemplo de JavaScript. 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.
ForgeRock	A Plataforma de identidade ForgeRock integra-se à AWS. É possível configurar a ForgeRock para transmitir tags de

Solução	Mais informações
	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.
Microsoft Active Directory Federation Services (AD FS)	Notas de campo: integração do Active Directory Federation Service com o AWS IAM Identity Center: esta publicação no blog AWS Architecture explica o fluxo de autenticação entre o AD FS e o AWS IAM Identity Center (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.

Solução	Mais informações
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 Identidad e do IAM para a AWS.
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.

Solução	Mais informações
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.
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 AWS Management Console: este registro no AWS Security Blog fornece um tutorial detalhado sobre como instalar o Shibboleth e configurá-lo como provedor de identidade 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 endpoint SAML da AWS em `https://region-code.signin.aws.amazon.com/saml`. Para obter uma lista das possíveis substituições de *`region-code`*, consulte a coluna Região em Endpoints de login 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 `NameID`

O trecho a seguir mostra um exemplo. Substitua os valores marcados pelos seus próprios valores. Deve haver exatamente um elemento `SubjectConfirmation` com um elemento `SubjectConfirmationData` que inclui ambos os atributos `NotOnOrAfter` e `Recipient`. Esses atributos incluem um valor que deve corresponder ao endpoint `https://`*`region-code`*`.signin.aws.amazon.com/saml` da AWS. Para obter uma lista dos possíveis valores de *`region-code`*, consulte a coluna Região em Endpoints de login da AWS. Para o valor AWS, você também pode usar `https://signin.aws.amazon.com/static/saml`, conforme mostrado no exemplo a seguir.

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`

```
<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://signin.aws.amazon.com/saml"/>
  </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`.

Atributo SAML **`PrincipalTag`**

(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 `CostCenter` e `Principal` 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>
```

Atributo SAML **`Role`**

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 AWS Management Console.

> **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>
```

Atributo SAML **`RoleSessionName`**

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 AWS Management Console. 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 (`johndoe`) 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>
```

Atributo SAML **`SessionDuration`**

(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 AWS Management Console 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 AWS Management Console 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*, consulte a coluna Região em Endpoints de login da AWS. Ou você pode usar a seguinte URL: `https://signin.aws.amazon.com/static/saml`. Observe que esse atributo estende sessões apenas para o AWS Management Console. 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 for definido, o menor valor dos 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 ter mais informações, consulte Revogação das credenciais de segurança temporárias da função 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>
```

Atributo SAML **`SourceIdentity`**

(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 64 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

(`johndoe`) 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 `DiegoRamirez` use o atributo a seguir.

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/SourceIdentity">
  <AttributeValue>DiegoRamirez</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. Você pode usar essas chaves para controlar o acesso a uma função. 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 política`ForAllValues` 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.

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

Atributo eduPerson ou eduOrg (chave **Name**)	É mapeado para essa chave de contexto da AWS (chave **FriendlyName**)	Tipo
urn:oid:1.3.6.1.4.1.5923.1.1.1.9	eduPersonScopedAffiliation	Lista de strings
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

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

Atributo do AD	É mapeado para essa chave de contexto da AWS	Tipo
`http://schemas.xmlsoap.org/claims/CommonName`	`commonName`	String
`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

Atributos 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 de usuários federados SAML 2.0 ao AWS Management Console

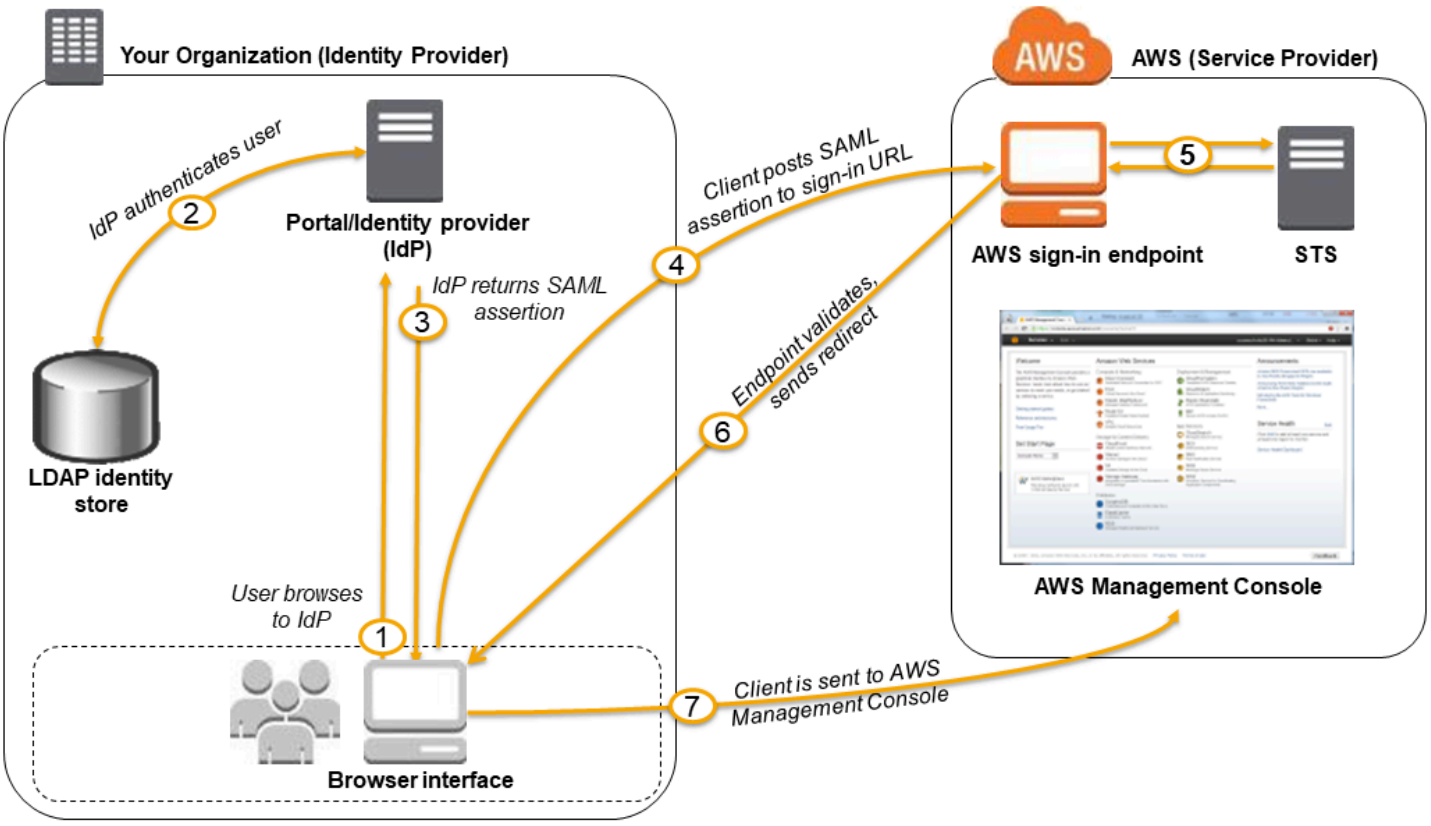
Você pode usar uma função para configurar o provedor de identidades (IdP) compatível com SAML 2.0 e o AWS para permitir que os usuários federados acessem o AWS Management Console. A função concede ao usuário permissões para executar tarefas no console. Se você desejar oferecer aos usuários federados SAML outras maneiras de acessar a AWS, consulte um dos seguintes tópicos:

- AWS CLI: Alternância para uma função do IAM (AWS CLI)
- Tools for Windows PowerShell: Alternância para uma função do IAM (Tools for Windows PowerShell)
- API da AWS: Alternância para uma função 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 AWS Management Console 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 AWS Management Console.



O diagrama ilustra as seguintes etapas:

1. O usuário acessa o portal de sua organização e seleciona a opção de ir para o AWS Management Console. 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. A AWS envia o URL de login de volta para o cliente como um redirecionamento.

7. O navegador cliente é redirecionado para o AWS Management Console. 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 AWS Management Console, 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 AWS Management Console 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 AWS Management Console 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 mais detalhes, consulte Criar um provedor de identidades SAML no IAM.

Configurar permissões na AWS para seus usuários federados

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 AWS Management Console. A política de confiança da função pode ter a seguinte aparência:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {"Federated": "arn:aws:iam::account-id:saml-provider/
ExampleOrgSSOProvider"},
    "Action": "sts:AssumeRoleWithSAML",
    "Condition": {"StringEquals": {
      "saml:edupersonorgdn": "ExampleOrg",
      "saml:aud": "https://signin.aws.amazon.com/saml"
    }}
  }]
}
```

> **Note**
>
> Os IdPs do SAML usados em uma política de confiança de função devem estar na mesma conta em que a função está.

Você pode incluir 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 login da).

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`. Para obter uma lista dos possíveis valores de *region-code*, consulte a coluna Região em Endpoints de login 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 depois o fornecem 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.

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.
- Você pode 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. Você pode 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 ter 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 datacenter ou em um terceiro externo na Web. Para obter mais informações sobre provedores de identidade externos, consulte Provedores de identidade e federação.

- 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 ter 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 agente 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.
- Federação OpenID Connect (OIDC): é possível permitir que os usuários iniciem sessão usando um provedor de identidade de terceiros conhecido, como Login with Amazon, Facebook, Google ou qualquer provedor compatível com o OIDC 2.0 para sua aplicação móvel ou Web. Não é necessário criar um código de início de sessão personalizado nem gerenciar suas próprias identidades de usuário. 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 ter mais informações, consulte Federação OIDC.

  A federação OIDC do AWS STS 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.

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 Criação de uma função para delegar 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 ter mais informações, consulte Uso de uma função do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2.

Outros produtos 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 Serviços da AWS que funcionam com o IAM.

Solicitação de 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 Uso de funções do IAM.

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.

AssumeRole: delegação e federação entre contas por meio de um agente de identidades personalizado

A operação da API `AssumeRole` é útil para permitir que os usuários existentes do IAM acessem 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 Comparação das operações de API do AWS STS. Para obter mais informações, consulte Criação de uma função para delegar permissões a um usuário do IAM e Configuração de acesso à API protegido por MFA.

Essa chamada deve ser feita usando credenciais de segurança da AWS válidas. Ao fazer essa chamada, passe as seguintes informações:

- O nome de recurso da Amazon (ARN) da função que o aplicativo deve assumir.
- A duração (opcional), que especifica a duração das credenciais de segurança temporárias. 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 Visualizar a configuração de duração máxima da sessão para uma função. Se você não passar esse parâmetro, as credenciais temporárias vão expirar em uma hora. O parâmetro `DurationSeconds` desta API é diferente do parâmetro HTTP `SessionDuration` que é usado para especificar a duração de uma sessão do console. Use o parâmetro HTTP `SessionDuration` na solicitação para o endpoint de federação para obter um token de login do console. Para ter mais informações, consulte Habilitar o acesso do agente de identidades personalizado ao console da AWS.
- Nome da sessão da função. Use esse valor de string para identificar a sessão quando uma função for usada por diferentes entidades de segurança. Para fins de segurança, os administradores podem visualizar este campo nos logs do AWS CloudTrail para ajudar a identificar quem executou uma ação na AWS. O administrador pode exigir que você especifique seu nome de usuário do IAM como o nome da sessão quando você assumir a função. Para ter mais informações, consulte `sts:RoleSessionName`.
- (Opcional) Identidade-fonte. Você pode exigir que os usuários especifiquem uma identidade-fonte quando assumirem uma função. Depois que a identidade-fonte é definida, o valor não pode ser alterado. Ele estará presente na solicitação para todas as ações realizadas durante a sessão de função. O valor da identidade-fonte persiste nas sessões de função encadeada. Você pode usar informações de identidade-fonte em logs do AWS CloudTrail para determinar quem executou

ações com uma função. Para obter mais informações sobre como usar a identidade-fonte, consulte Monitorar e controlar ações realizadas com funções assumidas.

- (Opcional) Políticas de sessão em linha ou gerenciadas. Estas políticas limitam as permissões da política baseada em identidade da função que são atribuídas à sessão da função. 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.
- Tags de sessão (opcionais). Você pode assumir uma função e usar as credenciais temporárias para fazer uma solicitação. Ao fazer isso, as tags principais da sessão incluirão as tags da função e as tags de sessão passadas. Se você fizer essa chamada usando credenciais temporárias, a nova sessão também herdará tags de sessão transitiva da sessão de chamada. Para obter mais informações sobre tags de sessão, consulte Passar tags de sessão no AWS STS.
- Informações de MFA (opcionais). Se configurada para usar a autenticação multifator (MFA), inclua o identificador para um dispositivo MFA e o código único fornecido pelo dispositivo.
- Um valor `ExternalId` (opcional) que pode ser usado para delegar o acesso à sua conta a terceiros. Esse valor ajuda a garantir que somente o terceiro especificado pode acessar a função. Para ter mais informações, consulte Como usar um ID externo ao conceder acesso aos seus recursos da AWS a terceiros.

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:

```
{"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>8</PackedPolicySize>
</AssumeRoleResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</AssumeRoleResponse>
```

> **Note**
>
> Uma conversão da AWS compacta as políticas de sessão e as tags de sessão passadas em um formato binário compactado que têm um limite separado. Sua solicitação pode falhar para esse limite mesmo que seu texto simples atenda aos outros requisitos. O elemento de resposta `PackedPolicySize` indica, em porcentagem, o quão perto as políticas e tags da sua solicitação estão do limite de tamanho superior.

AssumeRoleWithWebIdentity: federação por meio de um provedor de identidades baseado na Web

A operação da API `AssumeRoleWithWebIdentity` retorna um conjunto de credenciais de segurança temporárias para usuários federados que são autenticados por meio de um provedor de identidades público. Exemplos de provedores de identidades públicos incluem Login with Amazon, Facebook, Google e qualquer provedor de identidades compatível com OpenID Connect (OIDC). Essa operação é útil para a criação de aplicativos móveis ou aplicativos web baseados no cliente que exijam acesso à AWS. O uso desta operação significa que seus usuários não precisam de suas próprias identidades do AWS ou do IAM. Para ter mais informações, consulte Federação OIDC.

Em vez de chamar diretamente `AssumeRoleWithWebIdentity`, recomendamos que você use o Amazon Cognito e o provedor de credenciais do Amazon Cognito com os AWS SDKs para desenvolvimento móvel. Para obter mais informações, consulte Autenticação com o Amplify na documentação do Amplify.

Se você não estiver usando o Amazon Cognito, chame a ação `AssumeRoleWithWebIdentity` do AWS STS. Esta é uma chamada sem assinatura, o que significa que o aplicativo não precisa ter

acesso a nenhuma credencial de segurança da AWS para fazer a chamada. Ao fazer essa chamada, passe as seguintes informações:

- O nome de recurso da Amazon (ARN) da função que o aplicativo deve assumir. Se o seu aplicativo dá suporte a várias maneiras para os usuários fazerem login, você deve definir várias funções, uma por provedor de identidade. A chamada para `AssumeRoleWithWebIdentity` deve incluir o ARN da função que é específico para o provedor pelo qual o usuário fez login.
- O token que o aplicativo obtém do IdP (provedor de identidade) depois que o usuário é autenticado.
- Você pode configurar seu IdP para passar atributos em seu token como tags de sessão.
- A duração (opcional), que especifica a duração das credenciais de segurança temporárias. 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 Visualizar a configuração de duração máxima da sessão para uma função. Se você não passar esse parâmetro, as credenciais temporárias vão expirar em uma hora. O parâmetro `DurationSeconds` desta API é diferente do parâmetro HTTP `SessionDuration` que é usado para especificar a duração de uma sessão do console. Use o parâmetro HTTP `SessionDuration` na solicitação para o endpoint de federação para obter um token de login do console. Para ter mais informações, consulte Habilitar o acesso do agente de identidades personalizado ao console da AWS.
- Nome da sessão da função. Use esse valor de string para identificar a sessão quando uma função for usada por diferentes entidades de segurança. Por motivos de segurança, os administradores podem exibir esse campo em logs do AWS CloudTrail para saber quem executou uma ação na AWS. O administrador pode exigir que você forneça um valor específico para o nome da sessão ao assumir a função. Para ter mais informações, consulte `sts:RoleSessionName`.
- (Opcional) Identidade-fonte. Você pode exigir que os usuários federados especifiquem uma identidade-fonte quando assumirem uma função. Depois que a identidade-fonte é definida, o valor não pode ser alterado. Ele estará presente na solicitação para todas as ações realizadas durante a sessão de função. O valor da identidade-fonte persiste nas sessões de função encadeada. Você pode usar informações de identidade-fonte em logs do AWS CloudTrail para determinar quem executou ações com uma função. Para obter mais informações sobre como usar a identidade-fonte, consulte Monitorar e controlar ações realizadas com funções assumidas.
- (Opcional) Políticas de sessão em linha ou gerenciadas. Estas políticas limitam as permissões da política baseada em identidade da função que são atribuídas à sessão da função. 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.

> **Note**
>
> Uma chamada para `AssumeRoleWithWebIdentity` 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.

Quando você chama `AssumeRoleWithWebIdentity`, a AWS verifica a autenticidade do token. Por exemplo, dependendo do provedor, a AWS pode fazer uma chamada para o provedor e incluir o token que o aplicativo passou. Supondo que o provedor de identidade valide o token, 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 `SubjectFromWebIdentityToken` que contém o ID de usuário exclusivo.

Quando você tem as credenciais de segurança temporárias, pode usá-las 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. Como observado, por padrão, as credenciais expiram após uma hora. Se não usar a operação AmazonSTSCredentialsProvider no SDK da AWS, você e sua aplicação serão responsáveis por chamar `AssumeRoleWithWebIdentity` novamente. Chame esta operação para obter um novo conjunto de credenciais de segurança temporárias antes que as antigas expirem.

AssumeRoleWithSAML&emdash; federação por meio de um provedor de identidades corporativo compatível com SAML 2.0

A operação de API `AssumeRoleWithSAML` retorna um conjunto de credenciais de segurança temporárias para usuários federados que são autenticados 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 ter mais informações, consulte Federação SAML 2.0.

> **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.

Esta é uma chamada sem assinatura, o que significa que o aplicativo não precisa ter acesso a nenhuma credencial de segurança da AWS para fazer a chamada. Ao fazer essa chamada, passe as seguintes informações:

- O nome de recurso da Amazon (ARN) da função que o aplicativo deve assumir.
- O ARN do provedor SAML criado no IAM que descreve o provedor de identidade.
- A declaração do SAML, codificada em base64, que foi fornecida pelo provedor de identidade SAML em sua resposta de autenticação à solicitação de login de seu aplicativo.
- Você pode configurar seu IdP para passar atributos para sua declaração do SAML como tags de sessão.
- A duração (opcional), que especifica a duração das credenciais de segurança temporárias. 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 Visualizar a configuração de duração máxima da sessão para uma função. Se você não passar esse parâmetro, as credenciais temporárias vão expirar em uma hora. O parâmetro `DurationSeconds` desta API é diferente do

parâmetro HTTP `SessionDuration` que é usado para especificar a duração de uma sessão do console. Use o parâmetro HTTP `SessionDuration` na solicitação para o endpoint de federação para obter um token de login do console. Para ter mais informações, consulte Habilitar o acesso do agente de identidades personalizado ao console da AWS.

- (Opcional) Políticas de sessão em linha ou gerenciadas. Estas políticas limitam as permissões da política baseada em identidade da função que são atribuídas à sessão da função. 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.
- Nome da sessão da função. Use esse valor de string para identificar a sessão quando uma função for usada por diferentes entidades de segurança. Por motivos de segurança, os administradores podem exibir esse campo em logs do AWS CloudTrail para saber quem executou uma ação na AWS. O administrador pode exigir que você forneça um valor específico para o nome da sessão ao assumir a função. Para ter mais informações, consulte `sts:RoleSessionName`.
- (Opcional) Identidade-fonte. Você pode exigir que os usuários federados especifiquem uma identidade-fonte quando assumirem uma função. Depois que a identidade-fonte é definida, o valor não pode ser alterado. Ele estará presente na solicitação para todas as ações realizadas durante a sessão de função. O valor da identidade-fonte persiste nas sessões de função encadeada. Você pode usar informações de identidade-fonte em logs do AWS CloudTrail para determinar quem executou ações com uma função. Para obter mais informações sobre como usar a identidade-fonte, consulte Monitorar e controlar ações realizadas com funções assumidas.

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 o usuário federado.
- 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.

Quando você tem as credenciais de segurança temporárias, pode usá-las 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.

GetFederationToken: federação por meio de um agente de identidades personalizado

A operação de API `GetFederationToken` retorna um conjunto de credenciais de segurança temporárias para usuários federados. 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.

Ao fazer essa solicitação, você usa as credenciais de um usuário do IAM específico. As permissões das credenciais de segurança temporárias são determinadas pelas políticas de sessão que você transmite ao chamar `GetFederationToken`. As permissões de sessão resultantes são a interseção das políticas de usuário do IAM e as políticas de sessão que você passar. As políticas de sessão não podem ser usadas para conceder mais permissões do que as permitidas pela política baseada em

identidade do usuário do IAM que está solicitando federação. Para obter mais informações sobre as permissões de sessão da função, consulte Políticas de sessão.

Quando você usa as credenciais temporárias retornadas pela operação `GetFederationToken`, as tags principais da sessão incluem as tags do usuário e as tags de sessão passadas. Para obter mais informações sobre tags de sessão, consulte Passar tags de sessão no AWS STS.

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>4</PackedPolicySize>
</GetFederationTokenResult>
<ResponseMetadata>
<RequestId>c6104cbe-af31-11e0-8154-cbc7ccf896c7</RequestId>
</ResponseMetadata>
</GetFederationTokenResponse>
```

> **Note**
>
> Uma conversão da AWS compacta as políticas de sessão e as tags de sessão passadas em um formato binário compactado que têm um limite separado. Sua solicitação pode falhar para esse limite mesmo que seu texto simples atenda aos outros requisitos. O elemento de resposta `PackedPolicySize` indica, em porcentagem, o quão perto as políticas e tags da sua solicitação estão do limite de tamanho superior.

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 o usuário federado, 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`.

GetSessionToken: credenciais temporárias 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. Para obter mais informações, consulte Solicitação de credenciais de segurança temporárias ou GetSessionToken na Referência de API do AWS Security Token Service.

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.

`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. 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>
</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.

Comparação das operações de API 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 Uso de funções do IAM. 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.

Comparação de suas opções de API

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`.
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	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
	de identidade conhecido				
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 ter mais informações, consulte Configuração de acesso à API protegido por 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 função ou um usuário federado. 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 Visualizar a configuração de duração máxima da sessão para uma função.

[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 de usuários federados SAML 2.0 ao AWS Management Console 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 AWS Management Console transmitindo as credenciais para o endpoint de logon único de federação. Para ter mais informações, consulte Habilitar o acesso do agente de identidades personalizado ao console da AWS.

Uso de 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 estiver usando 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 mudará de acordo com 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 ter mais informações, consulte Gerenciar o AWS STS em uma Região da AWS.

Sumário

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 for Java
- Conceder acesso utilizando uma função do IAM — AWS SDK for .NET
- Criar uma função: AWS SDK for 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 for Python (Boto)), consulte Alternância para uma função 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 HTTPS diretas à AWS, você poderá 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.

Controle de permissões para 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.

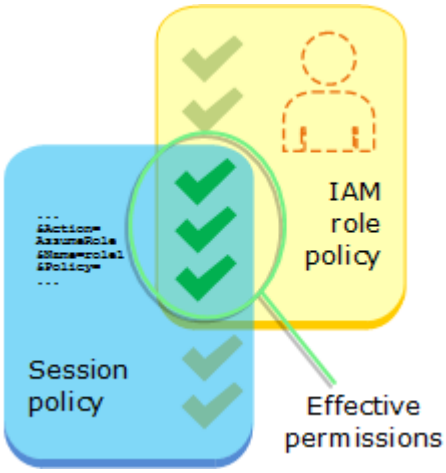
Os tópicos a seguir presumem que você tenha conhecimento das permissões e políticas da AWS. Para mais informações sobre esses tópicos, consulte Gerenciamento de acesso para recursos da AWS.

Tópicos

Permissões para AssumeRole, AssumeRoleWithSAML e AssumeRoleWithWebIdentity

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

```
{
  "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`**

```
{
  "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

```
{
  "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 64 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 AWS Management Console 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

```
{
  "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

```
{
  "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 AWS Management Console, consulte Habilitar o acesso de usuários federados SAML 2.0 ao AWS Management Console. 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 `AssumeRoleWithSAML` da AWS STS. 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 uma função e suas permissões na AWS para seus usuários federados.
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 AWS Management Console, 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": "johndoe",
    "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)

```
{
  "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://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)

```
{
  "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

```
{
  "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

```
{
  "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 ter 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 ao usuário federado são definidas em um de dois lugares:

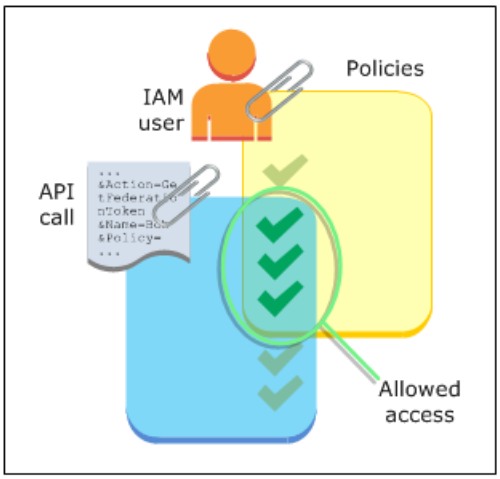
- A política de sessão transmitida como um parâmetro da chamada de API `GetFederationToken`. (Isso é mais comum.)

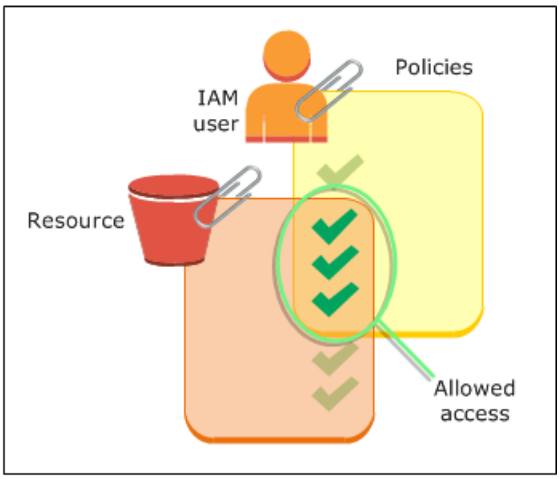
- Uma política com base em recursos que nomeia explicitamente o usuário federado no elemento `Principal` da política. (Isso é menos comum.)

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 e transmite as políticas de sessão, as permissões da sessão resultante são a interseção da política baseada em identidade do usuário do 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 do `token-app` do IAM e define o mais amplo conjunto de permissões necessários para seus usuários federados (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.

> **Note**
>
> A AWS fornece um aplicativo Java de amostra para atender a essa finalidade, o qual você pode fazer download aqui: Máquina de vendas de token para registro de identidade – Aplicativo web Java de amostra.

Example Exemplo de política anexada ao usuário do IAM **`token-app`** que chama **`GetFederationToken`**

```
{
  "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. Se esse usuário do IAM chamar `GetFederationToken` e não transmitir uma política como um parâmetro da chamada de API, o usuário federado resultante não terá permissões efetivas.

Política de sessão transmitida como um parâmetro

A forma mais comum para garantir que o usuário federado 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 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 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`**

```
{
  "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. 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 Serviços da AWS 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. Faça isso ao especificar o nome de recurso da Amazon (ARN) do usuário federado 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 chamado Carol acesse o bucket. Quando a política de exemplo descrita anteriormente é anexada ao usuário do IAM `token-app`, a usuária federada chamado Carol tem permissão para executar as ações `s3:GetObject`, `s3:PutObject` e `s3:DeleteObject` 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 seguinte política baseada em recursos explicitamente concedeu permissões ao usuário federado chamado Carol.

Lembre-se de que um usuário federado recebe permissões apenas quando essas permissões são concedidas explicitamente ao usuário do IAM e ao usuário federado. Elas também podem ser concedidas (dentro de uma conta) por uma política baseada em recurso que nomeie explicitamente o usuário federado no elemento `Principal` da política, como no exemplo a seguir.

Example Exemplo de política de bucket que permite acesso ao usuário federado

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Principal": {"AWS": "arn:aws:sts::account-id: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 AWS Management Console. Para executar essas operações, você deve ter uma política que permite acessar a ação `CreateUser`.

```
{
    "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 AWS Management Console especificando as credenciais para o endpoint de logon único de federação em `https://signin.aws.amazon.com/federation`. Para ter mais informações, consulte Habilitar o acesso do agente 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 Comparação das operações de API do AWS STS

Para obter mais informações sobre o acesso de API protegido por MFA usando `GetSessionToken`, consulte Configuração de acesso à API protegido por 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). É possível revogar essas credenciais, mas também é necessário alterar as permissões do perfil para impedir que atividades mal-intencionadas na conta usem credenciais comprometidas. 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. Revogue as credenciais de segurança temporárias do perfil para forçar todos os usuários que assumem o perfil a se autenticarem novamente e a solicitarem novas credenciais.

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.

> **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 associadas a um perfil
- Negar acesso a uma sessão específica
- Negar uma sessão de usuário com chaves de contexto de condição
- Negar um usuário de sessão com políticas baseadas em recursos

Negar acesso a todas as sessões 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

Esse procedimento nega permissões para todos os usuários que tenham permissões para assumir um perfil.

Para alterar ou remover permissões atribuídas a credenciais de segurança temporárias obtidas chamando as APIs `AssumeRole`, `AssumeRoleWithSAML`, `AssumeRoleWithWebIdentity`, `GetFederationToken` ou `GetSessionToken`, edite ou exclua a política de permissões 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 um usuário de sessão com políticas baseadas em recursos.

1. Faça login no AWS Management Console e abra o console do IAM.
2. No painel de navegação, escolha o nome do perfil a ser editado. Você pode usar a caixa de pesquisa para filtrar a lista.
3. Selecione a política relevante.
4. Escolha a aba Permissões.
5. Escolha a guia JSON e atualize a política para negar todos os recursos e ações.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```

6. 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 sessão específica

Quando você atualiza os perfis que podem ser assumidos com base em um IdP com uma política de negação ou exclui totalmente o perfil, todos os usuários com acesso ao perfil são interrompidos. É possível negar o acesso com base no elemento `Principal` 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 uma sessão de usuário com chaves de contexto de condição

Você pode usar chaves de contexto de condição em situações em que deseja negar acesso a sessões temporárias de credenciais de segurança específicas sem afetar as permissões do usuário ou perfil do IAM que criaram as credenciais.

Para obter mais informações sobre chaves de contexto de condição, consulte Chaves de contexto de condição globais da AWS.

> **Note**
>
> Se houver uma política baseada em recursos que permita o acesso da entidade principal, você também deverá adicionar uma instrução de negação explícita à política baseada em recursos após completar essas etapas.

Após atualizar a política, você poderá revogar as credenciais de segurança temporárias do perfil para revogar imediatamente todas as credenciais emitidas.

aws:PrincipalArn

É possível usar a chave de contexto de condição aws:PrincipalArn para negar acesso a um ARN de entidade principal específico. Faça isso especificando o identificador exclusivo (ID) do usuário, perfil ou usuário federado do IAM ao qual as credenciais de segurança temporárias estão associadas na condição de uma política.

1. No painel de navegação do console do IAM, escolha o nome do perfil a ser editado. Você pode usar a caixa de pesquisa para filtrar a lista.
2. Selecione a política relevante.
3. Escolha a aba Permissões.
4. Escolha a guia JSON e adicione uma instrução de negação para o ARN da entidade principal, conforme mostrado no exemplo a seguir.

```
{
  "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:sts::222222222222:federated-user/USERNAME"
          ]
        }
      }
    }
  ]
}
```

5. Na página Revisão, revise o Resumo da política e, em seguida, selecione Salvar alterações para salvar seu trabalho.

aws:userid

Você pode usar a chave de contexto de condição aws:userid para negar acesso a todas ou a sessões temporárias de credenciais de segurança específicas que estão associadas ao usuário ou perfil do IAM. Faça isso especificando o identificador exclusivo (ID) do usuário, perfil ou usuário federado do IAM ao 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 identificador exclusivo de um usuário do IAM. Especificar o identificador exclusivo de um usuário do IAM como valor para a chave de contexto `aws:userid` negará todas as sessões associadas ao usuário do IAM.
- `AROAXROLE1` representa o identificador exclusivo de um perfil do IAM. Especificar o identificador exclusivo de um perfil do IAM como valor para a chave de contexto `aws:userid` negará todas as sessões associadas ao perfil.

- `AROAXROLE2` representa o identificador exclusivo de uma sessão assumed-role. Na parte caller-specified-role-session-name do identificador exclusivo assumed-role, você pode 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 curinga para o nome da sessão do perfil, ele negará todas as sessões associadas ao perfil.
- `account-id:<federated-user-caller-specified-name>` representa o identificador exclusivo de uma sessão de usuário federado. O usuário federado é criado por um usuário do IAM que chama a API GetFederationToken. Se você especificar o identificador exclusivo para o usuário federado, ele negará a sessão do usuário federado nomeado sem afetar as permissões do perfil que criou as credenciais.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "aws:userId": [
            "AIDAXUSER1",
            "AROAXROLE1",
            "AROAXROLE2:<caller-specified-role-session-name>",
            "account-id:<federated-user-caller-specified-name>"
          ]
        }
      }
    }
  ]
}
```

Para exemplos específicos de valores de chave de entidade principal, consulte Valores de chave de principal. Para obter mais informações sobre identificadores exclusivos do IAM, consulte Identificadores exclusivos.

Negar um usuário de sessão com políticas baseadas em recursos

Se o ARN da entidade principal também estiver incluído em qualquer política baseada em recursos, você também deverá revogar o acesso com base nos valores `principalId` e `sourceIdentity` do usuário específico no elemento `Principal` de uma política baseada em recursos. Se você atualizar somente a política de permissões para o perfil, o usuário ainda poderá realizar as ações permitidas na política baseada em recursos.

1. Consulte Serviços da AWS que funcionam com o IAM para ver se o serviço oferece suporte a políticas baseadas em recursos.
2. Faça login no AWS Management Console e abra o console do serviço. Cada serviço tem uma localização diferente no console para anexar políticas.
3. Edite a instrução de política para especificar as informações de identificação da credencial:
   a. Em `Principal`, insira o ARN da credencial a ser negada.
   b. Em `Effect`, digite “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. Em `Resource`, insira o ARN do recurso de destino. Por exemplo: arn:aws:s3:::EXAMPLE-BUCKET.”

```
{
"Version": "2012-10-17",
  "Statement": {
    "Principal": [
            "arn:aws:iam::222222222222:role/ROLENAME",
            "arn:aws:iam::222222222222:user/USERNAME",
            "arn:aws:sts::222222222222:federated-user/USERNAME"
    ],
    "Effect": "Deny",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:::EXAMPLE-BUCKET"
  }
}
```

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 funções e usuários federados. 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 usuários federados ou funções, anexe uma política que conceda um ou ambos os seguintes privilégios:

- Para usuários federados acessarem uma função do IAM conceda acesso a `AssumeRole` doAWS STS.
- Para usuários federados que não precisam de uma função, conceda acesso ao AWS STS do `GetFederationToken`.

Para obter mais informações sobre as diferenças entre o `AssumeRole` e `GetFederationToken` operações de API, consulte Solicitação de 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.

```
{
  "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`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": "*"
  }]
}
```

> **Important**
>
> Quando você dá permissão a usuários do IAM para criar credenciais de segurança temporárias para usuários federados 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 Controle de permissões para 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 cujos nomes comecem com Manager (Gerente).

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:GetFederationToken",
    "Resource": ["arn:aws:sts::123456789012:federated-user/Manager*"]
  }]
}
```

Como conceder permissões para usar sessões de console com reconhecimento de identidade

As sessões de console com reconhecimento de identidade permitem que as IDs de sessão e de usuário de AWS IAM Identity Center sejam incluídos nas sessões de console de usuários da AWS quando eles fazem login. Por exemplo, o Amazon Q Developer Pro usa sessões de console com reconhecimento de identidade para personalizar a experiência do serviço. Para obter mais informações sobre sessões de console com reconhecimento de identidade, consulte Habilitar sessões de console com reconhecimento de identidade no Guia do usuário da AWS IAM Identity Center. 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 sessões de console com reconhecimento de identidade estejam disponíveis para um usuário, você deve usar uma política baseada em identidade para conceder a entidade principal do IAM a permissão `sts:SetContext` para o recurso que representa sua própria sessão de console.

> **Important**
>
> Por padrão, os usuários não têm permissão para definir o contexto para suas sessões de console com reconhecimento de 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 a seguir de política baseada em identidade concede a permissão `sts:SetContext` a uma entidade principal do IAM, permitindo que o diretor defina um contexto de sessão de console com reconhecimento de identidade para suas próprias sessões de 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sts:SetContext",
      "Resource": "arn:aws:sts::account-id:self"
    }
  ]
}
```

Gerenciar o AWS STS em uma Região da AWS

Por padrão, o AWS Security Token Service (AWS STS) está disponível como um serviço global, e todas as solicitações do AWS STS vão para um único endpoint em `https://sts.amazonaws.com`. A AWS recomenda o uso de endpoints regionais do AWS STS, em vez do endpoint global, para reduzir a latência, incorporar redundância e aumentar a validade do token de sessão.

- 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.

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 Account Management. 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 AWS Management Console, 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`

Ativar e desativar o AWS STS em uma Região da AWS

Quando você ativar endpoints do 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 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. Para regiões habilitadas manualmente, é necessário ativar a região na conta que faz a solicitação e na conta em que as credenciais temporárias são geradas.

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-east-1.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 ativar a região `ap-east-1`. Os usuários da conta A (ou de qualquer outra conta) podem chamar o endpoint do AWS STS `ap-east-1` para solicitar credenciais para a conta B, esteja ou não ativada a região em suas contas.

> **Note**
>
> Regiões ativas estão disponíveis para todas as pessoas que usam credenciais temporárias nessa conta. Para controlar quais usuários ou funções 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. Em Security Token Service (STS), na seção Endpoints, localize a região que deseja configurar e escolha Active (Ativa) ou Inactive (Inativa) na coluna de STS status (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 Account Management.

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();
```

O AWS STS 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 Account Management.

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 como configurar endpoints regionais do AWS STS usando uma variável de ambiente paraAWS 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.

Regiões e endpoints

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	 Sim	 Sim
Leste dos EUA (Norte da Virgínia)	sts.us-east-1.amazonaws.com	 Sim	 No (Não)

Nome da região	Endpoint	Ativa por padrão	Ativar/ desativar manualmente
Oeste dos EUA (N. da Califórnia)	sts.us-west-1.amazonaws.com	Sim	Sim
Oeste dos EUA (Oregon)	sts.us-west-2.amazonaws.com	Sim	Sim
África (Cidade do Cabo)	sts.af-south-1.amazonaws.com	Não	No (Não)
Ásia-Pacífico (Hong Kong)	sts.ap-east-1.amazonaws.com	Não	No (Não)
Ásia-Pacífico (Hyderabad)	sts.ap-south-2.amazonaws.com	Não	No (Não)
Ásia-Pacífico (Jacarta)	sts.ap-southeast-3.amazonaws.com	Não	No (Não)

Nome da região	Endpoint	Ativa por padrão	Ativar/ desativar manualmente
Ásia-Pacífico (Melbourne)	sts.ap-southeast-4.amazonaws.com	Não	No (Não)
Ásia-Pacífico (Mumbai)	sts.ap-south-1.amazonaws.com	Sim	Sim
Asia Pacific (Osaka)	sts.ap-northeast-3.amazonaws.com	Sim	Sim
Ásia-Pacífico (Seul)	sts.ap-northeast-2.amazonaws.com	Sim	Sim
Ásia-Pacífico (Singapura)	sts.ap-southeast-1.amazonaws.com	Sim	Sim
Ásia-Pacífico (Sydney)	sts.ap-southeast-2.amazonaws.com	Sim	Sim

Nome da região	Endpoint	Ativa por padrão	Ativar/ desativar manualmente
Ásia-Pacífico (Tóquio)	sts.ap-northeast-1.amazonaws.com	Sim	Sim
Canadá (Central)	sts.ca-central-1.amazonaws.com	Sim	Sim
Oeste do Canadá (Calgary)	sts.ca-west-1.amazonaws.com	Sim	Sim
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	Sim	Sim

Nome da região	Endpoint	Ativa por padrão	Ativar/ desativar manualmente
Europa (Irlanda)	sts.eu-west-1.amazonaws.com	Sim	Sim
Europa (Londres)	sts.eu-west-2.amazonaws.com	Sim	Sim
Europa (Milão)	sts.eu-south-1.amazonaws.com	Não	No (Não)
Europa (Paris)	sts.eu-west-3.amazonaws.com	Sim	Sim
Europa (Espanha)	sts.eu-south-2.amazonaws.com	Não	No (Não)
Europa (Estocolmo)	sts.eu-north-1.amazonaws.com	Sim	Sim

Nome da região	Endpoint	Ativa por padrão	Ativar/ desativar manualmente
Europa (Zurique)	sts.eu-central-2.amazonaws.com	Não	No (Não)
Israel (Tel Aviv)	sts.il-central-1.amazonaws.com	Não	No (Não)
Oriente Médio (Barém)	sts.me-south-1.amazonaws.com	Não	No (Não)
Oriente Médio (Emirados Árabes Unidos)	sts.me-central-1.amazonaws.com	Não	No (Não)
América do Sul (São Paulo)	sts.sa-east-1.amazonaws.com	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.

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 ter mais informações, consulte Registro em log de chamadas de API do IAM e do AWS STS com o AWS CloudTrail.

Usar tokens de portador

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 Solicitação assinada pelo Signature Versão 4 tradicional. 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.

Os tokens de portador do serviço AWS STS 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:

```
{
    "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.

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 AWSusando Windows Active Directory (AD), Active Directory Federation Services (ADFS) 2.0 e SAML (Security Assertion Markup Language) 2.0.

- Habilitar o acesso do agente 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 AWS Management Console.
- Como usar o Shibboleth para autenticação única no AWS Management Console.. Mostra como usar o Shibboleth e o SAML para fornecer aos usuários logon único (SSO) ao AWS Management Console.

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.

Habilitar o acesso do agente 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 AWS Management Console. 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 AWS Management Console.

> **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 de usuários federados SAML 2.0 ao AWS Management Console.

Para permitir que os usuários de sua organização acessem o AWS Management Console, você pode criar um agente 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 Uso de funções do IAM. 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 Visualizar a configuração de duração máxima da sessão para uma função.
   - 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**
  >
  > - Não use o parâmetro HTTP `SessionDuration` se você obteve suas credenciais temporárias com `GetFederationToken`. Isso fará com que a operação falhe.
  > - 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.

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 AWS Management Console 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 ofereça aos seus usuários federados acesso direto ao AWS Management Console. 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 dar a um usuário federado acesso aos seus recursos a partir do AWS Management Console

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 Controle de permissões para 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**
>
> A AWS 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 login da). 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**
>
> - Não use o parâmetro HTTP `SessionDuration` se você obteve suas credenciais temporárias com `GetFederationToken`. Isso fará com que a operação falhe.
> - 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.

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 ter mais informações, consulte Revogação das credenciais de segurança temporárias da função 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**
>
> A AWS é compatível com solicitações POST aqui.

Finalmente, crie o URL que seus usuários federados podem usar para acessar o AWS Management Console. 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 uma URL que ofereça aos usuários federados acesso direto ao AWS Management Console. 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 AWS SDK for Python (Boto3) e a API AssumeRole para obter credenciais de segurança temporárias.

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 federated user

# 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 federated user

# 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);
```

Exemplo de código usando Java

O exemplo a seguir mostra como usar Java para programaticamente construir um URL que oferece aos usuários federados acesso direto ao AWS Management Console. 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 programaticamente construir um URL que oferece aos usuários federados acesso direto ao AWS Management Console. 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
```

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`.
- Configuração de acesso à API protegido por 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?.

Recursos de etiquetas do IAM

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.

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 membro do AWS Organizations, consulte Políticas de tag no Guia do Usuário do Organizations para saber mais sobre como usar tags no 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 AWS 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 AWS Resource Groups.
- 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`**.

Tópicos

Etiquetar 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 AWS Management Console.

Gerenciar tags em usuários (console)

1. Faça login no AWS Management Console 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 Save changes (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.

Etiquetar funções 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 ter 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 AWS Management Console.

Gerenciar tags em funções (console)

1. Faça login no AWS Management Console 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 AWS Management Console.

Gerenciar tags em políticas gerenciadas pelo cliente (console)

1. Faça login no AWS Management Console 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 Save changes (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.

Etiquetar provedores de identidade do IAM

Você pode usar pares de chave-valor de etiqueta do IAM para adicionar atributos personalizados a provedores de identidade (IdPs) do IAM.

Você também pode usar tags no AWS STS para adicionar atributos personalizados ao assumir uma função ou federar um usuário. Para ter mais informações, consulte Passar tags de sessão no AWS STS.

Para saber mais sobre como aplicar tags IdPs no IAM, consulte os tópicos a seguir:

Tópicos

- Marcar provedores de identidade OpenID Connect (OIDC)
- Etiquetamento de provedores de identidade SAML do IAM

Marcar provedores de identidade 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 (usuário ou função) do IAM adicione, liste ou remova uma etiqueta para um provedor de identidade 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 AWS Management Console.

Gerenciar tags em provedores de identidade OIDC (console)

1. Faça login no AWS Management Console 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. Na seção Tags, escolha Manage tags (Gerenciar tags) e 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.

Etiquetamento de provedores de identidade 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 AWS Management Console.

Gerenciar tags em provedores de identidade SAML (console)

1. Faça login no AWS Management Console 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. Na seção Tags, escolha Manage tags (Gerenciar tags) e 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.

Etiquetamento de 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 de 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 AWS Certificate Manager (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 AWS Certificate Manager 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 ter 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 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 ter 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.

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

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 AWS Management Console 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 ter mais informações, consulte Encadeamento de funções com tags de sessão.

Comparar métodos para passar tags de sessão

Operation	Quem pode assumir a função	Método para passar tags	Método para definir tags transitivas
Operação da CLI `assume-role` ou	Usuário ou sessão do IAM	Parâmetro da API `Tags` ou opção da CLI `--tags`	Parâmetro da API `TransitiveTagKeys` ou

Operation	Quem pode assumir a função	Método para passar tags	Método para definir tags transitivas
da API AssumeRole			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	Atributo SAML `PrincipalTag`	Atributo SAML `TransitiveTagKeys`
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 suportado

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.

- O tamanho total compactado das políticas e etiquetas de sessão combinadas for muito grande. Se a operação falhar, a mensagem de erro indicará, em porcentagem, o quão perto as políticas e etiquetas combinadas estão do limite de tamanho superior.

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 a função assumida existente ou as etiquetas de usuário federado pela mesma chave de etiqueta, independentemente da capitalização dos caracteres.
- Você não pode passar tags de sessão usando o AWS Management Console.
- 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 ter 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 ter 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 ter 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 ter mais informações, consulte IAM e cotas do AWS STS.
- Uma conversão da AWS compacta as políticas de sessão passadas e as etiquetas de sessão combinadas em um formato binário compactado que tem um limite separado. Se você exceder esse limite, a mensagem de erro da AWS CLI ou da API da AWS mostrará, em porcentagem, o quão perto as políticas e etiquetas combinadas estão do limite de tamanho superior.

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 ter 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

```
{
    "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 ter 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.

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 AWS Management Console, consulte Habilitar o acesso de usuários federados SAML 2.0 ao AWS Management Console. 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 `AssumeRoleWithSAML` da AWS STS. 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. Configurar uma função e permissões na AWS para seus usuários federados
4. Concluir a configuração do IdP SAML e criar declarações para a resposta de autenticação SAML

A AWS 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 ter 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 a 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 AWS Management Console acesso, consulte Federação OIDC.

Para passar tags de sessão do OpenID Connect (OIDC), é necessário incluir as tags de sessão no JSON Web Token (JWT). Inclua tags de sessão no namespace `https://aws.amazon.com/` `tags` do 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 usa um token para chamar `AssumeRoleWithWebIdentity` com as etiquetas de sessão `Project`, `CostCenter` e `Department`. O token também define as tags `Project` e `CostCenter` como transitivas. As tags transitivas persistem durante o encadeamento de funções. Para ter mais informações, consulte Encadeamento de funções com tags de sessão.

Example Exemplo de JSON Web Token decodificado

```
{
    "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"
        ]
    }
}
```

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.

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.

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 etiquetas `Heart` e `Star` herdam as etiquetas 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.

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 etiqueta `Lightning` da função também se aplica à terceira sessão e não é definida como transitiva.

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 AWS IAM Identity Center.

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 ter 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 `AssumeRoleWithSAML` do AWS STS, 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

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. Você pode 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

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 obter mais informações, 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 Amazon S3. Por padrão, quando você cria uma trilha no console, ela é aplicada a todas as Regiões da . A trilha registra em log eventos de todas as Regiões na partição da AWS e entrega os arquivos de log para o bucket do Amazon S3 especificado por você. 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 Compatíveis com 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.

Você pode usar a informação registrada em log para mapear as chamadas feitas por um usuário federado 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 um usuário federado específico. Para obter mais informações, consulte o Elemento userIdentity do CloudTrail no Guia do usuário do AWS 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 eventos de login no AWS Management Console, nos fóruns de discussão da AWS e no AWS Marketplace. O CloudTrail registra tentativas de login bem-sucedidas e malsucedidas para usuários e usuários federados do IAM.

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 de segurança solicita credenciais temporárias, o tipo de entidade de segurança 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.

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 ter 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
Credenciais do Usuário raiz da conta da AWS	GetSessionToken	Identidade do usuário raiz	A conta do proprietário da função é a mesma que a conta de chamada	Identidade do usuário raiz
IAM user (Usuário do IAM)	GetSessionToken	Identidade do usuário do IAM	A conta do proprietário da função 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	Identidade do usuário do IAM

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
			da função é a mesma que a conta de chamada	
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 da função (nenhum usuário)
Usuário autenticado externamente	AssumeRoleWithSAML	n/a	Identidade do usuário do SAML	Somente identidade da função (nenhum usuário)
Usuário autenticado externamente	AssumeRoleWithWebIdentity	n/a	Identidade do usuário da web/ OIDC	Somente identidade da função (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 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	verdadeiro
AssumeRoleWithSAML	Incluído	verdadeiro
AssumeRoleWithWebIdentity	Incluído	verdadeiro
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": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/JaneDoe",
    "accountId": "444455556666",
    "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
    "userName": "JaneDoe",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2014-07-15T21:39:40Z"
```

```
      }
    },
    "invokedBy": "signin.amazonaws.com"
  },
  "eventTime": "2014-07-15T21:40:14Z",
  "eventSource": "iam.amazonaws.com",
  "eventName": "GetUserPolicy",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "signin.amazonaws.com",
  "userAgent": "signin.amazonaws.com",
  "requestParameters": {
    "userName": "JaneDoe",
    "policyName": "ReadOnlyAccess-JaneDoe-201407151307"
  },
  "responseElements": null,
  "requestID": "9EXAMPLE-0c68-11e4-a24e-d5e16EXAMPLE",
  "eventID": "cEXAMPLE-127e-4632-980d-505a4EXAMPLE"
}
```

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 AWS Management Console, 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 `JohnDoe` na conta 777788889999 chama a ação `AssumeRole` do AWS STS para assumir a função `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 `JohnDoe`.

```
{
```

```
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAQRSTUVWXYZEXAMPLE",
    "arn": "arn:aws:iam::777788889999:user/JohnDoe",
    "accountId": "777788889999",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "JohnDoe"
  },
  "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": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "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:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
    },
  "sourceIdentity": "JohnDoe"
  },
  "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": "JohnDoe-EC2-dev",
    "sourceIdentity": "JohnDoe",
    "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:JohnDoe-EC2-dev",
      "arn": "arn:aws:sts::111122223333:assumed-role/EC2-dev/JohnDoe-EC2-dev"
      },
  "sourceIdentity": "JohnDoe"
  },
  "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 `JohnDoe` 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 `JohnDoe1` 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/JohnDoe/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": "JohnDoe"
            },
            "sourceIdentity": "JohnDoe"
        }
    },
    "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": "JohnDoe",
        "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": "JohnDoe"
    },
    "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": "my-test-bucket-cross-account"
  },
  "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 de log do CloudTrail para uma solicitação 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": 1,
        "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 a seguir mostra um registro de log do CloudTrail para uma solicitação pela ação `AssumeRoleWithWebIdentity` do AWS STS. A solicitação inclui os atributos `CostCenter` e `Project` que são passados por meio do token de provedor de identidade 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.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "accounts.google.com:<id-of-
application>.apps.googleusercontent.com:<id-of-user>",
    "userName": "<id of user>",
```

```
    "identityProvider": "accounts.google.com"
  },
  "eventTime": "2016-03-23T01:39:51Z",
  "eventSource": "sts.amazonaws.com",
  "eventName": "AssumeRoleWithWebIdentity",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "192.0.2.101",
  "userAgent": "aws-cli/1.3.23 Python/2.7.6 Linux/2.6.18-164.el5",
  "requestParameters": {
    "sourceIdentity": "MyWebIdentityUser",
    "durationSeconds": 3600,
    "roleArn": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
    "roleSessionName": "MyAssignedRoleSessionName"
        "principalTags": {
            "CostCenter": "24680",
            "Project": "Pegasus"
        },
        "transitiveTagKeys": [
            "CostCenter",
            "Project"
        ],
  },
  "responseElements": {
    "provider": "accounts.google.com",
    "subjectFromWebIdentityToken": "<id of user>",
    "sourceIdentity": "MyWebIdentityUser",
    "audience": "<id of application>.apps.googleusercontent.com",
    "credentials": {
      "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
      "expiration": "Mar 23, 2016, 2:39:51 AM",
      "sessionToken": "<encoded session token blob>"
    },
    "assumedRoleUser": {
      "assumedRoleId": "AROACQRSTUVWRAOEXAMPLE:MyAssignedRoleSessionName",
      "arn": "arn:aws:sts::444455556666:assumed-role/FederatedWebIdentityRole/
MyAssignedRoleSessionName"
    }
  },
  "resources": [
    {
      "ARN": "arn:aws:iam::444455556666:role/FederatedWebIdentityRole",
      "accountId": "444455556666",
      "type": "AWS::IAM::Role"
    }
```

```
  ],
  "requestID": "6EXAMPLE-e595-11e5-b2c7-c974fEXAMPLE",
  "eventID": "bEXAMPLE-0b30-4246-b28c-e3da3EXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "444455556666"
}
```

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/JohnDoe",
    "accountId": "111122223333",
    "userName": "JohnDoe"
  },
  "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"
}
```

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

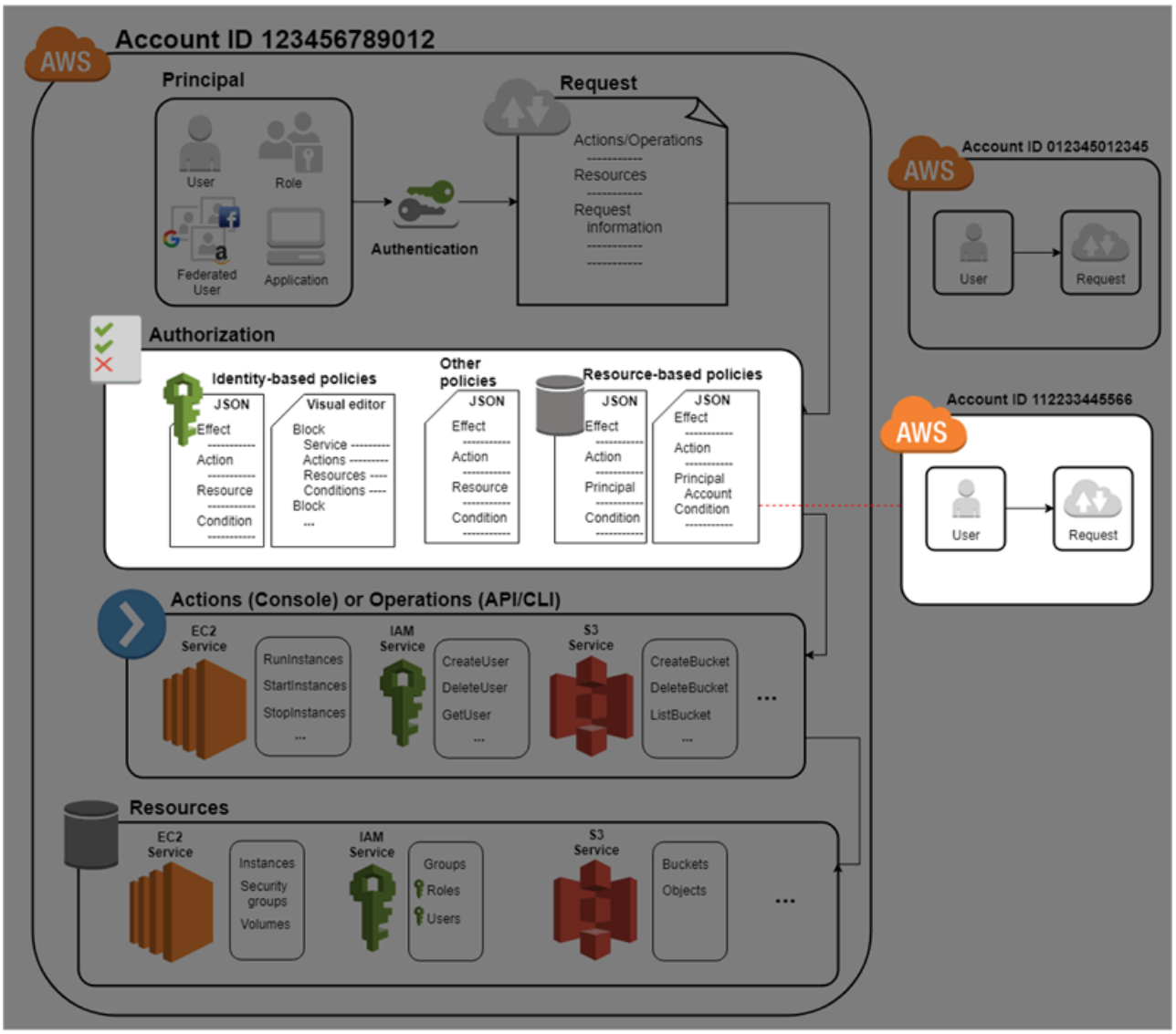
Em 21 de setembro de 2022, a AWS fez alterações no comportamento da política de confiança de perfil do IAM para exigir permissões explícitas em uma política de segurança de perfil quando um perfil assume a si mesmo. Os perfis do IAM na lista de permissões de comportamento legadas têm um campo additionalEventData 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.

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 IAM.

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.

O AWS 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, uma SCP do Organizations ou uma 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.

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 IAM

Você gerencia o acesso na AWS criando políticas e anexando-as às identidades do IAM (usuários, grupos de usuários ou funções) ou aos 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 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. A AWS oferece suporte a seis tipos de políticas: políticas baseadas em identidade, políticas baseadas em recurso, limites de permissões, SCPs do Organizations, ACLs 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 AWS Management Console, 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.
- 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 Organizations: use uma política de controle de serviço (SCP) do AWS Organizations para definir o número máximo de permissões para os membros da conta de uma organização ou unidade organizacional (UO). As SCPs limitam as permissões que as políticas baseadas em identidade ou políticas baseadas em recurso concedem a entidades (usuários ou funções) dentro da conta, mas 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.
- Políticas de sessão: transmita políticas de sessão avançadas ao usar a AWS CLIou 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 função. 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 em linha.

Políticas baseadas em atributos

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 ao principal especificado para executar ações específicas nesse recurso e definem em que condições isso se aplica. As políticas baseadas em recurso são políticas em linha. Não há políticas baseadas em recurso 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 atributo. Adicionar uma entidade principal entre contas à política baseada em atributo 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 atributo 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 Serviços da AWS 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?.

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, a entidade poderá executar apenas as ações que são permitidas por ambas as políticas baseadas em identidade e seus limites de permissões. As políticas baseadas em recurso que especificam o usuário ou a função como entidades principais não são limitadas pelo limite de permissões. 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.

Políticas de controle de serviço (SCPs)

O AWS Organizations é um serviço para agrupar e gerenciar centralmente as Contas da AWS que a sua empresa possui. Se você habilitar todos os atributos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. SCPs são políticas JSON que especificam o máximo de permissões para uma organização ou unidade organizacional (UO). O SCP limita as permissões para entidades em contas-membro, incluindo cada .Usuário raiz da conta da AWS Uma negação explícita em qualquer uma dessas políticas substitui a permissão.

Para obter mais informações sobre o Organizations e SCPs, consulte Como as SCPs funcionam 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 compatíveis 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.

Políticas de sessão

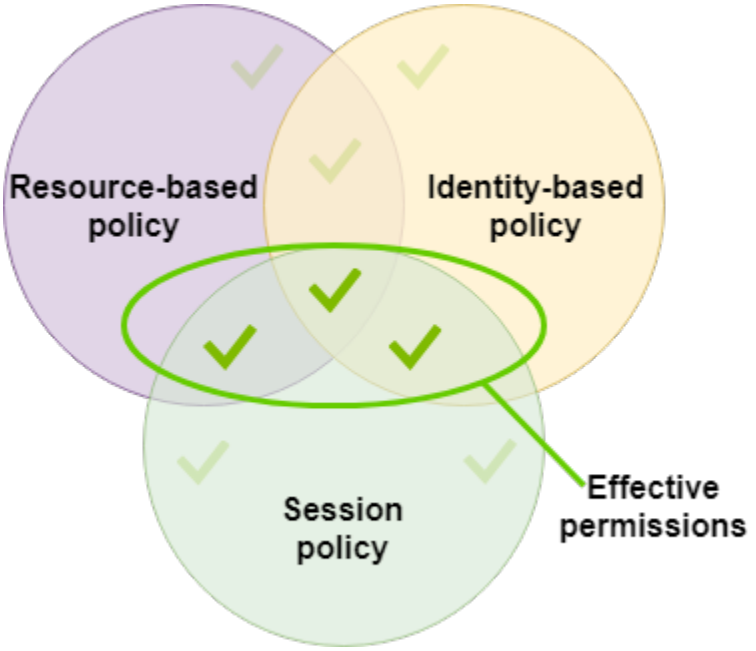
As políticas de sessão são políticas avançadas que você transmite como um parâmetro quando você cria de forma programática uma sessão temporária para uma função ou usuário federado. As permissões para uma sessão são a interseção das políticas baseadas em identidade para a entidade (usuário ou função) do IAM usada para criar a sessão e as políticas da sessão. As permissões também podem ser provenientes de uma política baseada em atributo. Uma negação explícita em qualquer uma dessas políticas substitui a permissão.

Você pode criar uma sessão de função e transmitir políticas de sessão de forma programática usando as operações de API `AssumeRole`, `AssumeRoleWithSAML` ou

`AssumeRoleWithWebIdentity`. Você pode passar um único documento de política JSON de sessão em linha usando o parâmetro `Policy`. Você pode 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 Solicitação de credenciais de segurança temporárias.

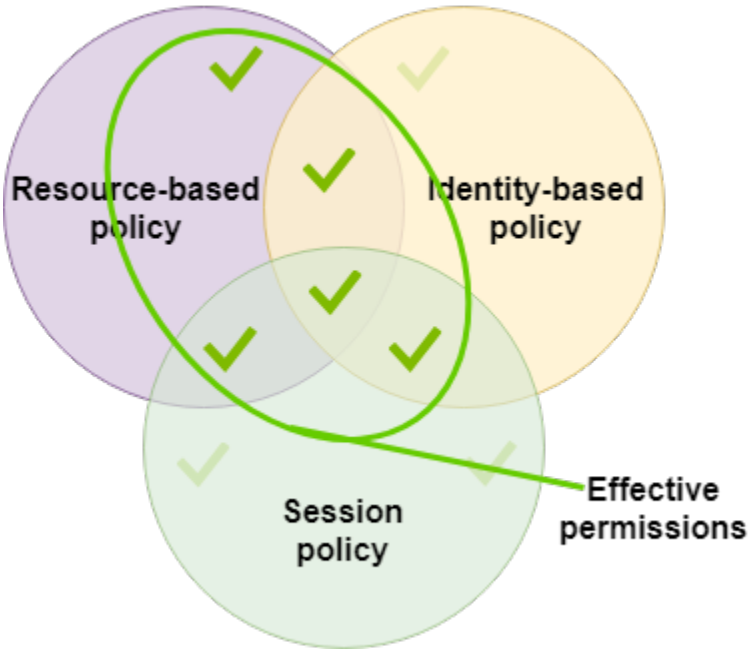
Quando você criar uma sessão de usuário federado, use as chaves de acesso do usuário do IAM para chamar de forma programática a operação da API `GetFederationToken`. Você também deve transmitir as políticas da sessão. As permissões da sessão resultam da interseção da política baseada em identidade do usuário e da política de sessão. Para obter mais informações sobre a criação de uma sessão de usuário federado, consulte GetFederationToken: federação por meio de um agente 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.

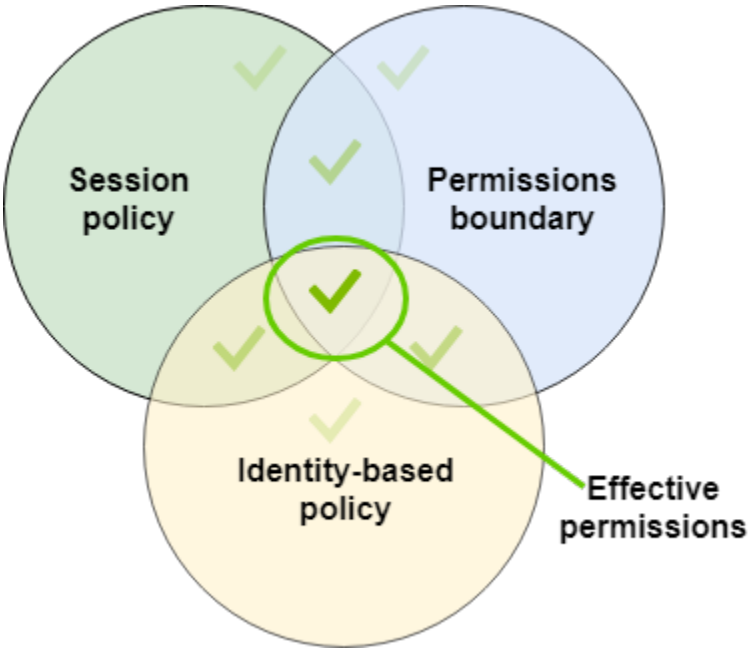


Uma política baseada em recurso pode especificar o ARN da sessão como uma entidade principal. Nesse caso, as permissões da política baseada em recurso são adicionadas depois que a sessão for criada. As permissões da política baseada em recurso não são limitadas pela política de sessão. A

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 de segurança 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 quaisquer SCPs da 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 são documentos de políticas JSON com sintaxe restrita que você anexa a uma unidade organizacional (UO) do AWS Organizations. 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. Você pode usar o editor visual no AWS Management Console 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 Criação de políticas do IAM e Edição de políticas do IAM.

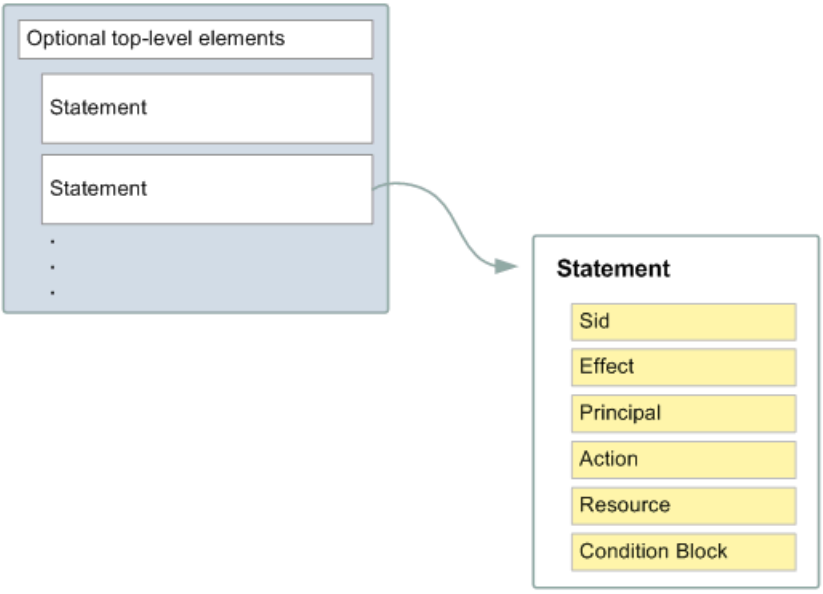
Quando você cria ou edita uma política JSON, o IAM pode executar a validação de políticas para ajudar você a criar uma política eficaz. O IAM identifica erros de sintaxe JSON, enquanto o IAM Access Analyzer fornece verificações de políticas adicionais com recomendações para ajudar você a refinar ainda mais suas políticas. Para saber mais sobre validação de política, consulte Validação de políticas 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 ter 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. Você pode 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 apenas em algumas circunstâncias): se você criar uma política baseada em recurso, deverá indicar a conta, o usuário, a função ou o usuário federado 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 apenas 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, esse elemento será opcional. Se você não incluir esse elemento, o recurso ao qual a ação se aplica será o recurso ao qual a política está anexada.

- 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 elementos 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 `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 `confidential-data`.

```
{
  "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:::confidential-data",
        "arn:aws:s3:::confidential-data/*"
      ],
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
}
```

Exemplos de sintaxe de política JSON

A seguinte política baseada em identidade permite que a entidade de segurança implícita liste um único bucket do Amazon S3 chamado `example_bucket`:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_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 `mybucket`. 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.)

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "1",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::account-id:root"]},
    "Action": "s3:*",
    "Resource": [
      "arn:aws:s3:::mybucket",
      "arn:aws:s3:::mybucket/*"
    ]
  }]
}
```

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 Noções básicas sobre níveis de acesso em resumos de políticas.
- 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. Você pode 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. 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 Gerar políticas com base na atividade de acesso.

- 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 Organizations. Você pode usar essa informação para identificar permissões desnecessárias, de forma que você possa refinar suas políticas do IAM ou do Organizations para melhor aderir ao princípio de privilégio mínimo. Para ter mais informações, consulte Refinar permissões na AWS usando as 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.

- Controle de acesso e autenticação do Amazon DynamoDB no Guia do desenvolvedor do Amazon DynamoDB
- Políticas de bucket e políticas de usuário 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.

Tópicos

Políticas gerenciadas pela AWS

Uma política gerenciada pela AWS é uma política independente que é criada e administrada pela AWS. Política independente significa que a política tem seu próprio nome de 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, consulte ARNs do IAM). Para obter uma lista das políticas gerenciadas do AWS para o Serviços da AWS, consulte Políticas gerenciadas pela AWS.

As políticas gerenciadas pela AWS facilitam a atribuição das devidas permissões a usuários, grupos e perfis. 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 fizer isso, a atualização afetará todas as entidades principais (usuários, grupos e funções) às quais a política está anexada. É mais provável que a AWS atualize uma política gerenciada da 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 serviços e recursos 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.

As políticas gerenciadas pela AWS de acesso total definem permissões para administradores de serviço concedendo acesso total a um serviço.

- AmazonDynamoDBFullAccess
- IAMFullAccess

As políticas gerenciadas pela AWS para usuários avançados fornecem acesso total a serviços e recursos da AWS, mas não permitem o gerenciamento de usuários e grupos.

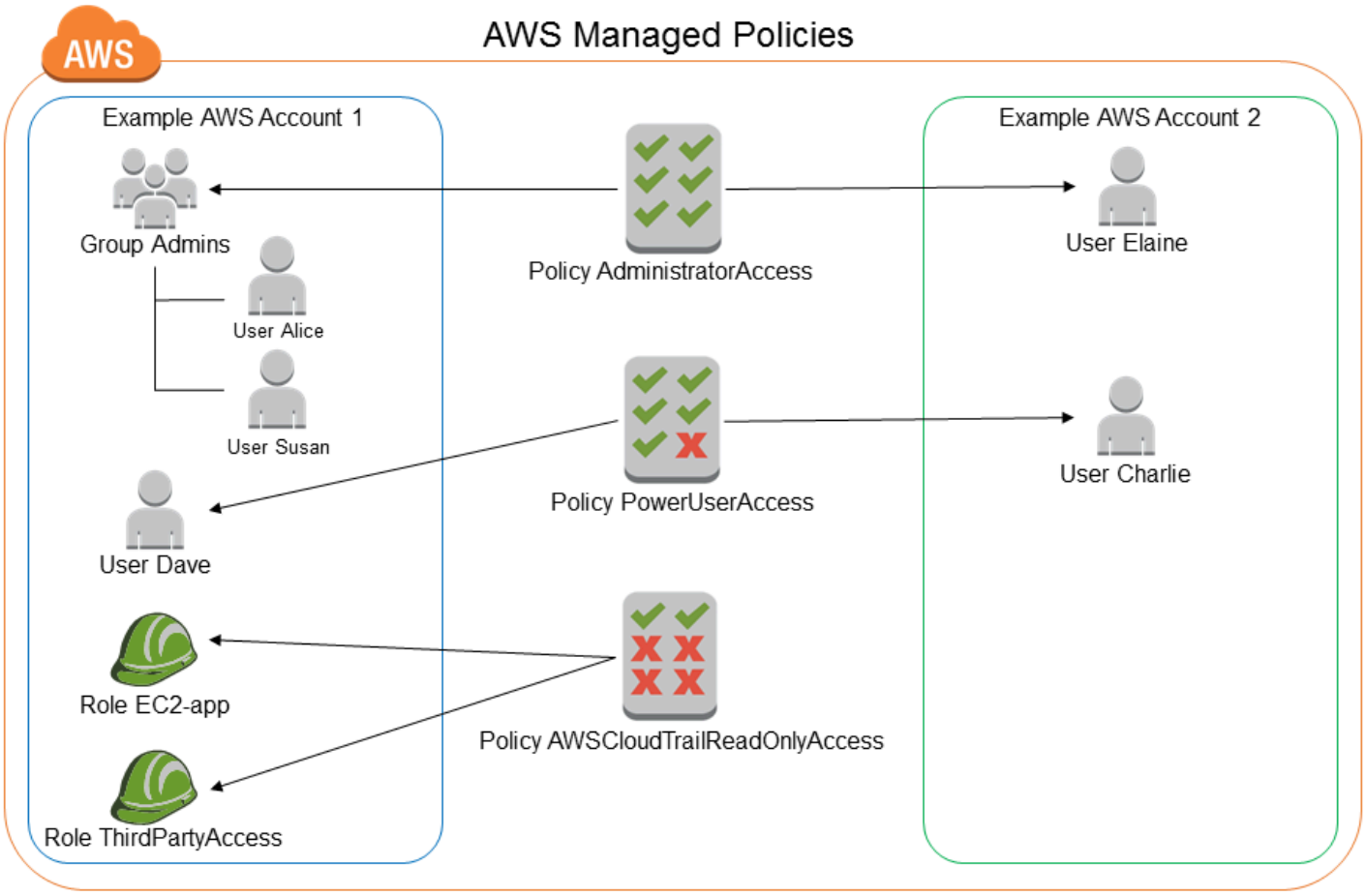
- AWSCodeCommitPowerUser
- AWSKeyManagementServicePowerUser

As políticas gerenciadas pela AWS de acesso parcial fornecem níveis específicos de acesso aos serviços da AWS sem fornecer permissões de nível de acesso ao gerenciamento de permissões.

- AmazonMobileAnalyticsWriteOnlyAccess
- AmazonEC2ReadOnlyAccess

Uma categoria especialmente útil de políticas gerenciadas pela AWS são as projetadas para 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 Organizations, use a função de trabalho PowerUserAccess. Para obter uma lista e as descrições das políticas de função de trabalho, consulte Políticas gerenciadas pela AWS 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

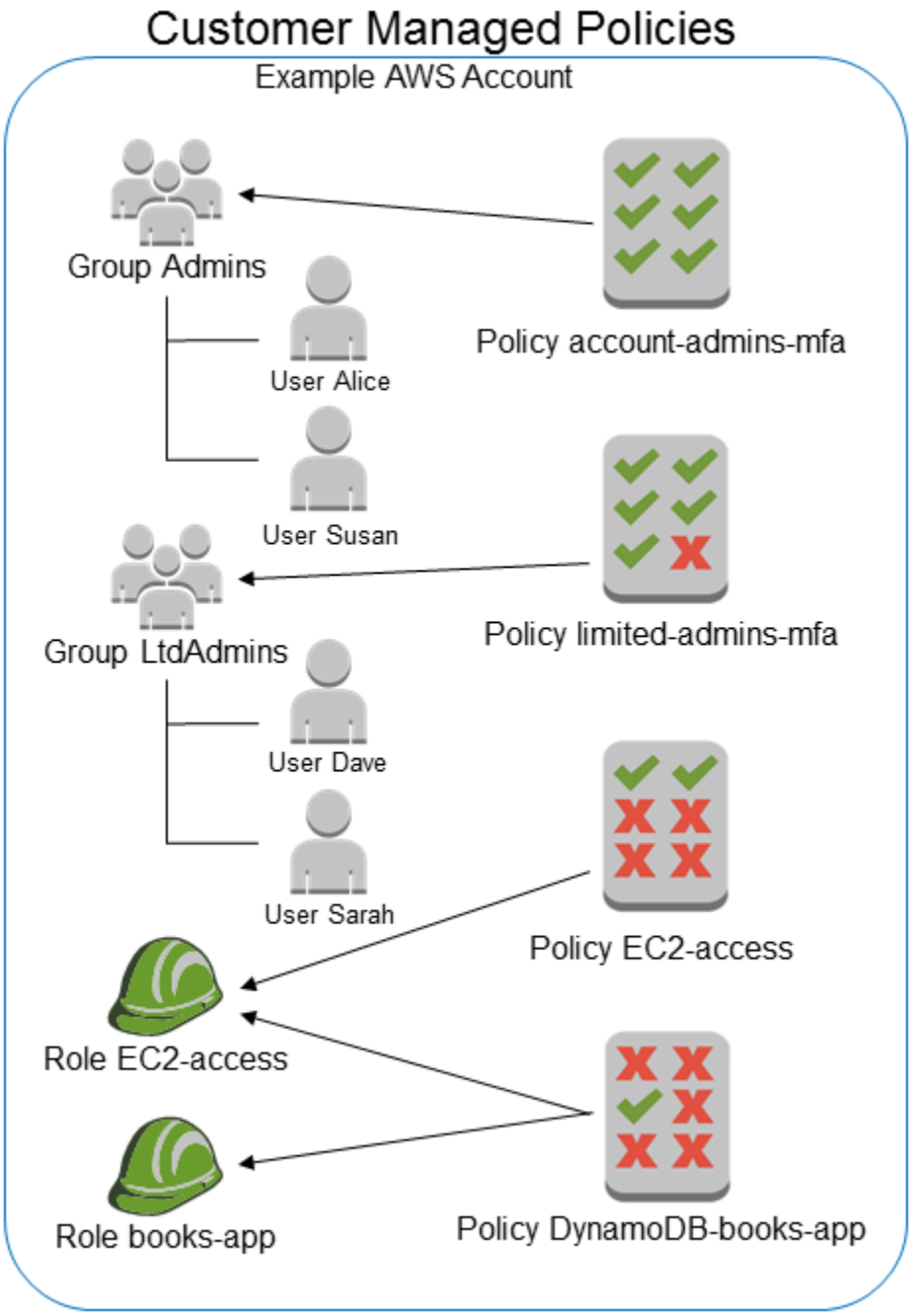
Você pode criar políticas independentes em sua Conta da AWS que podem ser anexadas a entidades principais (usuários, grupos e perfis). Crie essas políticas gerenciadas pelo cliente para seus casos de uso específicos e você poderá alterá-las e atualizá-las quantas vezes quiser. Assim como nas políticas 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 ter mais informações, consulte Criação de políticas do IAM.

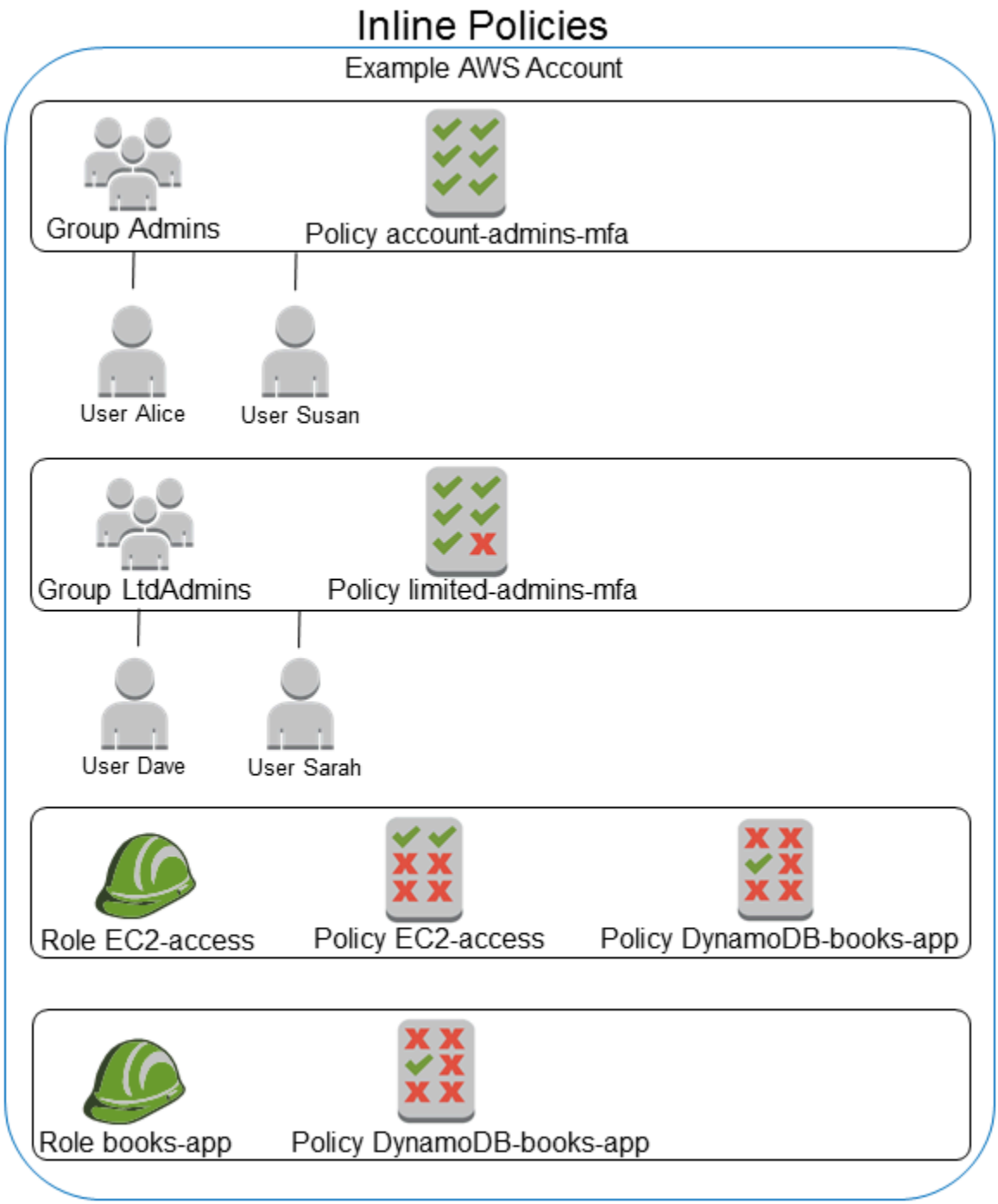


Políticas em linha

A política em linha é uma política criada para uma única identidade do IAM (um 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. 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 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 de caracteres para políticas gerenciadas é maior do que o limite de caracteres para políticas em linha. Se você atingir o limite de tamanho de 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.

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 AWS Management Console para excluir a identidade, as políticas incorporadas nela também são excluídas porque fazem parte da entidade principal.

Conceitos básicos de políticas gerenciadas

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 Políticas gerenciadas pela AWS. 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 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.

As políticas gerenciadas pela AWS são criadas para fornecer permissões para vários casos de uso comuns. Para obter mais informações sobre políticas gerenciadas pela AWSque são projetadas para funções de trabalho específicas, consulte Políticas gerenciadas pela AWS para funções de trabalho.

Para obter uma lista de políticas gerenciadas pela AWS, consulte o Guia de referência de políticas gerenciadas pela AWS.

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.

Para converter uma política em linha em uma política gerenciada

1. Faça login no AWS Management Console 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 guia Permissions (Permissões).
5. Para grupos de usuários, 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, escolha Policies (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 de usuários, marque a caixa de seleção ao lado do nome da sua nova política, escolha Add permissions (Adicionar permissões) e, em seguida, escolha Attach policy (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 Avançar 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. Uma política gerenciada defasada aparece com um ícone de aviso próximo a ela na lista Policies (Políticas) no console do IAM.

Uma política obsoleta tem as seguintes características:

- Ela continua a funcionar para todos os usuários, grupos e funções atualmente conectados. Nada é rompido.
- Não é possível anexá-la a novos usuários, grupos ou funções. Se você desanexá-la de uma entidade atual, não poderá anexá-la novamente.
- Depois de desanexá-la de todas as entidades atuais, ela não ficará mais visível e não poderá mais ser usada de nenhuma forma.

Se algum usuário, grupo ou função precisar da política, você deverá anexar a nova política. Quando você recebe um aviso de que uma política foi preterida, recomendamos que você planeje anexar imediatamente todos os usuários, grupos e funções à política substitua e desanexá-los da política depreciada. Se você continuar a usar a política depreciada, ela poderá carregar riscos que serão reduzidos apenas pela alternância para a política substituta.

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 riscos potenciais 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	O uso do	Aplicado em	Chaves de contexto de condições globais
Identidade	Somente identidades confiáveis podem acessar meus recursos	Política baseada em recurso	Recursos	aws:PrincipalOrgID aws:PrincipalOrgPaths
	Somente identidades confiáveis são permitidas na minha rede	Política de endpoint da VPC	Rede	aws:PrincipalAccount aws:PrincipalIsAwsService
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	O uso do	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:ViaAWSService aws:PrincipalIsAwsService
	Seus recursos somente podem acessados de redes esperadas	Política baseada em recurso	Recursos	

Você pode 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.

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 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 chaves de condição globais a seguir ajudam a impor controles de perímetro de identidade. Use essas chaves em políticas baseadas em recursos para restringir o acesso aos recursos ou em políticas de endpoint de VPC para restringir o acesso às suas redes.

- aws:PrincipalOrgID — Você pode 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 — Você pode usar essa chave de condição para garantir que o usuário ou perfil do IAM, o usuário federado ou AUsuário raiz da conta da AWS que fez a solicitação pertençam à unidade organizacional (OU) especificada em AWS Organizations.
- aws:PrincipalAccount — Você pode usar essa chave de condição para garantir que os recursos só possam ser acessados pela conta da entidade principal especificada na política.
- aws:PrincipalIsAWSService e aws:SourceOrgID (alternativamente aws:SourceOrgPaths e aws:SourceAccount) — Você pode usar essas chaves de condição para garantir que, quando as AWS service (Serviço da AWS)entidades principais 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 somente as suas identidades somente possam acessar recursos confiáveis e que somente recursos confiáveis possam ser acessados de sua rede. Os recursos confiáveis incluem recursos de propriedade de suas contas da AWS ou de serviços da AWS que atuam em seu nome.

As chaves de condição globais a seguir ajudam a impor controles de perímetro de recurso. 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 de VPC para restringir quais recursos podem ser acessados de suas redes.

- aws:ResourceOrgID — Você pode usar essa chave de condição para garantir que o recurso que está sendo acessado pertence à organização especificada em AWS Organizations.
- aws:ResourceOrgPaths — Você pode usar essa chave de condição para garantir que o recurso que está sendo acessado pertence à unidade organizacional especificada em AWS Organizations.
- aws:ResourceAccount — Você pode usar essa chave de condição para garantir que o recurso que está sendo acessado pertence à conta em AWS Organizations.

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 incluem seus data centers on-premises e nuvens privadas virtuais (VPCs) ou redes de serviços da AWS agindo em seu nome.

As chaves de condição globais a seguir ajudam a impor controles de perímetro de rede. Use essas chaves nas Políticas de controle de serviços (SCPs) para restringir as redes a partir das quais suas identidades podem se comunicar ou em políticas baseadas em recursos para restringir o acesso aos recursos às redes esperadas.

- aws:SourceIp — Você pode usar essa chave de condição para garantir que o endereço IP do solicitante esteja dentro de um intervalo de IP especificado.
- aws:SourceVpc — Você pode usar essa chave de condição para garantir que o endpoint da VPC pelo qual a solicitação passa pertença à VPC especificada.
- aws:SourceVpce — Você pode usar essa chave de condição para garantir que a solicitação passe pelo endpoint da VPC especificada.
- aws:ViaAWSService — Você pode 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 — Você pode usar essa chave de condição para garantir que Serviços da AWS possa acessar seus recursos usando Responsáveis pelos serviços da AWS.

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.
- Whitepaper: Construindo um perímetro de dados na AWS — Este documento descreve as práticas recomendadas e os serviços disponíveis para criar um perímetro em torno de suas identidades, recursos e redes na AWS.
- Webinar: Construindo um perímetro de dados na AWS — Saiba onde e como implementar controles de perímetro de dados com base em diferentes cenários de risco.
- 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 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 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 `ShirleyRodriguez` 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 `ShirleyRodriguez`:

```
{
    "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 `ShirleyRodriguez` 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 `ShirleyRodriguez`:

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

```
    "Effect": "Allow",
    "Action": "iam:CreateUser",
    "Resource": "*"
  }
}
```

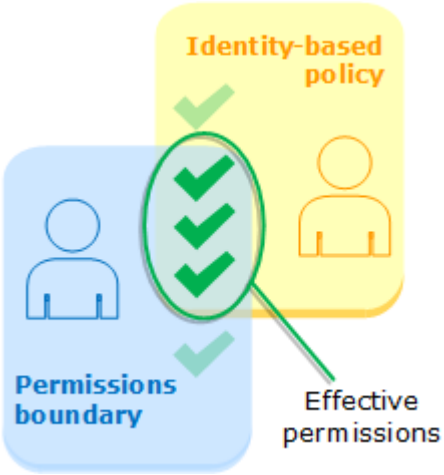
Esta política permite criar um usuário no IAM. Se você anexar essa política de permissões à usuária `ShirleyRodriguez`, e Shirley tentar criar um usuário, a operação falhará. A falha ocorre porque o limite de permissões não permite a operação `iam:CreateUser`. Dadas essas duas políticas, Shirley não 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 do Organizations ou políticas de sessão. Para obter mais informações sobre os diversos tipos diferentes de políticas, consulte Políticas e permissões no IAM.

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.

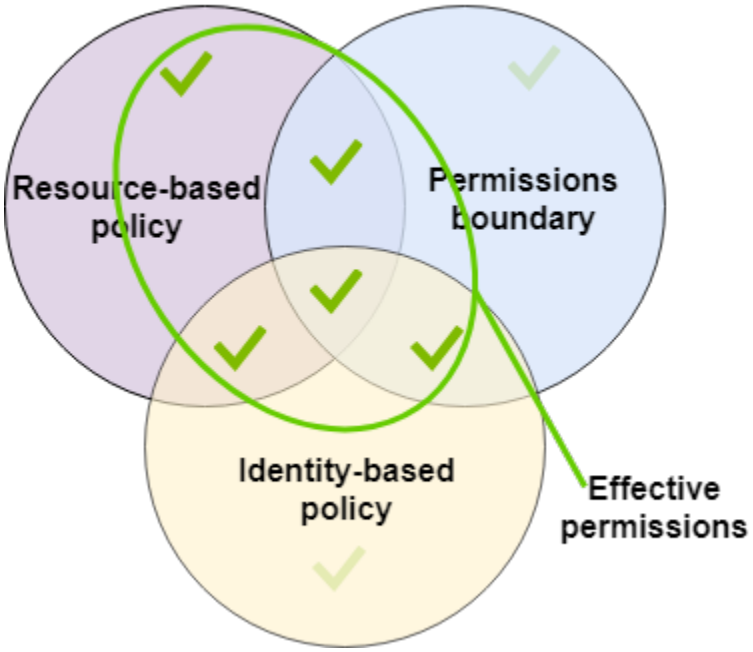
Políticas baseadas em identidade com limites: as políticas baseadas em identidade são políticas em linha ou gerenciadas que são anexadas a um usuário, grupo de usuários ou função. As políticas baseadas em identidade concedem permissão para a entidade, e os limites de permissões limitam essas permissões. 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 usuário federado) 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.

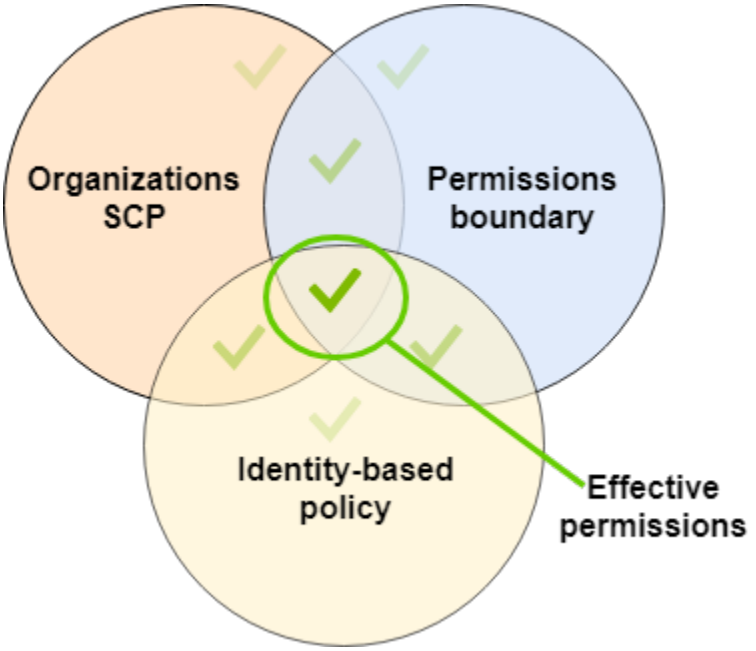
Sessão de função do IAM: na mesma conta, as políticas baseadas em recursos que concedem permissões a um ARN de sessão de função do IAM concedem permissões diretamente à sessão de função assumida. As permissões concedidas diretamente a uma sessão não são limitadas por uma negação implícita em uma política 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 usuários federados do IAM

Sessões de usuário federado do IAM: uma sessão de usuário federado do IAM 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.

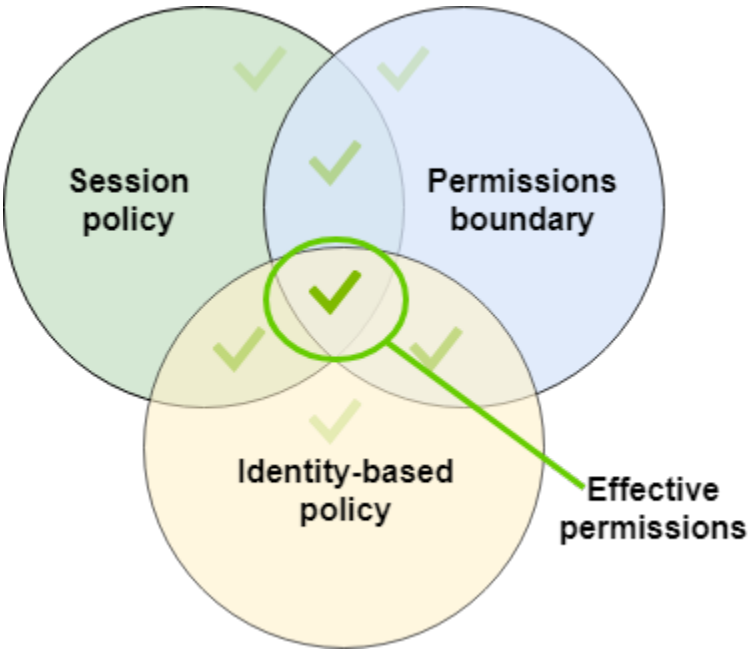
No entanto, se uma política baseada em recursos conceder permissão ao ARN do usuário do IAM que se federou, as solicitações feitas pelo usuário federado durante a sessão serão limitadas por uma negação implícita em um limite de permissão ou política de sessão.

SCPs do Organizations: as SCPs são aplicadas a uma Conta da AWS inteira. 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.



Você pode 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 a operação da API da AWS, você deve ter permissões para a ação `organizations:DescribeOrganization` da sua entidade do Organizations. Você deve ter permissões adicionais para executar a operação no console do 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 Conta da AWS da X-Company. 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

```
{
    "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 AWS Management Console. 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.

```
{
    "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.

```
{
    "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 AWS Management Console. 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. Para obter uma lista de serviços que oferecem suporte a políticas baseadas em recursos, consulte Serviços da AWS 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

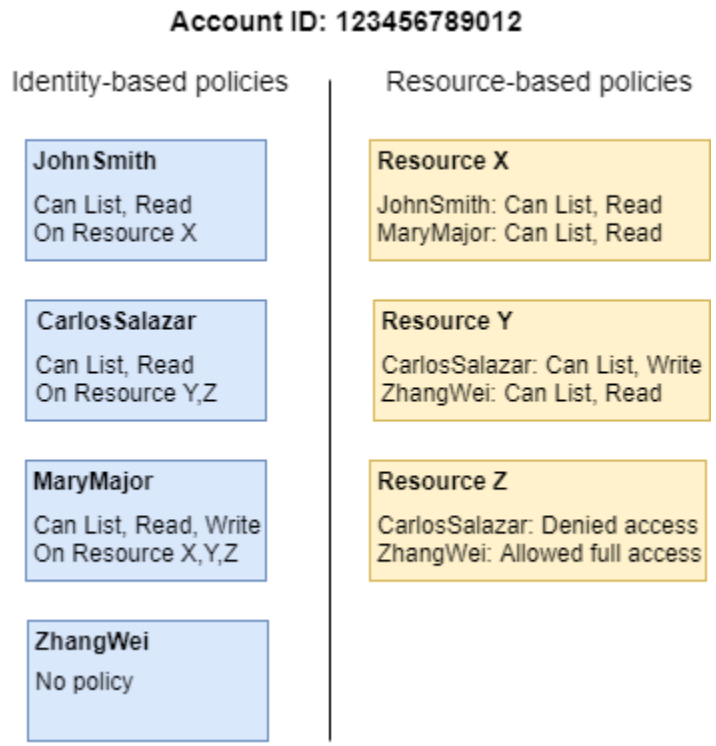
de confiança (organização confiável ou conta) têm acesso para assumir as suas funções, consulte O que é o IAM Access Analyzer?. As políticas baseadas em recursos são apenas em linha, não gerenciadas.

> **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 Serviços da AWS 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 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 `JohnSmith`, `CarlosSalazar` e `MaryMajor`. Algumas das ações nessas políticas podem ser realizadas em recursos específicos. Por exemplo, o usuário `JohnSmith` 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 `JohnSmith` permite a lista de usuários `MaryMajor` e `Resource X` e o acesso de leitura ao recurso.



O exemplo de conta `123456789012` permite que os seguintes usuários realizem as ações listadas:

- JohnSmith: John pode realizar ações de listagem e leitura em `Resource X`. Ele recebe essa permissão pela política baseada em identidade pelo usuário e a política baseada em recurso em `Resource X`.
- CarlosSalazar: Carlos pode realizar ações de listagem, leitura e gravação em `Resource Y`, mas o acesso a `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 ter mais informações, consulte Lógica da avaliação de política.
- MaryMajor: Mary pode realizar operações de listagem, leitura e gravação em `Resource X`, `Resource Y` e `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.
- ZhangWei: Zhang tem acesso total a `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.

Controle de acesso aos recursos da AWS usando 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 AWS Management Console 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.

O AWS 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 IAM.

A AWS 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 (grupos de usuários, usuários e funções) 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 seus usuários, grupos de usuários e funções, 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 `DOC-EXAMPLE-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 `DOC-`

EXAMPLE-BUCKET1 do S3. Como o limite de permissões não permite nenhuma ação do IAM, ele impede que Zhang exclua seu limite (ou de alguém).

```
{
    "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:::DOC-EXAMPLE-BUCKET1",
                "arn:aws:s3:::DOC-EXAMPLE-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 Criação de políticas do IAM.
- 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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Policies (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 usuáros 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 ter 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 ter 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. É possível controlar quem pode anexar e desvincular as políticas nas entidades de segurança (grupos de usuários, usuários e funções). 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 Criação de políticas do IAM.
- 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

```
{
  "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

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:DeletePolicyVersion",
      "iam:SetDefaultPolicyVersion"
    ],
    "Resource": "arn:aws:iam::account-id: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 anexe políticas gerenciadas somente aos grupos de usuários e às funções 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, portanto, correspondem a todos os grupos de usuários e funções 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

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id: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 anexe somente as políticas gerenciadas que incluem o caminho /TEAM-A/ apenas aos grupos de usuários e às funções 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”.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": [
      "iam:AttachGroupPolicy",
      "iam:AttachRolePolicy"
    ],
    "Resource": [
      "arn:aws:iam::account-id:group/TEAM-A/*",
      "arn:aws:iam::account-id:role/TEAM-A/*"
    ],
    "Condition": {"ArnLike":
      {"iam:PolicyARN": "arn:aws:iam::account-id: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 Criação de políticas do IAM.
- 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 ter mais informações, consulte Fornecer 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 Recursos de etiquetas do IAM.

> **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 ter 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 ter 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 ter 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 Criação de políticas do IAM.

> **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.

```
{
    "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.

```
{
    "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 Recursos de etiquetas do IAM. 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 AWS Management Console 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.

O AWS 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 IAM.

A AWS 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 Serviços da AWS 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 ResourceTag/***key-name*** caso queira permitir o acesso ao recurso com base nas tags que estão anexadas ao recurso.
- 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 Criação de políticas do IAM.

> **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 Conceder permissões a um usuário para passar uma função 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 `Owner` da instância tiver o valor do nome do usuário. Esta política define permissões para acesso programático e do console.

```
{
    "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-roe` tentar iniciar uma instância do Amazon EC2, a instância deve ser etiquetada como `0wner=richard-roe` ou `owner=richard-roe`. Caso contrário, o acesso será negado a ele. A chave de tag `0wner` corresponde a `0wner` e `owner` porque os nomes da chave de condição não fazem distinção entre maiúsculas e minúsculas. Para ter 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`.

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Sid": "AllQueueActions",
        "Effect": "Allow",
        "Action": "sqs:DeleteQueue",
        "Resource": "arn:aws:sqs:us-east-2::${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 `production` ou `preprod`. 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`.

```
{
    "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 as Tag Policies (Políticas de etiquetas) no Guia do usuário do 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 Gerenciamento de 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`.

```
{
    "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`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::APN-account-ID: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::APN-ACCOUNT-ID:role/APNPartner
```

> **Note**
>
> Recomendamos usar uma ID externa em situações multilocatárias. Para obter mais detalhes, consulte Como usar um ID externo ao conceder acesso aos seus recursos da AWS a 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 Fornecer 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, usuários federados, funções do IAM, sessões de função assumida ou produtos 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 Serviços da AWS 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 mais informações, acesse Controle de acesso 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, acesse Apêndice: A linguagem da política de acesso no Guia do desenvolvedor do Amazon Simple Queue Service.

Delegar permissões da AWS em uma política baseada em recursos

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.

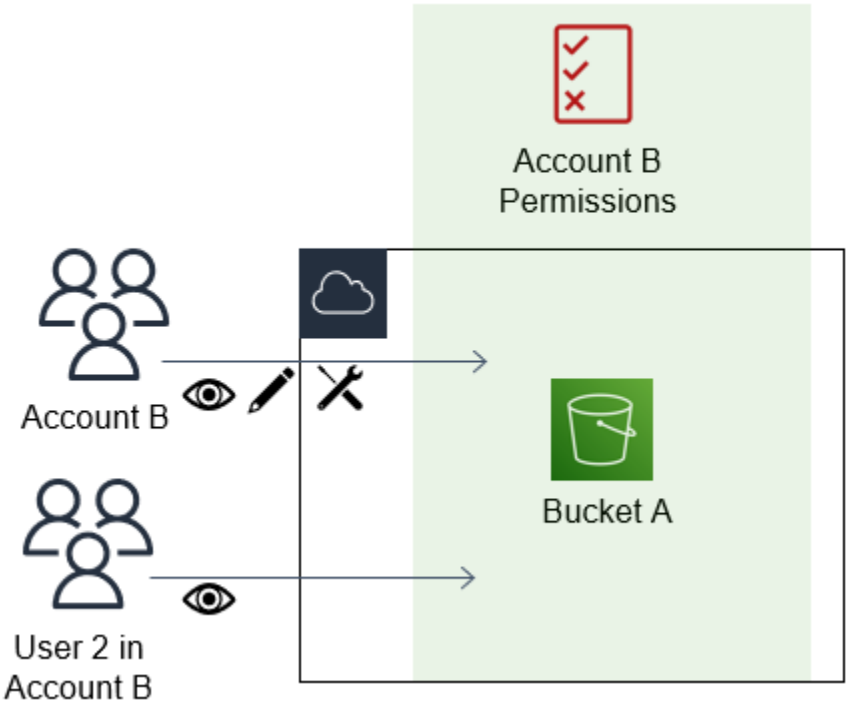
> **Important**
>
> No acesso entre contas, uma entidade principal precisa de uma `Allow` na política de identidade e uma política baseada em recursos.

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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalAccess",
            "Effect": "Allow",
            "Principal": {"AWS": "arn:aws:iam::AccountB: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.

```
{
    "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 funções: usuários, aplicações e serviços.

> **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.

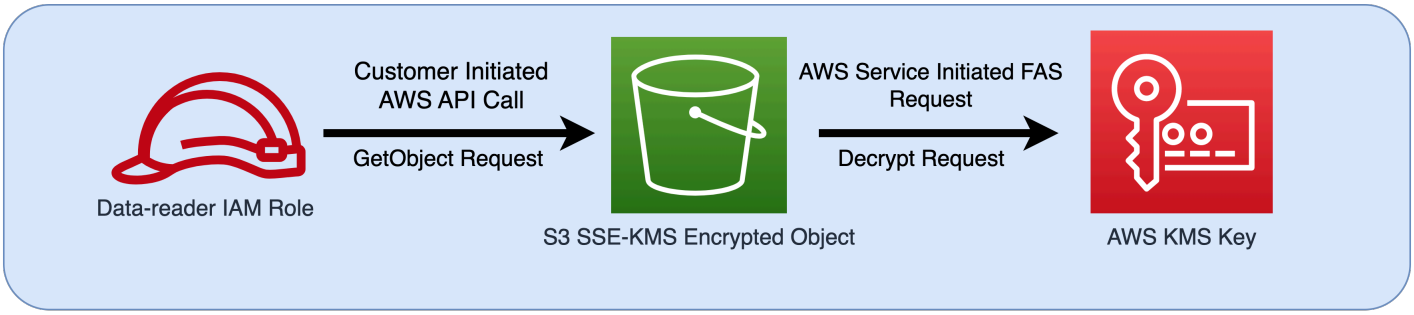
Sessões de acesso direto

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

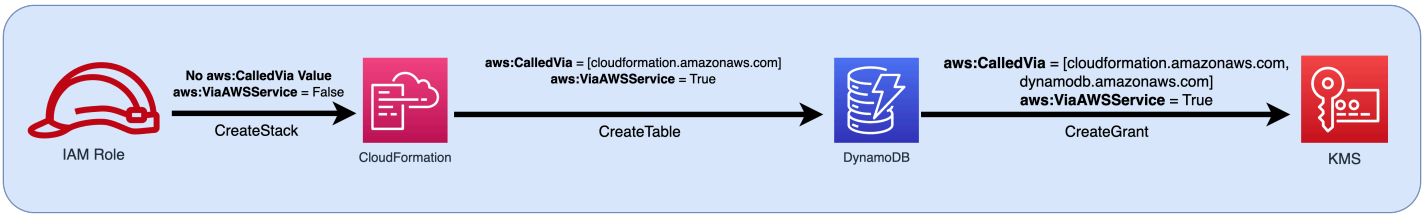
permissões corretas na chave do KMS. As solicitações para o Amazon S3 e o AWS KMS são autorizadas usando as permissões do perfil, e elas só são bem-sucedidas se o leitor de dados tem permissões tanto para o objeto do Amazon S3 quanto para a chave do AWS KMS.



> **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.

```
{
  "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": [
            "example_vpc"
```

```
          ]
        }
      }
    },
    {
      "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. Você pode 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 usuários federados acessem seu próprio diretório base do Amazon S3, de forma programática e no console (Visualize 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.

```
{
    "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 AWS Management Console. 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.

```
{
    "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 AWS Management Console 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 AWS Management Console pela primeira vez. Recomendamos que você não conceda permissões a novos usuários até que eles façam login. Para ter 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 ter mais informações, consulte Configuração de acesso à API protegido por 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`.

```
{
    "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.

```
{
    "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 AWS Management Console 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`.

```
{
    "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 AWS Management Console 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 ter mais informações, consulte Não estou autorizado a executar: iam:DeleteVirtualMFADevice.

```
{
    "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 AWS Management Console na página Credenciais de segurança. Essa página do AWS Management Console 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 `ChangeOwnPasswords`.

```
{
    "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 AWS Management Console 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`.

```
{
    "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 AWS Support 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 AWS Chatbot 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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DenyAllOutsideRequestedRegions",
            "Effect": "Deny",
            "NotAction": [
                "cloudfront:*",
                "iam:*",
                "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 Chaves de contexto de condição globais da AWS.

```
{
    "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 `DenyAllS3ResoucesOutsideAccountExceptDataExchange` 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.

```
{
  "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.

```
{
    "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.

```
{
    "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.

```
{
    "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.

```
{
    "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 Controlar o acesso aos recursos do Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux.

```
{
    "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.

```
{
    "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.

```
{
   "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:region:111122223333:security-group/*"
         ],
         "Condition": {
            "StringEquals": {
               "aws:ResourceTag/Department": "Test"
            }
         }
      },
      {
         "Effect": "Allow",
         "Action": [
            "ec2:ModifySecurityGroupRules"
         ],
         "Resource": [
            "arn:aws:ec2:region: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.

```
{
    "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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "StartStopIfTags",
            "Effect": "Allow",
            "Action": [
                "ec2:StartInstances",
                "ec2:StopInstances"
```

```
            ],
            "Resource": "arn:aws:ec2:region:account-id: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.

```
{
    "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.

```
{
    "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.

```
{
  "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 `TokenCode` e `SerialNumber` ao chamar a operação `GetSessionToken`. Essa operação retorna credenciais temporárias que foram autenticadas usando

MFA. Para saber mais sobre GetSessionToken, consulte GetSessionToken: credenciais temporárias 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 `TerminateInstances` e `StopInstances` 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.

```
{
    "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 the section called “Determinar se uma solicitação é permitida ou negada em uma conta”.

> **Important**
>
> A chave de condição `aws:SourceIp` nega acesso a um 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 Chaves de contexto de condição globais da AWS.

```
{
    "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.

```
{
    "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 Conta da AWS atual. Esta política concede as permissões necessárias para concluir esta ação na API ou AWS CLI da AWS de maneira programática.

Você pode acessar o console do simulador de política do IAM em: https://policysim.aws.amazon.com/

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:GetGroup",
                "iam:GetGroupPolicy",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetUser",
                "iam:GetUserPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListAttachedRolePolicies",
                "iam:ListAttachedUserPolicies",
                "iam:ListGroups",
                "iam:ListGroupPolicies",
                "iam:ListGroupsForUser",
                "iam:ListRolePolicies",
                "iam:ListRoles",
                "iam:ListUserPolicies",
                "iam:ListUsers"
            ],
            "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 Fornecer acesso a um usuário do IAM em outra Conta da AWS de sua propriedade.

```
{
    "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 AWS Management Console. 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.

```
{
    "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 Chaves de contexto de múltiplos valores.

```
{
    "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 Chaves de contexto de múltiplos valores.

```
{
```

```
    "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 `JobFunction` ou `Department` 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 Chaves de contexto de múltiplos valores.

```
{
    "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 credencial, consulte Obter relatórios de credenciais da sua Conta da AWS.

```
{
    "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 AWS Management Console. 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`.

```
{
    "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.

```
{
    "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.

Uma função de serviço é uma função do IAM que especifica um produto da AWS como a entidade de segurança que pode assumir a função. Isso permite que o serviço assuma a função e acesse recursos em outros serviços em seu nome. Para permitir que o Amazon CloudWatch assuma a função que você passa, é necessário especificar a entidade de segurança de serviço `cloudwatch.amazonaws.com` como a entidade de segurança na política de confiança de sua

função. 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 Serviços da AWS 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 da função vinculada a 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 Conceder permissões a um usuário para passar uma função para um serviço da AWS.

```
{
    "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.

```
{
```

```
    "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, consulteIAM: permite acesso somente leitura ao console do IAM sem relatórios.

```
{
    "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 instrução `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.

```
{
    "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 Definição de uma política de senhas de contas para usuários do IAM.

```
{
    "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.

```
{
    "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`. 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.

Você pode acessar o simulador de política do IAM em: https://policysim.aws.amazon.com/

```
{
    "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.

```
{
    "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.

```
{
    "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 as informações do último acesso ao serviço para uma política do 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 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. Esta política permite que o solicitante recupere os dados de qualquer entidade do 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 as informações do último acesso.

```
{
    "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.

```
{
    "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 `AllowAccessToS3ResourcesInSpecificAccountsAndSpecificService1` na política.

```
{
  "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.

```
{
    "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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "rds:*",
            "Resource": ["arn:aws:rds:region:*:*"]
        },
        {
            "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.

```
{
    "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.

```
{
    "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 S3. Esta política permite acesso apenas a objetos com um nome que inclua `cognito`, o nome da aplicação e o ID de usuário federado 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.

```
{
  "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 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 federados 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 federados 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 do usuário federado é um identificador exclusivo atribuído à função 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 o usuário federado assumiu sua função.

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 do usuário federado é `AROAXXT2NJT7D3SIQN7Z6:John`. Esta política permite que o usuário federado John acesse o bucket do Amazon S3 com o prefixo `AROAXXT2NJT7D3SIQN7Z6:John`.

```
{
    "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:::bucket-name",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:userid}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name/home/${aws:userid}",
                "arn:aws:s3:::bucket-name/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 `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 `Production` 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 `Production` de forma programática, o administrador do S3 deve usar credenciais temporárias geradas nos últimos 30 minutos usando a operação de API GetSessionToken.

```
{
    "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:::Production/*",
                "arn:aws:s3:::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.

```
{
    "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:::bucket-name",
            "Condition": {
                "StringLike": {
                    "s3:prefix": [
                        "",
                        "home/",
                        "home/${aws:username}/*"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::bucket-name/home/${aws:username}",
                "arn:aws:s3:::bucket-name/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 the section called "Determinar se uma solicitação é permitida ou negada em uma conta".

> **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.

```
{
    "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/*"
            ]
        }
    ]
}
```

Amazon S3: permite acesso de leitura e gravação a objetos em um bucket do 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 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).

```
{
    "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).

```
{
    "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:::bucket-name"]
        },
        {
            "Sid": "AllObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object",
            "Resource": ["arn:aws:s3:::bucket-name/*"]
        }
```

```
    ]
}
```

Gerenciamento de 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.

Consulte estes recursos para obter mais detalhes:

- Para obter mais informações sobre os diferentes tipos de políticas do IAM, consulte Políticas e permissões no IAM.
- Para obter informações gerais sobre o uso de políticas no IAM, consulte Gerenciamento de acesso para recursos da AWS.
- 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 obter mais informações, consulte IAM e cotas do AWS STS.

Tópicos

Criação de políticas do IAM

Uma política é uma entidade que, quando anexada a uma identidade ou recurso, define suas permissões. Você pode usar o AWS Management Console, a AWS CLI ou 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. 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 é possível gerenciá-las. Uma política em linha é aquela que você cria e incorpora diretamente em um usuário, grupo ou função 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.

Como utilizar políticas gerenciadas pelo cliente em vez de políticas em linha. Também é melhor usar políticas gerenciadas pelo cliente em vez de políticas gerenciadas pela AWS. As políticas gerenciadas pela AWS geralmente fornecem amplas permissões administrativas ou somente leitura. Para maior segurança, conceda menos privilégios, que concedem 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 AWS Management Console, 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 políticas 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 AWS Management Console, 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 AWS CloudFormation para adicionar ou atualizar políticas, consulte a Referência a tipos de recursos do AWS Identity and Access Management, no Guia do usuário do AWS CloudFormation.

Tópicos

- Criar políticas do IAM (console)

- Criação de políticas do IAM (AWS CLI)
- Como 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 AWS Management Console 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.

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 AWS Management Console 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 IAM. 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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.
3. Escolha Create policy (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 Next (Avançar).

   > **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 AWS Management Console, você pode gerar um modelo de política JSON ou YAML que pode ser usado em modelos do AWS CloudFormation.

Para isso, no editor de políticas, escolha Ações e depois escolha Gerar modelo do CloudFormation. Para saber mais sobre o AWS 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 etiquetas no IAM, consulte Recursos de etiquetas do IAM.
11. Escolha Create Policy (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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.
3. Escolha Create policy (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 as 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 AWS Management Console, você pode gerar um modelo de política JSON ou YAML que pode ser usado em modelos do AWS CloudFormation.

Para isso, no editor de políticas, escolha Ações e depois escolha Gerar modelo do CloudFormation. Para saber mais sobre o AWS 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 etiquetas no IAM, consulte Recursos de etiquetas do IAM.
13. Escolha Create Policy (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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.
3. Escolha Create policy (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. Em seguida, escolha Next (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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, escolha Políticas.
3. Escolha Create policy (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, erros ou avisos gerais gerados durante a validação de política e depois escolha Next (Avançar). Ou personalize sua política no Visual editor (Editar visual). Em seguida, escolha Next (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.

Criação de 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 obter 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

Como 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 políticas 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 AWS Management Console, 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 AWS Management Console. 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 AWS Management Console 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.

Gerar políticas com base na atividade de acesso

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.

Por exemplo, imagine que você é um desenvolvedor e sua equipe de engenharia tem trabalhado em um projeto para criar uma nova aplicação. Para incentivar a experimentação e permitir que a equipe avance rapidamente, você configurou uma função com permissões amplas enquanto a aplicação

está em desenvolvimento. Agora, a aplicação está pronta para produção. Antes de lançar a aplicação na conta de produção, você deseja identificar apenas as permissões que a função precisa para que a aplicação funcione. Isso ajuda a ter maior aderência às práticas recomendadas para conceder privilégio mínimo. Você pode gerar uma política com base na atividade de acesso da função que tem usado para a aplicação na conta de desenvolvimento. E pode refinar ainda mais a política gerada e anexá-la a uma entidade em sua conta de produção.

Para saber mais sobre a geração de políticas do IAM Access Analyzer, consulte Geração de políticas do IAM Access Analyzer.

Testar as políticas do IAM com o simulador de políticas do IAM

Para obter mais informações sobre como e por que usar políticas do IAM, consulte Políticas e permissões no IAM.

Você pode acessar o console do simulador de política do IAM em: https://policysim.aws.amazon.com/

> **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 usando o simulador de políticas para confirmar que obteve os resultados desejados. Para obter mais informações, consulte Como o simulador de políticas do IAM funciona.

Conceitos básicos do simulador de políticas do IAM

Com o simulador de políticas do IAM, é possível testar e solucionar problemas de políticas baseadas em identidade e limites de permissões do IAM. Veja algumas ações comuns que você pode executar com o simulador de políticas:

- Teste as políticas baseadas em identidade anexadas a usuários, grupos de usuários ou perfis do IAM em sua Conta da AWS. Se mais de uma política estiver anexada ao usuário, grupo de usuários ou função, você pode testar todas as políticas ou selecionar políticas individuais para testar. Você pode testar quais ações são permitidas ou negadas pelas políticas selecionadas para recursos específicos.
- Teste e solucione o problema do efeito dos limites de permissões em entidades do IAM. É possível simular somente um limite de permissões por vez.

- Teste os efeitos de políticas baseadas em recursos em usuários do IAM que estão anexadas aos recursos da AWS, como buckets do Amazon S3, filas do Amazon SQS, tópicos do Amazon SNS ou cofres do Amazon S3 Glacier. Para usar uma política baseada em recursos no simulador de políticas para usuários do IAM, você deve incluir o recurso na simulação. Você também deve marcar a caixa de seleção para incluir a política desse recurso na simulação.

  > **Note**
  >
  > A simulação de políticas baseadas em recursos não é compatível com perfis do IAM.

- Se a sua Conta da AWS for membro de uma organização no AWS Organizations, você poderá testar o impacto das políticas de controle de serviços (SCPs) em suas políticas baseadas em identidade.

  > **Note**
  >
  > O simulador de políticas não avalia SCPs que tenham alguma condição.

- Teste novas políticas baseadas em identidade que ainda não foram anexadas a um usuário, grupo de usuários ou perfil digitando-as ou copiando-as no simulador. Elas são usadas apenas na simulação e não são salvas. Não é possível digitar nem copiar uma política baseada em recursos no simulador de políticas.
- Teste as políticas baseadas em identidade com serviços, ações e recursos selecionados. Por exemplo, você pode testar para garantir que sua política permita que uma entidade execute as ações `ListAllMyBuckets`, `CreateBucket` e `DeleteBucket` no serviço Amazon S3 em um bucket específico.
- Simule cenários reais fornecendo chaves de contexto, como um endereço IP ou uma data, que estão incluídas em elementos `Condition` nas políticas que estão sendo testadas.

  > **Note**
  >
  > O simulador de políticas não simulará as etiquetas fornecidas como entrada se a política baseada em identidade na simulação não tiver um elemento `Condition` que verifique explicitamente as etiquetas.

- Identifique a instrução específica em uma política baseada em identidade que resulta na permissão ou negação de acesso a determinado recurso ou ação.

Tópicos

Como o simulador de políticas do IAM funciona

O simulador de políticas avalia as declarações na política baseada em identidade e as entradas fornecidas durante a simulação. 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 usando o simulador de políticas para confirmar que obteve os resultados desejados.

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 podem fazer alterações indesejadas em seu ambiente de produção da AWS. O simulador de políticas não leva em consideração os principais valores do contexto real na produção.
- Como o simulador de políticas não simula a execução das ações selecionadas, ele não pode relatar nenhuma resposta à solicitação simulada. O único resultado retornado é se a ação solicitada seria permitida ou negada.
- Se você editar uma política no simulador, essas alterações afetarão apenas o simulador de políticas. A política correspondente em sua Conta da AWS permanecerá inalterada.
- Não é possível testar políticas de controle de serviço (SCPs) com qualquer condição.
- O simulador de políticas não oferece suporte à simulação de perfis e usuários do IAM para acesso entre contas.

> **Note**
>
> O simulador de políticas do IAM não determina quais serviços oferecem suporte a chaves de condição globais para autorização. Por exemplo, o simulador de políticas não identifica quando um serviço não oferece suporte a `aws:TagKeys`.

Permissões necessárias para usar o simulador de políticas do IAM

Você pode usar o console ou a API do simulador de políticas para testar políticas. Por padrão, os usuários do console podem testar políticas que ainda não foram anexadas a um usuário, grupo de usuários ou perfil digitando-as ou copiando-as no console do simulador de políticas. Essas políticas são usadas apenas na simulação e não divulgam informações confidenciais. Os usuários da API devem ter permissões para testar políticas não anexadas. Você pode permitir que usuários do console ou da API testem as políticas anexadas a usuários, grupos de usuários ou perfis do IAM na sua Conta da AWS. Para fazer isso, você deve conceder a permissão para recuperar essas políticas. Para testar políticas baseadas em recursos, os usuários devem ter permissão para recuperar a política do recurso.

Para obter exemplos de políticas do console e da API que permitem a um usuário simular políticas, consulte the section called “Exemplo de políticas do AWS Identity and Access Management (IAM)”.

Permissões necessárias para usar o console do simulador de políticas

Você pode permitir que os usuários testem políticas anexadas a usuários, grupos de usuários ou perfis do IAM na sua Conta da AWS. Para fazer isso, você deve conceder aos usuários permissões para recuperar essas políticas. Para testar políticas baseadas em recursos, os usuários devem ter permissão para recuperar a política do recurso.

Para visualizar uma política de exemplo que permita usar o console do simulador de políticas para políticas anexadas a um usuário, grupo de usuários ou função, consulte IAM: acessar o console do simulador de política.

Para visualizar um exemplo de política que permita o uso do console do simulador de políticas somente para os usuários com um caminho específico, consulte IAM: acessar o console do simulador de políticas com base no caminho do usuário.

Para criar uma política para permitir o uso do console do simulador de políticas para apenas um tipo de entidade, use os seguintes procedimentos.

Para permitir que os usuários do console simulem políticas para os usuários

Inclua as seguintes ações em sua política:

- `iam:GetGroupPolicy`
- `iam:GetPolicy`

- iam:GetPolicyVersion
- iam:GetUser
- iam:GetUserPolicy
- iam:ListAttachedUserPolicies
- iam:ListGroupsForUser
- iam:ListGroupPolicies
- iam:ListUserPolicies
- iam:ListUsers

Para permitir que os usuários do console simulem políticas para grupos de usuários

Inclua as seguintes ações em sua política:

- iam:GetGroup
- iam:GetGroupPolicy
- iam:GetPolicy
- iam:GetPolicyVersion
- iam:ListAttachedGroupPolicies
- iam:ListGroupPolicies
- iam:ListGroups

Para permitir que os usuários do console simulem políticas para funções

Inclua as seguintes ações em sua política:

- iam:GetPolicy
- iam:GetPolicyVersion
- iam:GetRole
- iam:GetRolePolicy
- iam:ListAttachedRolePolicies
- iam:ListRolePolicies
- iam:ListRoles

Para testar políticas baseadas em recursos, os usuários devem ter permissão para recuperar a política do recurso.

Para permitir que usuários do console testem políticas baseadas em recurso em um bucket do Amazon S3

Inclua a seguinte ação em sua política:

- `s3:GetBucketPolicy`

Por exemplo, a política a seguir usa essa ação para permitir que os usuários do console simulem uma política baseada em recurso em um bucket específico do Amazon S3.

```
{
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": "s3:GetBucketPolicy",
            "Resource":"arn:aws:s3:::bucket-name/*"
          }
        ]
      }
```

Permissões necessárias para usar a API do simulador de políticas

As operações de API do simulador de políticas GetContextKeyForCustomPolicy e SimulateCustomPolicy permitem testar as políticas que ainda não estão anexadas a um usuário, um grupo de usuários ou uma função. Para testar essas políticas, passe as políticas como strings para a API. Essas políticas são usadas apenas na simulação e não divulgam informações confidenciais. Também pode usar a API para testar políticas que estejam anexadas a usuários, grupos de usuários ou perfis do IAM na sua Conta da AWS. Para fazer isso, é necessário conceder aos usuários permissões para chamar GetContextKeyForPrincipalPolicy e SimulatePrincipalPolicy.

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 criar uma política para permitir o uso da API do simulador de políticas para apenas um tipo de política, use os seguintes procedimentos.

Para permitir que os usuários da API simulem políticas transmitidas diretamente para a API como strings

Inclua as seguintes ações em sua política:

- `iam:GetContextKeysForCustomPolicy`
- `iam:SimulateCustomPolicy`

Para permitir que os usuários da API simulem as políticas anexadas a usuários, grupos de usuários, funções ou recursos do IAM

Inclua as seguintes ações em sua política:

- `iam:GetContextKeysForPrincipalPolicy`
- `iam:SimulatePrincipalPolicy`

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 seguinte recurso: `arn:aws:iam::777788889999:user/alice`.

Para visualizar um exemplo de política que permita o uso da API do simulador de políticas somente para os usuários com um caminho específico, consulte IAM: acessar a API do simulador de política com base no caminho do usuário.

Uso do simulador de políticas do IAM (console)

Por padrão, os usuários podem testar políticas que ainda não foram anexadas a um usuário, grupo de usuários ou função digitando-as ou copiando-as no console do simulador de políticas. Essas políticas são usadas apenas na simulação e não divulgam informações confidenciais.

Como testar uma política que não está anexada a um usuário, grupo de usuários ou uma função (console)

1. Abra o console do simulador de política do IAM em: https://policysim.aws.amazon.com/.
2. No menu Modo: na parte superior da página, escolha Nova política.
3. Na Sandbox de políticas, selecione Criar nova política.
4. Digite ou copie uma política no simulador de políticas e use o simulador de políticas, como descrito nas etapas a seguir.

Depois de ter permissão para usar o console do simulador de políticas do IAM, você poderá usar o simulador de políticas para testar um usuário, um grupo de usuários, um perfil ou uma política de recursos do IAM.

Para testar uma política que está anexada a um usuário, grupo de usuários ou função (console)

1. Abra o console do simulador de política do IAM em https://policysim.aws.amazon.com/.

   > **Note**
   >
   > Para fazer login no simulador de políticas como um usuário do IAM, use o URL de login exclusivo para fazer login no AWS Management Console. Em seguida, vá para https://policysim.aws.amazon.com/. Para obter mais informações sobre como fazer login como usuário do IAM, consulte Como os usuários do IAM fazem login na AWS.

   O simulador de políticas é aberto no modo Existing Policies (Políticas existentes) e lista os usuários do IAM em sua conta em Users, Groups, and Roles (Usuários, grupos e perfis).

2. Escolha a opção apropriada para sua tarefa:

Para testar isso:	Faça o seguinte:
Uma política anexada a um usuário	Escolha Usuários na lista Usuários, grupos e funções. Em seguida, escolha o usuário.
Uma política anexada a um grupo de usuários	Escolha Grupos na lista Usuários, grupos e funções. Em seguida, escolha o grupo de usuários.
Uma política anexada a uma função	Escolha Funções na lista Usuários, grupos e funções. Em seguida, escolha a função.
Uma política anexada a um recurso	Consulte Step 9.

Para testar isso:	Faça o seguinte:
Uma política personalizada para um usuário, um grupo de usuários ou uma função	Selecione Create New Policy (Criar nova política). No novo painel Policies (Políticas), digite ou cole uma política e escolha Apply (Aplicar).

> **Dica**
>
> Para testar uma política anexada ao grupo de usuários, você pode iniciar o simulador de políticas do IAM diretamente do console do IAM: no painel de navegação, escolha User groups (Grupos de usuários). Escolha o nome do grupo no qual você deseja testar uma política e, em seguida, selecione a guia Permissões. Escolha Simulate (Simular).
> Para testar uma política gerenciada pelo cliente que esteja anexada a um usuário: no painel de navegação, escolha Usuários. Escolha o nome do usuário no qual deseja testar uma política. Em seguida, selecione a guia Permissões e expanda a política que você deseja testar. À direita, escolha Simular política. O IAM Policy Simulator (Simulador de políticas do IAM) é aberto em uma nova janela e exibe a política selecionada no painel Policies (Políticas).

3. (Opcional) Se sua conta for membro de uma organização no AWS Organizations, marque a caixa de seleção ao lado de AWS OrganizationsSCPs (SCPs do AWS Organizations) para incluir SCPs em sua avaliação simulada. SCPs são políticas JSON que especificam o máximo de permissões para uma organização ou unidade organizacional (UO). O SCP limita as permissões para entidades em contas-membro. Se um bloco de SCP bloquear um serviço ou uma ação, nenhuma entidade na conta em questão poderá acessar esse serviço nem executar essa ação. Isso é verdadeiro mesmo que um administrador conceda permissões explicitamente a esse serviço ou ação por meio de uma política de recurso ou do IAM.

   Se sua conta não for membro de uma organização, a caixa de seleção não será exibida.

4. (Opcional) Você também pode testar uma política definida como um limite de permissões para uma entidade do (usuário ou função) do IAM, mas não para grupos de usuários. Se uma política de limite de permissões estiver atualmente definida para a entidade, ela aparecerá no painel Policies (Políticas) . É possível definir apenas um limite de permissões para uma entidade. Para testar um limite de permissões diferente, é possível criar um limite de permissões personalizado.

Para fazer isso, escolha Create New Policy (Criar nova política). Um novo painel Policies (Políticas) é aberto. No menu, escolha Custom IAM Permissions Boundary Policy (Política de limite de permissões do IAM personalizado). Insira um nome para a nova política e digite ou copie uma política no espaço abaixo. Escolha Apply (Aplicar) para salvar a política. Depois, escolha Back (Voltar) para retornar ao painel Policies (Políticas) original. Marque a caixa de seleção ao lado do limite de permissões que deseja usar para a simulação.

5. (Opcional) É possível testar apenas um subconjunto de políticas anexadas a um usuário, grupo de usuários ou função. Para fazer isso, no painel Policies (Políticas), desmarque a caixa de seleção ao lado de cada política que deseja excluir.
6. Em Simulador de políticas, escolha Selecionar serviço e, em seguida, o serviço a ser testado. Em seguida, escolha Selecionar ações e escolha uma ou mais ações a serem testadas. Embora os menus mostrem as seleções disponíveis somente para um serviço por vez, todos os serviços e ações que você selecionou aparecerão em Configurações e resultados da ação.
7. (Opcional) Se alguma das políticas que você escolher em Step 2 e Step 5 incluir condições com chaves de condições globais da AWS, forneça valores para essas chaves. Você pode fazer isso expandindo a seção Configurações globais e digitando valores para os nomes de chaves exibidos.

   > **Warning**
   >
   > Se você deixar o valor para uma chave de condição vazio, essa chave será ignorada durante a simulação. Em alguns casos, isso resulta em um erro e a simulação não é executada. Em outros casos, a simulação é executada, mas os resultados talvez não sejam confiáveis. Nesses casos, a simulação não corresponde às condições reais que incluem um valor para a chave de condição ou a variável.

8. (Opcional) Cada ação selecionada aparece na lista Configurações e resultados da ação com Não simulado mostrado na coluna Permissão até que você realmente execute a simulação. Antes de executar a simulação, você pode configurar cada ação com um recurso. Para configurar ações individuais para um cenário específico, escolha a seta para expandir a linha da ação. Se a ação for compatível com permissões no nível do recurso, você poderá digitar o nome de recurso da Amazon (ARN) do recurso específico cujo acesso você deseja testar. Por padrão, cada recurso é definido como um caractere curinga (*). Você também pode especificar um valor para qualquer chave de contexto de condição. Conforme mencionado anteriormente, as chaves com valores vazios são ignoradas, o que pode causar falhas na simulação ou resultados não confiáveis.

a. Escolha a seta ao lado do nome da ação para expandir cada linha e configure todas as informações adicionais necessárias para simular de forma precisa a ação em seu cenário. Se a ação exigir permissões no nível do recurso, você poderá digitar o nome de recurso da Amazon (ARN) do recurso específico ao qual deseja simular o acesso. Por padrão, cada recurso é definido como um caractere curinga (*).

b. Se a ação der suporte às permissões no nível do serviço, mas não precisar delas, selecione Adicionar recurso para selecionar o tipo de recurso que você deseja adicionar à simulação.

c. Se qualquer uma das políticas selecionadas incluir um elemento `Condition` que faz referência a uma chave de contexto para o serviço dessa ação, esse nome de chave será exibido abaixo da ação. Você pode especificar o valor a ser usado durante a simulação dessa ação para o recurso especificado.

Ações que exigem grupos de tipos de recursos diferentes

Algumas ações exigem tipos de recursos diferentes em circunstâncias diferentes. Cada grupo de tipos de recursos está associado a um cenário. Se um deles se aplicar a sua simulação, selecione-o, e o simulador de política exigirá os tipos de recursos apropriados para esse cenário. A lista a seguir mostra cada uma das opções de cenário com suporte e os recursos que você deve definir para executar a simulação.

Cada um dos seguintes cenários do Amazon EC2 a seguir exige que você especifique os recursos `instance`, `image` e `security-group`. Se o seu cenário incluir um volume do EBS, especifique esse `volume` como um recurso. Se o cenário do Amazon EC2 incluir uma Virtual Private Cloud (VPC), forneça o recurso `network-interface`. Se ele incluir uma sub-rede IP, especifique o recurso `subnet`. Para obter mais informações sobre as opções de cenário do Amazon EC2, consulte Plataformas com suporte no Guia do usuário do Amazon EC2.

- EC2-VPC-InstanceStore

  instância, imagem, security-group, interface de rede

- EC2-VPC-InstanceStore-Subnet

  instância, imagem, security-group, interface de rede, sub-rede

- EC2-VPC-EBS

  instância, imagem, security-group, interface de rede, volume

- EC2-VPC-EBS-Subnet

  instância, imagem, security-group, interface de rede, sub-rede, volume

9. (Opcional) Se você deseja incluir uma política com base em recursos em sua simulação, primeiro selecione as ações que você deseja simular nesse recurso em Step 6. Expanda as linhas das ações selecionadas e digite o nome de recurso da Amazon (ARN) do recurso com uma política que você deseja simular. Em seguida, selecione Incluir política de recursos ao lado da caixa de texto nome de recurso da Amazon (ARN). O simulador de políticas do IAM atualmente oferece suporte a políticas baseadas em recurso apenas dos seguintes serviços: Amazon S3 (apenas políticas baseadas em recurso; ACLs não são compatíveis no momento), Amazon SQS, Amazon SNS e armazenamentos desbloqueados do S3 Glacier (armazenamentos bloqueados não têm suporte no momento).
10. Escolha Executar simulação no canto superior direito.

    A coluna Permissão em cada linha de Configurações e resultados da ação exibe o resultado da simulação dessa ação no recurso especificado.
11. Para ver qual instrução em uma política permitiu ou negou explicitamente uma ação, selecione o link ***N*** instruções correspondentes na coluna Permissões para expandir a linha. Em seguida, escolha o link Mostrar instrução. O painel Política mostra a política relevante com a instrução que afetou o resultado realçado da simulação.

> **Note**
>
> Se uma ação for implicitamente negada, ou seja, se a ação for negada apenas porque não é explicitamente permitida, as opções List (Listar) e Mostrar instrução não são exibidas.

Solução de problemas em mensagens do console do simulador de políticas do IAM

A tabela a seguir lista as mensagens informativas e de aviso que você pode encontrar ao usar o simulador de políticas do IAM. A tabela também fornece as etapas necessárias para solucioná-los.

Message	Etapas para solução de problemas
Esta política foi editada. As alterações não serão salvas na sua conta.	Nenhuma ação necessária.

Message	Etapas para solução de problemas
	Essa mensagem é apenas informativa. Se você editar uma política existente no simulador de políticas do IAM, a alteração não afetará sua Conta da AWS. O simulador de políticas permite fazer alterações nas políticas apenas para fins de teste.
Não é possível obter o recurso de política. Motivo: *mensagem de erro detalhada*	O simulador de políticas não é capaz de acessar uma política baseada em recursos solicitada. Certifique-se de que o nome de recurso da Amazon (ARN) esteja correto e que o usuário que executa a simulação tenha permissão para ler a política do recurso.
Uma ou mais políticas exigem valores nas configurações da simulação. Sem esses valores, poderá ocorrer falha na simulação.	Essa mensagem será exibida se a política que você está testando contiver chaves de condição ou variáveis, mas não tiver fornecido os valores para essas chaves ou variáveis em Configurações da simulação. Para ignorar essa mensagem, escolha Simulation Settings (Configurações da simulação) e insira um valor para cada chave de condição ou variável.
Você alterou políticas. Esses resultados não são mais válidos.	Essa mensagem será exibida se você tiver alterado a política selecionada durante a exibição dos resultados no painel Resultados. Os resultados mostrados no painel Resultados não são atualizados dinamicamente. Para ignorar essa mensagem, escolha Executar simulação novamente para exibir novos resultados da simulação com base nas alterações efetuadas no painel Políticas.

Message	Etapas para solução de problemas
O recurso que você digitou para essa simulação não corresponde a esse serviço.	Essa mensagem será exibida se você tiver digitado um nome de recurso da Amazon (ARN) no painel Configurações da simulação que não corresponda ao serviço que você escolheu para a simulação atual. Por exemplo, esta mensagem aparece se você especific ar um ARN para um recurso do Amazon DynamoDB, mas escolher o Amazon Redshift como o serviço a ser simulado. Para ignorar essa mensagem, faça o seguinte: • Remova o nome de recurso da Amazon (ARN) na caixa do painel Configurações da simulação. • Escolha o serviço correspondente ao nome de recurso da Amazon (ARN) especificado em Configurações da simulação.
Esta ação pertence a um serviço que oferece suporte a mecanismos especiais de controle de acesso, além de políticas baseadas em recurso, como ACLs do Amazon S3 ou políticas do S3 Glacier Vault Lock. O simulador de políticas não dá suporte a esses mecanismo s, portanto, os resultados podem ser diferentes de seu ambiente de produção.	Nenhuma ação necessária. Essa mensagem é apenas informativa. Na versão atual, o simulador de políticas avalia políticas anexadas a usuários e grupos de usuários e pode avaliar políticas baseadas em recurso para o Amazon S3, Amazon SQS, Amazon SNS e S3 Glacier. O simulador de políticas não dá suporte a todos os mecanismo s de controle de acesso suportados por outros serviços da AWS.

Message	Etapas para solução de problemas
Atualmente o DynamoDB FGAC não tem suporte.	Nenhuma ação necessária. Essa mensagem informativa se refere a um controle de acesso granular. O controle de acesso refinado é a capacidade de usar condições da política do IAM para determinar quem pode acessar itens de dados individuais e atributos em tabelas e índices do DynamoDB. Isso também se refere às ações que podem ser executadas em tais tabelas e índices. A versão atual do simulador de políticas do IAM não oferece suporte a esse tipo de condição de política. Para obter mais informações sobre o controle de acesso refinado do DynamoDB, consulte Controle de acesso refinado do DynamoDB.
Você tem políticas que não estão em conformidade com a sintaxe da política. Você pode usar a validação de política para revisar as atualizações recomendadas para suas políticas.	Essa mensagem aparecerá na parte superior da lista de políticas se você tiver políticas que não estejam em conformidade com a gramática das políticas do IAM. Para simular essas políticas, revise as opções de validação de política em Validação de políticas do IAM para identificar e corrigir essas políticas.
Essa política deve ser atualizada para seguir as regras de sintaxe de política mais recentes.	Essa mensagem será exibida se você tiver políticas que não estejam em conformidade com a gramática das políticas do IAM. Para simular essas políticas, revise as opções de validação de política em Validação de políticas do IAM para identificar e corrigir essas políticas .

Uso do simulador de políticas do IAM (AWS CLI e API da AWS)

Em geral, os comandos do simulador de políticas exigem chamar operações de API para fazer duas coisas:

1. Avaliar as políticas e retornar a lista de chaves de contexto às quais elas fazem referência. Você precisa saber quais chaves de contexto são referenciadas para poder fornecer valores para elas na próxima etapa.
2. Simular as políticas, fornecendo uma lista de ações, recursos e chaves de contexto usados durante a simulação.

Por motivos de segurança, as operações de API foram divididas em dois grupos:

- Operações de API que simulam apenas as políticas que são transmitidas diretamente para a API como strings. Esse conjunto inclui GetContextKeysForCustomPolicy e SimulateCustomPolicy.
- Operações de API que simulam as políticas anexadas a um usuário, grupo de usuários, função ou recurso especificado do IAM. Como essas operações de API podem revelar detalhes de permissões atribuídas a outras entidades do IAM, você deve considerar a restrição do acesso a essas operações de API. Esse conjunto inclui GetContextKeysForPrincipalPolicy e SimulatePrincipalPolicy. Para obter mais informações sobre como restringir o acesso a operações de API, consulte Exemplo de políticas do AWS Identity and Access Management (IAM).

Nos dois casos, as operações de API simulam o efeito de uma ou mais políticas em uma lista de ações e recursos. Cada ação é combinada com cada recurso, e a simulação determina se as políticas permitem ou negam essa ação para esse recurso. Você também pode fornecer valores para qualquer chave de contexto referenciada por suas políticas. Você pode obter a lista de chaves de contexto às quais as políticas fazem referência primeiro chamando `GetContextKeysForCustomPolicy` ou `GetContextKeysForPrincipalPolicy`. Se você não fornecer um valor para uma chave de contexto, a simulação ainda será executada. No entanto, os resultados podem não ser confiáveis porque o simulador de políticas não pode incluir essa chave de contexto na avaliação.

Para obter a lista de chaves de contexto (AWS CLI, API da AWS)

Use o seguinte para avaliar uma lista de políticas e retornar uma lista de chaves de contexto que são usadas nas políticas.

- AWS CLI: `aws iam get-context-keys-for-custom-policy` e `aws iam get-context-keys-for-principal-policy`
- API da AWS: `GetContextKeysForCustomPolicy` e `GetContextKeysForPrincipalPolicy`

Para simular políticas do IAM (AWS CLI API da AWS)

Use o seguinte para simular políticas do IAM para determinar as permissões em vigor de um usuário.

- AWS CLI: `aws iam simulate-custom-policy` e `aws iam simulate-principal-policy`
- API da AWS: `SimulateCustomPolicy` e `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 AWS Management Console, 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

Ao associar políticas de permissões a identidades (usuários, grupos de usuáros e funções), a terminologia e os procedimentos variam, caso você esteja trabalhando com uma política gerenciada ou 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 IAM.
- Para obter informações sobre como validar as políticas do IAM, consulte Validação de políticas 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 as informações do último acesso.

Adicionar permissões de identidade do IAM (console)

Você pode usar a AWS Management Console 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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (Políticas).
3. Na lista de políticas, escolha o nome da política a ser definida. Você pode 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 AWS Management Console 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 guia Permissions (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 Serviços da AWS 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 AWS Management Console 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. Faça um dos seguintes 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 Create policy (Criar política).

Para alterar o limite de permissões para uma ou mais entidades (console)

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (Políticas).
3. Na lista de políticas, escolha o nome da política a ser definida. Você pode 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. Você pode 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 AWS Management Console 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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (Políticas).
3. Na lista de políticas, marque o botão de seleção ao lado do nome da política a ser desanexada. Você pode 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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (Políticas).
3. Na lista de políticas, escolha o nome da política a ser definida. Você pode 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 AWS Management Console 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 guia Permissions (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 de usuários e funções) 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 de usuários e funções) 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

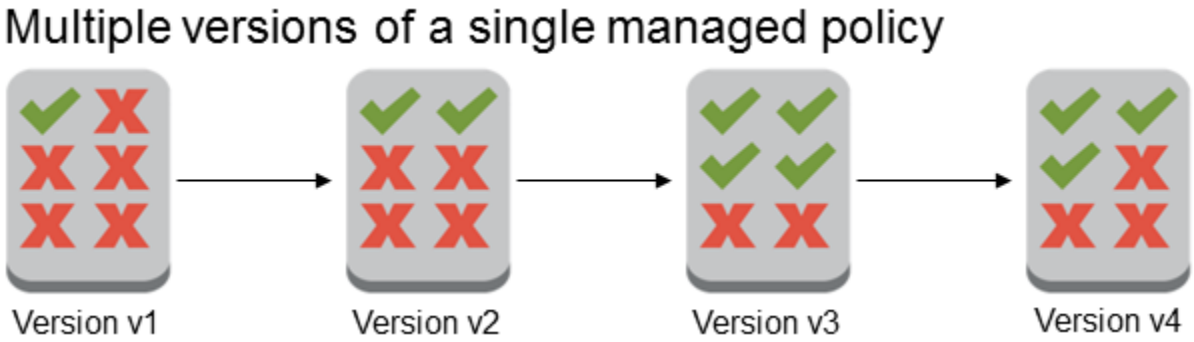
3. Para excluir uma política em linha de uma identidade (usuário, grupo de usuários ou função que não seja uma função vinculada ao serviço), chame uma das seguintes operações:

   - 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

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**
>
> Negar a ação `iam:SetDefaultPolicyVersion` na política de um usuário não impede que ele crie uma nova versão da política e a configure como padrão.

Você pode usar a seguinte política para negar a um usuário acesso para alterar uma política existente gerenciada pelo cliente:

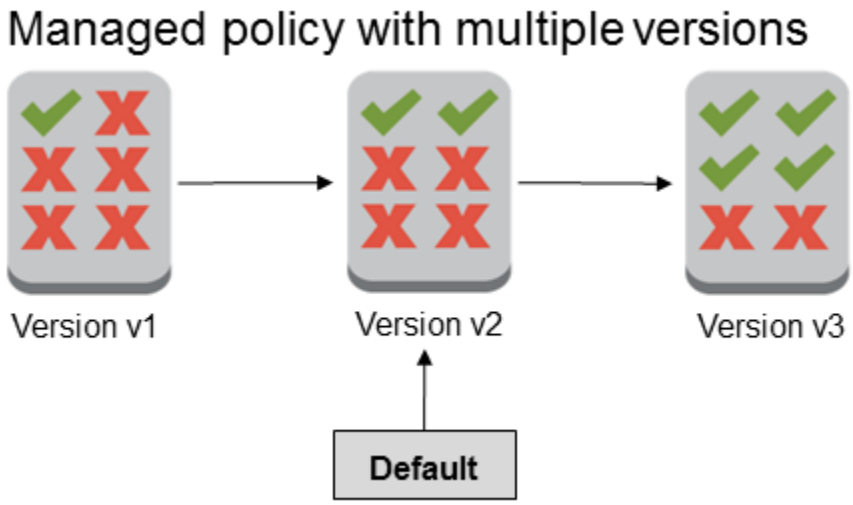
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "iam:CreatePolicyVersion",
                "iam:SetDefaultPolicyVersion"
            ],
            "Resource": "arn:aws:iam::*:policy/POLICY-NAME"
```

```
        }
    ]
}
```

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 operativa, isto é, a versão em vigor para todas as entidades de segurança (usuários, grupos de usuários e funções) as 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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (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 Edição de políticas gerenciadas pelo cliente (AWS CLI).

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 AWS Management Console. 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.

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 AWS Management Console, 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 “Edição de 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.

Edição de 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 obter mais informações, consulte IAM e cotas do AWS STS.

Tópicos

- Visualizar acesso à política
- Edição de políticas gerenciadas pelo cliente (console)
- Edição de políticas em linha (console)
- Edição de políticas gerenciadas pelo cliente (AWS CLI)
- Edição de políticas gerenciadas pelo cliente (API da AWS)

Visualizar acesso à política

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 as 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. 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 AWS Management Console 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, consulte Versionamento de políticas do IAM.

Para editar uma política gerenciada pelo cliente (console)

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (Políticas).
3. Na lista de políticas, escolha o nome da política a editar. Você pode usar a caixa de pesquisa para filtrar a lista de políticas.
4. Escolha a guia Permissões e depois escolha Editar.

5. Faça um dos seguintes 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, erros ou avisos gerais gerados durante a validação de política e depois escolha Next (Avançar).

   > **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.

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.

Para definir a versão padrão de uma política gerenciada pelo cliente (console)

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (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 excluir uma versão de uma política gerenciada pelo cliente (console)

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (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. Em seguida, selecione Excluir.
5. Confirme que você deseja excluir a versão e, em seguida, selecione Excluir.

Edição de políticas em linha (console)

Você pode editar uma política em linha no AWS Management Console.

Para editar uma política em linha para um grupo, grupo de usuários ou função (console)

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. Faça um dos seguintes 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, erros ou avisos gerais gerados durante a validação de política e depois escolha Next (Avançar). 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.

Edição de políticas gerenciadas pelo cliente (AWS CLI)

Você pode editar uma política gerenciada pelo cliente na AWS Command Line Interface (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 de usuários e funções) 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

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

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

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 de usuários e funções) 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

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

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

Exclusão de políticas do IAM

Você pode excluir políticas do IAM usando o AWS Management Console, o AWS Command Line Interface (AWS CLI) ou a API do IAM.

> **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 diferença entre políticas gerenciadas e em linha, consulte Políticas gerenciadas e em linha.

Para obter informações gerais sobre políticas do IAM, consulte Políticas e permissões 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.

Tópicos

- Visualizar acesso à política
- Exclusão de políticas do IAM (console)
- Exclusão de políticas do IAM (AWS CLI)

- Exclusão de políticas do IAM (API da AWS)

Visualizar acesso à política

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 as informações do último acesso.

Exclusão de políticas do IAM (console)

Você pode excluir uma política gerenciada pelo cliente para removê-la de sua Conta da AWS. Você não pode excluir políticas gerenciadas pela AWS.

Para excluir uma política gerenciada pelo cliente (console)

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies (Políticas).
3. Marque o botão de seleção ao lado da política gerenciada pelo cliente que você deseja excluir. Você pode 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.

Para excluir uma política em linha de um grupo de usuários, usuário ou função (console)

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. 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`**.

Exclusão de políticas do IAM (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 de usuários e funções) 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

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

Exclusão de políticas do IAM (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 de usuários e funções) 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

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 as 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
- Atividade de solução de problemas para entidades do IAM e do Organizations
- Onde a AWS rastreia informações acessadas por último
- Visualização das informações acessadas pela última vez para o IAM
- Visualização das informações de último acesso do 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 Visualização das informações acessadas pela última vez para o 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 Visualização das informações de último acesso do Organizations.

Para obter cenários de exemplo de uso das informações do último acesso para tomar decisões sobre as permissões que você concede às suas entidades do 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 do acesso mais recente em um relatório para alterar as permissões de uma identidade do IAM ou do Organizations, analise os detalhes a seguir 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 AWS Management Console, 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 dados do Amazon S3.

- 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 AWS Management Console, 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 uma função ou um usuário federado, 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 Elementos da política JSON da AWS:Principal.
- Recursos do IAM: as informações de acesso mais recente do IAM incluem recursos do IAM (perfis, usuários, grupos de usuários e políticas) em sua conta. As informações de acesso mais recente do Organizations incluem entidades principais (usuários do IAM, perfis do IAM ou o Usuário raiz da conta da AWS) na entidade especificada do 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 Usar funções vinculadas ao serviço Para saber como os diferentes tipos de política são avaliados para permitir ou negar acesso, consulte Lógica da avaliação de política.

- Organizations policy types (Tipos de política do Organizations): as informações do AWS Organizations incluem somente os serviços permitidos pelas políticas de controle de serviço (SCPs) herdadas de uma entidade do 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.
- Specifying a policy ID (Especificar um ID de política): ao usar a AWS CLI ou a API da AWS para gerar um relatório das informações do último acesso do 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 Organizations especificada ou nos filhos da entidade. Para obter mais informações, consulte aws iam generate-organizations-access-report ou GenerateOrganizationsAccessReport.
- Organizations management account (Conta de gerenciamento do Organizations): você deve fazer login na conta de gerenciamento da 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 e outras entidades do Organizations não retornam informações de atividades na conta de gerenciamento.
- Organizations settings (Configurações do Organizations): um administrador deve habilitar as SCPs na raiz de sua organização para que você possa gerar dados do Organizations.

Permissões obrigatórias

Para visualizar as informações acessadas por último no AWS Management Console, 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.

```
{
    "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 raiz, UO ou da conta no 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 AWS CLI ou a API da AWS para visualizar informações do último acesso ao serviço para o 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 Organizations. Além disso, permite o acesso somente leitura a todo o Organizations. Esta política define permissões para acesso programático e do console.

```
{
    "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 Organizations. Para visualizar um exemplo de política, consulte IAM: visualizar as informações do último acesso ao serviço para uma política do Organizations.

Atividade de solução de problemas para entidades do IAM e do Organizations

Em alguns casos, a tabela de informações acessadas por último do AWS Management Console 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 obter mais informações, consulte Tipos de políticas ou Avaliação de políticas 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 Organizations, certifique-se de que você esteja conectado usando as credenciais da conta de gerenciamento do 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

Nome da região	Região	Data de início do rastreamento
Leste dos EUA (Norte da Virgínia)	us-east-1	1 de outubro de 2015
Oeste dos EUA (Norte 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
Europa (Milão)	eu-south-1	28 de abril de 2020

Nome da região	Região	Data de início do rastreamento
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.

Visualização das informações acessadas pela última vez para o IAM

Você pode visualizar informações acessadas pela última vez para o IAM usando o AWS Management Console, 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 as 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 do IAM na guia Access Advisor (Consultor de acesso) no console do IAM.

Para visualizar informações do IAM (console)

1. Faça login no AWS Management Console 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, função ou política para abrir a página Summary (Resumo) e escolha a guia Access Advisor (Consultor de acesso). 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-with-entities` e `get-service-last-accessed-details` 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 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 `GetServiceLastAccessedDetailsWithEntities` e `GetServiceLastAccessedDetails` 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 em uma única conta.

   - ListPoliciesGrantingServiceAccess

Visualização das informações de último acesso do 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 as informações do último acesso.

Ao fazer login no console do IAM usando credenciais da conta de gerenciamento do AWS Organizations, você pode visualizar as informações de qualquer entidade em sua organização. As entidades do 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 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 Organizations especificada ou a subárvore da entidade.

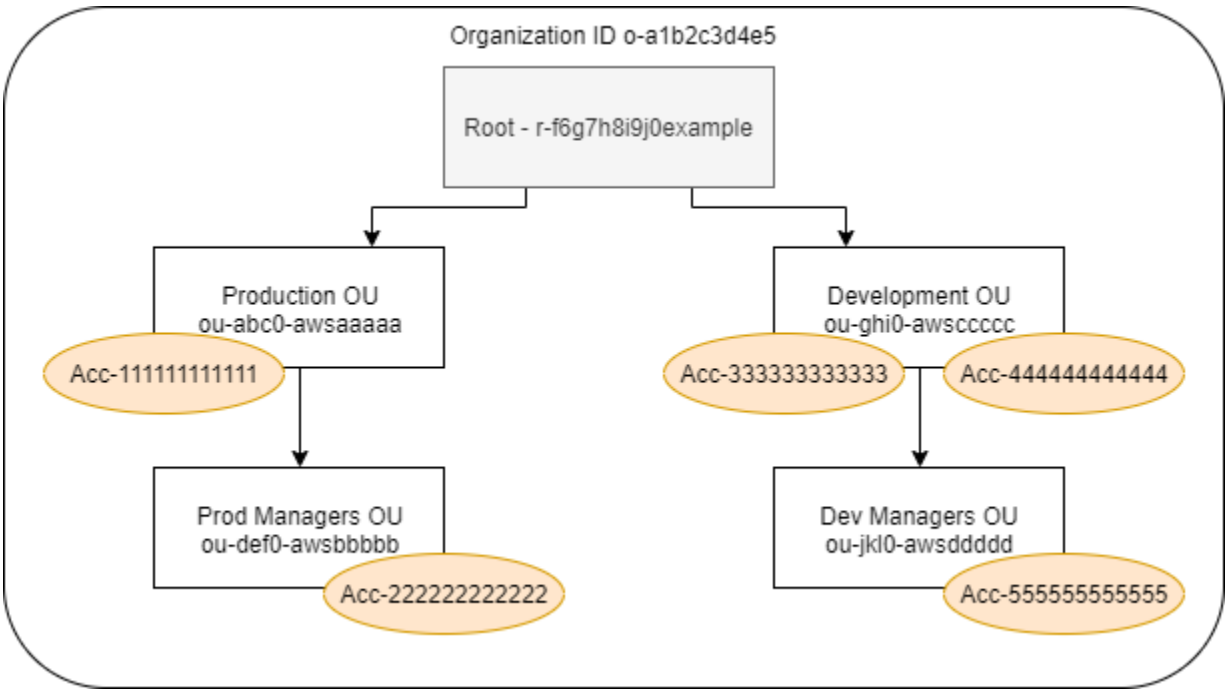
Ao gerar um relatório programático para uma política, você deve especificar uma entidade do Organizations. Esse relatório inclui uma lista de serviços que são permitidos pela SCP especificada. Para cada serviço, ele inclui a atividade de conta mais recente na entidade ou filhos da entidade filhos que 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 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 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 AWS Management Console usando as credenciais da conta de gerenciamento do 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 Edit in AWS Organizations (Editar no AWS Organizations) para visualizar detalhes adicionais e editar a SCP no console do 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 AWS Management Console usando as credenciais da conta de gerenciamento do 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 Edit in AWS Organizations (Editar no AWS Organizations) para visualizar detalhes adicionais e editar a SCP no console do 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 AWS Management Console usando as credenciais da conta de gerenciamento do 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 AWS Management Console usando as credenciais da conta de gerenciamento do 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 Edit in AWS Organizations (Editar no AWS Organizations) para visualizar detalhes adicionais e editar a SCP no console do Organizations. Para obter mais informações, consulte Atualizar uma SCP no Guia do usuário do AWS Organizations.

Visualizar informações do Organizations (AWS CLI)

Você pode usar a AWS CLI para recuperar informações do último acesso ao serviço de sua raiz do Organizations, UO conta ou política.

Para visualizar as informações do último acesso ao serviço Organizations (AWS CLI)

1. Use as credenciais de sua conta de gerenciamento do Organizations com as permissões necessárias do IAM e Organizations e confirme se as SCPs estão habilitadas para sua 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 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 Organizations (API da AWS)

Você pode usar a API da AWS para recuperar as informações do último acesso ao serviço de sua raiz do Organizations, UO conta ou política.

Para visualizar as informações do último acesso ao serviço Organizations (API da AWS)

1. Use as credenciais de sua conta de gerenciamento do Organizations com as permissões necessárias do IAM e Organizations e confirme se as SCPs estão habilitadas para sua 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 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 as 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.

```
{

    "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 AWS Management Console. 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 decide reduzir as permissões de política para incluir apenas os cinco serviços e as ações necessárias do IAM e do Organizations. 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.

```
{
    "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 faz login no console do IAM e escolhe a opção Users (Usuários), escolhe o nome `nikhilj` e, em seguida, escolhe a guia Access Advisor (Consultor de acesso).

Martha analisa a coluna Last Accessed Acessado pela última vez em) e nota que Nikhil não acessou recentemente o IAM, o CloudTrailo , Route 53, o Amazon Elastic Transcoder e vários outros produtos da AWS. Nikhil acessou o Amazon S3. Martha escolhe o S3 na lista de serviços e descobre que Nikhil realizou algumas ações `List` do 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 AWS `IAMFullAccess` 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:

- Users: Exclusão de um usuário
- Grupos: Excluir um grupo
- Funções: Excluir uma função
- Políticas: Excluir uma política gerenciada (também desvincula a política de 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 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 IAM, ele localiza os dados do 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 Edit in AWS Organizations (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 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.

Serviço	Prefixo do serviço
AWS Identity and Access Management Access Analyzer	access-analyzer
AWS Account Management	conta
AWS Certificate Manager	acm
Amazon Managed Workflows for Apache Airflow	airflow
AWS Amplify	amplify
AWS Amplify UI Builder	amplifyuibuilder
Amazon AppIntegrations	app-integrations
AWS AppConfig	appconfig

Serviço	Prefixo do serviço
Amazon AppFlow	appflow
AWS Application Cost Profiler	application-cost-profiler
Amazon CloudWatch Application Insights	applicationinsights
AWS App Mesh	appmesh
Amazon AppStream 2.0	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	batch (lote)
Amazon Braket	braket
AWS Budgets	orçamentos
AWS Cloud9	cloud9
AWS CloudFormation	cloudformation
Amazon CloudFront	cloudfront
AWS CloudHSM	cloudhsm

Serviço	Prefixo do serviço
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
Notificações do AWS CodeStar	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
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect	conectar
AWS Cost and Usage Report	cur
AWS Glue DataBrew	databrew

Serviço	Prefixo do serviço
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
AWS Directory Service	ds
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
Amazon Elastic Kubernetes Service	eks
Amazon Elastic Inference	elastic-inference
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk

Serviço	Prefixo do serviço
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR em EKS (EMR Containers)	emr-containers
Amazon EMR Serverless	emr-serverless
Amazon OpenSearch Service	es
Amazon EventBridge	eventos
Amazon CloudWatch Evidently	evidently
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	gamelift
Amazon Location Service	geo
Amazon S3 Glacier	glacier
Amazon Managed Grafana	grafana
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation

Serviço	Prefixo do serviço
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
Armazenamento de identidades do AWS	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 para Apache Kafka	kafka
Amazon Managed Streaming for Kafka Connect	kafkaconnect

Serviço	Prefixo do serviço
Amazon Kendra	kendra
Amazon Kinesis	kinesis
Amazon Kinesis Analytics V2	kinesisanalytics
AWS Key Management Service	kms
AWS Lambda	lambda
Amazon Lex	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
Amazon Lookout for Vision	lookoutvision
AWS Mainframe Modernization	m2
Amazon Managed Blockchain	managedblockchain
AWS Elemental MediaConnect	mediaconnect
AWS Elemental MediaConvert	mediaconvert
AWS Elemental MediaLive	medialive
AWS Elemental MediaStore	mediastore
AWS Elemental MediaTailor	mediatailor

Serviço	Prefixo do serviço
Amazon MemoryDB para Redis	memorydb
AWS Application Migration Service	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 Panorama	panorama
AWS Performance Insights	pi
Amazon EventBridge Pipes	pipes
Amazon Polly	polly
Amazon Connect Customer Profiles	profile
Amazon QLDB	qldb
AWS Resource Access Manager	ram

Serviço	Prefixo do serviço
AWSLixeira	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
AWS Resource Groups	resource-groups
AWS RoboMaker	robomaker
AWS Identity and Access Management Roles Anywhere	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery Controls	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	sagemaker-geospatial
Savings Plans	savingsplans
Amazon EventBridge Schemas	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub	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	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
Amazon Connect Voice ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Wisdom	wisdom
Amazon WorkLink	worklink
Amazon WorkSpaces	espaços de trabalho

Serviço	Prefixo do serviço
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.

Prefixo do serviço	Ações
access-analyzer	access-analyzer:ApplyArchiveRule
	access-analyzer:CancelPolicyGeneration
	access-analyzer:CheckAccessNotGranted
	access-analyzer:CheckNoNewAccess
	access-analyzer:CreateAccessPreview
	access-analyzer:CreateAnalyzer
	access-analyzer:CreateArchiveRule
	access-analyzer:DeleteAnalyzer
	access-analyzer:DeleteArchiveRule
	access-analyzer:GetAccessPreview
	access-analyzer:GetAnalyzedResource
	access-analyzer:GetAnalyzer
	access-analyzer:GetArchiveRule
	access-analyzer:GetFinding
	access-analyzer:GetGeneratedPolicy

Prefixo do serviço	Ações
	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:UpdateArchiveRule access-analyzer:UpdateFindings access-analyzer:ValidatePolicy
account	account:DeleteAlternateContact account:DisableRegion account:EnableRegion account:GetAlternateContact account:GetContactInformation account:GetRegionOptStatus account:ListRegions account:PutAlternateContact account:PutContactInformation

Prefixo do serviço	Ações
acm	acm:DeleteCertificate acm:DescribeCertificate acm:ExportCertificate acm:GetAccountConfiguration acm:GetCertificate acm:ImportCertificate acm:ListCertificates acm:PutAccountConfiguration acm:RenewCertificate acm:RequestCertificate acm:ResendValidationEmail acm:UpdateCertificateOptions
airflow	airflow:CreateCliToken airflow:CreateEnvironment airflow:CreateWebLoginToken airflow:DeleteEnvironment airflow:GetEnvironment airflow:ListEnvironments airflow:UpdateEnvironment

Prefixo do serviço	Ações
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
	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
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
app-integrations	app-integrations:CreateApplication app-integrations:CreateDataIntegration 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:UpdateEventIntegration

Prefixo do serviço	Ações
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:GetApplication appconfig:GetConfiguration appconfig:GetConfigurationProfile appconfig:GetDeployment appconfig:GetDeploymentStrategy appconfig:GetEnvironment appconfig:GetExtension

Prefixo do serviço	Ações
	appconfig:GetExtensionAssociation appconfig:GetHostedConfigurationVersion appconfig:ListApplications appconfig:ListConfigurationProfiles appconfig:ListDeployments appconfig:ListDeploymentStrategies appconfig:ListEnvironments appconfig:ListExtensionAssociations appconfig:ListExtensions appconfig:ListHostedConfigurationVersions appconfig:StartDeployment appconfig:StopDeployment appconfig:UpdateApplication appconfig:UpdateConfigurationProfile appconfig:UpdateDeploymentStrategy appconfig:UpdateEnvironment appconfig:UpdateExtension appconfig:UpdateExtensionAssociation appconfig:ValidateConfiguration

Prefixo do serviço	Ações
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
	appflow:UpdateFlow
application-cost-profiler	application-cost-profiler:DeleteReportDefinition application-cost-profiler:GetReportDefinition application-cost-profiler:ImportApplicationUsage application-cost-profiler:ListReportDefinitions application-cost-profiler:PutReportDefinition application-cost-profiler:UpdateReportDefinition

Prefixo do serviço	Ações
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:ListLogPatterns

Prefixo do serviço	Ações
	applicationinsights:ListLogPatternSets applicationinsights:ListProblems applicationinsights:ListWorkloads applicationinsights:RemoveWorkload applicationinsights:UpdateApplication applicationinsights:UpdateComponent applicationinsights:UpdateComponentConfiguration applicationinsights:UpdateLogPattern applicationinsights:UpdateWorkload

Prefixo do serviço	Ações
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
	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
appstream	appstream:AssociateAppBlockBuilderAppBlock
	appstream:AssociateApplicationFleet
	appstream:AssociateApplicationToEntitlement
	appstream:AssociateFleet
	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:CreateUpdatedImage
	appstream:CreateUsageReportSubscription
	appstream:CreateUser

Prefixo do serviço	Ações
	appstream:DeleteAppBlock
	appstream:DeleteAppBlockBuilder
	appstream:DeleteApplication
	appstream:DeleteDirectoryConfig
	appstream:DeleteEntitlement
	appstream:DeleteFleet
	appstream:DeleteImage
	appstream:DeleteImageBuilder
	appstream:DeleteImagePermissions
	appstream:DeleteStack
	appstream:DeleteUsageReportSubscription
	appstream:DeleteUser
	appstream:DescribeAppBlockBuilderAppBlockAssociations
	appstream:DescribeAppBlockBuilders
	appstream:DescribeAppBlocks
	appstream:DescribeApplicationFleetAssociations
	appstream:DescribeApplications
	appstream:DescribeDirectoryConfigs
	appstream:DescribeEntitlements
	appstream:DescribeFleets
	appstream:DescribeImageBuilders

Prefixo do serviço	Ações
	appstream:DescribeImagePermissions
	appstream:DescribeImages
	appstream:DescribeSessions
	appstream:DescribeStacks
	appstream:DescribeUsageReportSubscriptions
	appstream:DescribeUsers
	appstream:DescribeUserStackAssociations
	appstream:DisableUser
	appstream:DisassociateAppBlockBuilderAppBlock
	appstream:DisassociateApplicationFleet
	appstream:DisassociateApplicationFromEntitlement
	appstream:DisassociateFleet
	appstream:EnableUser
	appstream:ExpireSession
	appstream:ListAssociatedFleets
	appstream:ListAssociatedStacks
	appstream:ListEntitledApplications
	appstream:StartAppBlockBuilder
	appstream:StartFleet
	appstream:StartImageBuilder
	appstream:StopAppBlockBuilder

Prefixo do serviço	Ações
	appstream:StopFleet appstream:StopImageBuilder appstream:UpdateAppBlockBuilder appstream:UpdateApplication appstream:UpdateDirectoryConfig appstream:UpdateEntitlement appstream:UpdateFleet appstream:UpdateImagePermissions appstream:UpdateStack

Prefixo do serviço	Ações
appsync	appsync:AssociateApi appsync:AssociateMergedGraphqlApi appsync:AssociateSourceGraphqlApi appsync:CreateApiCache appsync:CreateApiKey appsync:CreateDataSource appsync:CreateDomainName appsync:CreateFunction appsync:CreateGraphqlApi appsync:CreateResolver appsync:CreateType appsync:DeleteApiCache appsync:DeleteApiKey appsync:DeleteDataSource appsync:DeleteDomainName appsync:DeleteFunction appsync:DeleteGraphqlApi appsync:DeleteResolver appsync:DeleteType appsync:DisassociateApi appsync:DisassociateMergedGraphqlApi

Prefixo do serviço	Ações
	appsync:DisassociateSourceGraphqlApi appsync:EvaluateCode appsync:EvaluateMappingTemplate appsync:FlushApiCache appsync:GetApiAssociation appsync:GetApiCache appsync:GetDataSource appsync:GetDataSourceIntrospection appsync:GetDomainName appsync:GetFunction appsync:GetGraphqlApi appsync:GetGraphqlApiEnvironmentVariables appsync:GetIntrospectionSchema appsync:GetResolver appsync:GetSchemaCreationStatus appsync:GetSourceApiAssociation appsync:GetType appsync:ListApiKeys appsync:ListDataSources appsync:ListDomainNames appsync:ListFunctions

Prefixo do serviço	Ações
	appsync:ListGraphqlApis appsync:ListResolvers appsync:ListResolversByFunction appsync:ListSourceApiAssociations appsync:ListTypes appsync:ListTypesByAssociation appsync:PutGraphqlApiEnvironmentVariables appsync:StartDataSourceIntrospection appsync:StartSchemaCreation appsync:StartSchemaMerge appsync:UpdateApiCache appsync:UpdateApiKey appsync:UpdateDataSource appsync:UpdateDomainName appsync:UpdateFunction appsync:UpdateGraphqlApi appsync:UpdateResolver appsync:UpdateSourceApiAssociation appsync:UpdateType

Prefixo do serviço	Ações
aps	aps:CreateAlertManagerDefinition
	aps:CreateLoggingConfiguration
	aps:CreateRuleGroupsNamespace
	aps:CreateScraper
	aps:CreateWorkspace
	aps:DeleteAlertManagerDefinition
	aps:DeleteLoggingConfiguration
	aps:DeleteRuleGroupsNamespace
	aps:DeleteScraper
	aps:DeleteWorkspace
	aps:DescribeAlertManagerDefinition
	aps:DescribeLoggingConfiguration
	aps:DescribeRuleGroupsNamespace
	aps:DescribeScraper
	aps:DescribeWorkspace
	aps:GetDefaultScraperConfiguration
	aps:ListRuleGroupsNamespaces
	aps:ListScrapers
	aps:ListWorkspaces
	aps:PutAlertManagerDefinition
	aps:PutRuleGroupsNamespace

Prefixo do serviço	Ações
	aps:UpdateLoggingConfiguration aps:UpdateWorkspaceAlias

Prefixo do serviço	Ações
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
	athena:GetCapacityAssignmentConfiguration athena:GetCapacityReservation athena:GetDatabase athena:GetDataCatalog athena:GetNamedQuery athena:GetNotebookMetadata athena:GetPreparedStatement athena:GetQueryExecution athena:GetQueryResults athena:GetQueryResultsStream athena:GetQueryRuntimeStatistics athena:GetSession athena:GetSessionStatus athena:GetTableMetadata athena:GetWorkGroup athena:ImportNotebook athena:ListApplicationDPUSizes athena:ListCalculationExecutions athena:ListCapacityReservations athena:ListDatabases athena:ListDataCatalogs

Prefixo do serviço	Ações
	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 athena:UpdateNamedQuery athena:UpdateNotebook

Prefixo do serviço	Ações
	athena:UpdateNotebookMetadata athena:UpdatePreparedStatement athena:UpdateWorkGroup

Prefixo do serviço	Ações
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
	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:ListAssessmentFrameworks
	auditmanager:ListAssessmentFrameworkShareRequests
	auditmanager:ListAssessmentReports
	auditmanager:ListAssessments
	auditmanager:ListControlDomainInsights

Prefixo do serviço	Ações
	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
ajuste de escala automático	autoscaling:AttachInstances autoscaling:AttachLoadBalancers autoscaling:AttachLoadBalancerTargetGroups 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
	autoscaling:DescribeAutoScalingNotificationTypes autoscaling:DescribeInstanceRefreshes autoscaling:DescribeLaunchConfigurations autoscaling:DescribeLifecycleHooks autoscaling:DescribeLifecycleHookTypes autoscaling:DescribeLoadBalancers autoscaling:DescribeLoadBalancerTargetGroups autoscaling:DescribeMetricCollectionTypes autoscaling:DescribeNotificationConfigurations autoscaling:DescribePolicies autoscaling:DescribeScalingActivities autoscaling:DescribeScalingProcessTypes autoscaling:DescribeScheduledActions autoscaling:DescribeTerminationPolicyTypes autoscaling:DescribeTrafficSources autoscaling:DescribeWarmPool autoscaling:DetachInstances autoscaling:DetachLoadBalancers autoscaling:DetachLoadBalancerTargetGroups autoscaling:DetachTrafficSources autoscaling:DisableMetricsCollection

Prefixo do serviço	Ações
	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
backup	backup:CancelLegalHold backup:CreateBackupPlan backup:CreateBackupSelection backup:CreateBackupVault backup:CreateFramework backup:CreateLegalHold backup:CreateLogicallyAirGappedBackupVault backup:CreateReportPlan backup:CreateRestoreTestingPlan backup:CreateRestoreTestingSelection backup:DeleteBackupPlan backup:DeleteBackupSelection backup:DeleteBackupVault backup:DeleteBackupVaultAccessPolicy backup:DeleteBackupVaultLockConfiguration backup:DeleteBackupVaultNotifications backup:DeleteFramework backup:DeleteRecoveryPoint backup:DeleteReportPlan backup:DeleteRestoreTestingPlan backup:DeleteRestoreTestingSelection

Prefixo do serviço	Ações
	backup:DescribeBackupJob backup:DescribeBackupVault backup:DescribeCopyJob backup:DescribeFramework backup:DescribeGlobalSettings backup:DescribeProtectedResource backup:DescribeRecoveryPoint backup:DescribeRegionSettings backup:DescribeReportJob backup:DescribeReportPlan backup:DescribeRestoreJob backup:DisassociateRecoveryPoint backup:DisassociateRecoveryPointFromParent backup:ExportBackupPlanTemplate backup:GetBackupPlan backup:GetBackupPlanFromJSON backup:GetBackupPlanFromTemplate backup:GetBackupSelection backup:GetBackupVaultAccessPolicy backup:GetBackupVaultNotifications backup:GetLegalHold

Prefixo do serviço	Ações
	backup:GetRecoveryPointRestoreMetadata backup:GetRestoreJobMetadata backup:GetRestoreTestingInferredMetadata backup:GetRestoreTestingPlan backup:GetRestoreTestingSelection backup:GetSupportedResourceTypes backup:ListBackupJobs backup:ListBackupJobSummaries backup:ListBackupPlans backup:ListBackupPlanTemplates backup:ListBackupPlanVersions backup:ListBackupSelections backup:ListBackupVaults backup:ListCopyJobs backup:ListCopyJobSummaries backup:ListFrameworks backup:ListLegalHolds backup:ListProtectedResources backup:ListRecoveryPointsByBackupVault backup:ListRecoveryPointsByLegalHold backup:ListRecoveryPointsByResource

Prefixo do serviço	Ações
	backup:ListReportJobs backup:ListReportPlans backup:ListRestoreJobs backup:ListRestoreJobsByProtectedResource backup:ListRestoreJobSummaries backup:ListRestoreTestingPlans backup:ListRestoreTestingSelections backup:PutBackupVaultAccessPolicy backup:PutBackupVaultLockConfiguration backup:PutBackupVaultNotifications backup:PutRestoreValidationResult backup:StartBackupJob backup:StartCopyJob backup:StartReportJob backup:StartRestoreJob backup:StopBackupJob backup:UpdateBackupPlan backup:UpdateFramework backup:UpdateGlobalSettings backup:UpdateRecoveryPointLifecycle backup:UpdateRegionSettings

Prefixo do serviço	Ações
	backup:UpdateReportPlan backup:UpdateRestoreTestingPlan backup:UpdateRestoreTestingSelection

Prefixo do serviço	Ações
batch (lote)	batch:CancelJob batch:CreateComputeEnvironment batch:CreateJobQueue batch:CreateSchedulingPolicy batch:DeleteComputeEnvironment batch:DeleteJobQueue batch:DeleteSchedulingPolicy batch:DeregisterJobDefinition batch:DescribeComputeEnvironments batch:DescribeJobDefinitions batch:DescribeJobQueues batch:DescribeJobs batch:DescribeSchedulingPolicies batch:ListJobs batch:ListSchedulingPolicies batch:RegisterJobDefinition batch:SubmitJob batch:TerminateJob batch:UpdateComputeEnvironment batch:UpdateJobQueue batch:UpdateSchedulingPolicy

Prefixo do serviço	Ações
braket	braket:AcceptUserAgreement braket:AccessBraketFeature braket:CancelJob braket:CancelQuantumTask braket:CreateJob braket:CreateQuantumTask braket:GetDevice braket:GetJob braket:GetQuantumTask braket:GetServiceLinkedRoleStatus braket:GetUserAgreementStatus braket:SearchDevices braket:SearchJobs braket:SearchQuantumTasks

Prefixo do serviço	Ações
orçamentos	budgets:ModifyBudget
	budgets:CreateBudgetAction
	budgets:ModifyBudget
	budgets:ModifyBudget
	budgets:ModifyBudget
	budgets:DeleteBudgetAction
	budgets:ModifyBudget
	budgets:ModifyBudget
	budgets:ViewBudget
	budgets:DescribeBudgetAction
	budgets:DescribeBudgetActionHistories
	budgets:DescribeBudgetActionsForAccount
	budgets:DescribeBudgetActionsForBudget
	budgets:ViewBudget
	budgets:ViewBudget
	budgets:ViewBudget
	budgets:ViewBudget
	budgets:ViewBudget
	budgets:ExecuteBudgetAction
	budgets:ModifyBudget
	budgets:UpdateBudgetAction

Prefixo do serviço	Ações
	budgets:ModifyBudget budgets:ModifyBudget
cloud9	cloud9:CreateEnvironmentEC2 cloud9:CreateEnvironmentMembership cloud9:DeleteEnvironment cloud9:DeleteEnvironmentMembership cloud9:DescribeEnvironmentMemberships cloud9:DescribeEnvironments cloud9:DescribeEnvironmentStatus cloud9:ListEnvironments cloud9:UpdateEnvironment cloud9:UpdateEnvironmentMembership

Prefixo do serviço	Ações
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:DescribeGeneratedTemplate
	cloudformation:DescribeOrganizationsAccess
	cloudformation:DescribePublisher

Prefixo do serviço	Ações
	cloudformation:DescribeResourceScan
	cloudformation:DescribeStackDriftDetectionStatus
	cloudformation:DescribeStackEvents
	cloudformation:DescribeStackInstance
	cloudformation:DescribeStackResource
	cloudformation:DescribeStackResourceDrifts
	cloudformation:DescribeStackResources
	cloudformation:DescribeStacks
	cloudformation:DescribeStackSet
	cloudformation:DescribeStackSetOperation
	cloudformation:DescribeType
	cloudformation:DescribeTypeRegistration
	cloudformation:DetectStackDrift
	cloudformation:DetectStackResourceDrift
	cloudformation:DetectStackSetDrift
	cloudformation:EstimateTemplateCost
	cloudformation:ExecuteChangeSet
	cloudformation:GetGeneratedTemplate
	cloudformation:GetStackPolicy
	cloudformation:GetTemplate
	cloudformation:GetTemplateSummary

Prefixo do serviço	Ações
	cloudformation:ImportStacksToStackSet
	cloudformation:ListChangeSets
	cloudformation:ListExports
	cloudformation:ListGeneratedTemplates
	cloudformation:ListImports
	cloudformation:ListResourceScanRelatedResources
	cloudformation:ListResourceScanResources
	cloudformation:ListResourceScans
	cloudformation:ListStackInstanceResourceDrifts
	cloudformation:ListStackInstances
	cloudformation:ListStackResources
	cloudformation:ListStackSetAutoDeploymentTargets
	cloudformation:ListStackSetOperationResults
	cloudformation:ListStackSetOperations
	cloudformation:ListStackSets
	cloudformation:ListTypeRegistrations
	cloudformation:ListTypes
	cloudformation:ListTypeVersions
	cloudformation:PublishType
	cloudformation:RecordHandlerProgress
	cloudformation:RegisterPublisher

Prefixo do serviço	Ações
	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
cloudfront	cloudfront:AssociateAlias cloudfront:CreateCachePolicy cloudfront:CreateCloudFrontOriginAccessIdentity cloudfront:CreateContinuousDeploymentPolicy 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:DeleteCachePolicy cloudfront:DeleteCloudFrontOriginAccessIdentity cloudfront:DeleteContinuousDeploymentPolicy cloudfront:DeleteDistribution cloudfront:DeleteFieldLevelEncryptionConfig

Prefixo do serviço	Ações
	cloudfront:DeleteFieldLevelEncryptionProfile cloudfront:DeleteFunction cloudfront:DeleteKeyGroup cloudfront:DeleteKeyValueStore cloudfront:DeleteMonitoringSubscription cloudfront:DeleteOriginAccessControl cloudfront:DeleteOriginRequestPolicy cloudfront:DeletePublicKey cloudfront:DeleteRealtimeLogConfig cloudfront:DeleteResponseHeadersPolicy cloudfront:DeleteStreamingDistribution cloudfront:DescribeFunction cloudfront:DescribeKeyValueStore cloudfront:GetCachePolicy cloudfront:GetCachePolicyConfig cloudfront:GetCloudFrontOriginAccessIdentity cloudfront:GetCloudFrontOriginAccessIdentityConfig cloudfront:GetContinuousDeploymentPolicy cloudfront:GetContinuousDeploymentPolicyConfig cloudfront:GetDistributionConfig cloudfront:GetFieldLevelEncryption

Prefixo do serviço	Ações
	cloudfront:GetFieldLevelEncryptionConfig cloudfront:GetFieldLevelEncryptionProfile cloudfront:GetFieldLevelEncryptionProfileConfig cloudfront:GetFunction cloudfront:GetInvalidation cloudfront:GetKeyGroup 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:ListCachePolicies cloudfront:ListCloudFrontOriginAccessIdentities

Prefixo do serviço	Ações
	cloudfront:ListConflictingAliases cloudfront:ListContinuousDeploymentPolicies cloudfront:ListDistributions cloudfront:ListDistributionsByCachePolicyId cloudfront:ListDistributionsByKeyGroup cloudfront:ListDistributionsByOriginRequestPolicyId cloudfront:ListDistributionsByRealtimeLogConfig cloudfront:ListDistributionsByResponseHeadersPolicyId cloudfront:ListDistributionsByWebACLId cloudfront:ListFieldLevelEncryptionConfigs cloudfront:ListFieldLevelEncryptionProfiles cloudfront:ListFunctions cloudfront:ListInvalidations cloudfront:ListKeyGroups cloudfront:ListKeyValueStores cloudfront:ListOriginAccessControls cloudfront:ListOriginRequestPolicies cloudfront:ListPublicKeys cloudfront:ListRealtimeLogConfigs cloudfront:ListResponseHeadersPolicies cloudfront:ListStreamingDistributions

Prefixo do serviço	Ações
	cloudfront:PublishFunction cloudfront:TestFunction cloudfront:UpdateCachePolicy cloudfront:UpdateCloudFrontOriginAccessIdentity cloudfront:UpdateContinuousDeploymentPolicy cloudfront:UpdateDistribution cloudfront:UpdateFieldLevelEncryptionConfig cloudfront:UpdateFieldLevelEncryptionProfile cloudfront:UpdateFunction cloudfront:UpdateKeyGroup cloudfront:UpdateKeyValueStore cloudfront:UpdateOriginAccessControl cloudfront:UpdateOriginRequestPolicy cloudfront:UpdatePublicKey cloudfront:UpdateRealtimeLogConfig cloudfront:UpdateResponseHeadersPolicy

Prefixo do serviço	Ações
cloudhsm	cloudhsm:CreateHapg
	cloudhsm:CreateHsm
	cloudhsm:CreateLunaClient
	cloudhsm:DeleteBackup
	cloudhsm:DeleteHapg
	cloudhsm:DeleteHsm
	cloudhsm:DeleteLunaClient
	cloudhsm:DescribeBackups
	cloudhsm:DescribeClusters
	cloudhsm:DescribeHapg
	cloudhsm:DescribeHsm
	cloudhsm:DescribeLunaClient
	cloudhsm:GetConfig
	cloudhsm:InitializeCluster
	cloudhsm:ListAvailableZones
	cloudhsm:ListHapgs
	cloudhsm:ListHsms
	cloudhsm:ListLunaClients
	cloudhsm:ModifyBackupAttributes
	cloudhsm:ModifyCluster
	cloudhsm:ModifyHapg

Prefixo do serviço	Ações
	cloudhsm:ModifyHsm cloudhsm:ModifyLunaClient cloudhsm:RestoreBackup

Prefixo do serviço	Ações
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
	cloudsearch:ListDomainNames cloudsearch:UpdateAvailabilityOptions cloudsearch:UpdateDomainEndpointOptions cloudsearch:UpdateScalingParameters cloudsearch:UpdateServiceAccessPolicies

Prefixo do serviço	Ações
cloudtrail	cloudtrail:CancelQuery
	cloudtrail:CreateChannel
	cloudtrail:CreateEventDataStore
	cloudtrail:CreateTrail
	cloudtrail:DeleteChannel
	cloudtrail:DeleteEventDataStore
	cloudtrail:DeleteResourcePolicy
	cloudtrail:DeleteTrail
	cloudtrail:DeregisterOrganizationDelegatedAdmin
	cloudtrail:DescribeQuery
	cloudtrail:DescribeTrails
	cloudtrail:DisableFederation
	cloudtrail:GetChannel
	cloudtrail:GetEventDataStore
	cloudtrail:GetEventDataStoreData
	cloudtrail:GetEventSelectors
	cloudtrail:GetImport
	cloudtrail:GetInsightSelectors
	cloudtrail:GetQueryResults
	cloudtrail:GetResourcePolicy
	cloudtrail:GetTrail

Prefixo do serviço	Ações
	cloudtrail:GetTrailStatus
	cloudtrail:ListChannels
	cloudtrail:ListEventDataStores
	cloudtrail:ListImportFailures
	cloudtrail:ListImports
	cloudtrail:ListPublicKeys
	cloudtrail:ListQueries
	cloudtrail:ListTrails
	cloudtrail:LookupEvents
	cloudtrail:PutEventSelectors
	cloudtrail:PutInsightSelectors
	cloudtrail:PutResourcePolicy
	cloudtrail:RegisterOrganizationDelegatedAdmin
	cloudtrail:RestoreEventDataStore
	cloudtrail:StartEventDataStoreIngestion
	cloudtrail:StartImport
	cloudtrail:StartLogging
	cloudtrail:StartQuery
	cloudtrail:StopEventDataStoreIngestion
	cloudtrail:StopImport
	cloudtrail:StopLogging

Prefixo do serviço	Ações
	cloudtrail:UpdateChannel cloudtrail:UpdateEventDataStore cloudtrail:UpdateTrail

Prefixo do serviço	Ações
cloudwatch	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:GetDashboard cloudwatch:GetInsightRuleReport cloudwatch:GetMetricStream cloudwatch:ListDashboards cloudwatch:ListManagedInsightRules cloudwatch:ListMetricStreams cloudwatch:PutAnomalyDetector

Prefixo do serviço	Ações
	cloudwatch:PutCompositeAlarm cloudwatch:PutDashboard cloudwatch:PutInsightRule cloudwatch:PutManagedInsightRules cloudwatch:PutMetricAlarm cloudwatch:PutMetricStream cloudwatch:SetAlarmState cloudwatch:StartMetricStreams cloudwatch:StopMetricStreams

Prefixo do serviço	Ações
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
	codeartifact:GetRepositoryEndpoint
	codeartifact:GetRepositoryPermissionsPolicy
	codeartifact:ListDomains
	codeartifact:ListPackageGroups
	codeartifact:ListPackages
	codeartifact:ListPackageVersionAssets
	codeartifact:ListPackageVersionDependencies
	codeartifact:ListPackageVersions
	codeartifact:ListRepositories
	codeartifact:ListRepositoriesInDomain
	codeartifact:PublishPackageVersion
	codeartifact:PutDomainPermissionsPolicy
	codeartifact:PutPackageMetadata
	codeartifact:PutPackageOriginConfiguration
	codeartifact:PutRepositoryPermissionsPolicy
	codeartifact:ReadFromRepository
	codeartifact:UpdatePackageVersionsStatus
	codeartifact:UpdateRepository

Prefixo do serviço	Ações
codedeploy	codedeploy:BatchGetApplicationRevisions codedeploy:BatchGetApplications codedeploy:BatchGetDeploymentGroups codedeploy:BatchGetDeploymentInstances codedeploy:BatchGetDeployments codedeploy:BatchGetDeploymentTargets 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
	codedeploy:GetDeploymentConfig codedeploy:GetDeploymentGroup codedeploy:GetDeploymentInstance codedeploy:GetDeploymentTarget codedeploy:GetOnPremisesInstance codedeploy:ListApplicationRevisions codedeploy:ListApplications codedeploy:ListDeploymentConfigs codedeploy:ListDeploymentGroups codedeploy:ListDeploymentInstances codedeploy:ListDeployments codedeploy:ListDeploymentTargets codedeploy:ListGitHubAccountTokenNames codedeploy:ListOnPremisesInstances codedeploy:PutLifecycleEventHookExecutionStatus codedeploy:RegisterApplicationRevision codedeploy:RegisterOnPremisesInstance codedeploy:SkipWaitTimeForInstanceTermination codedeploy:StopDeployment codedeploy:UpdateApplication codedeploy:UpdateDeploymentGroup

Prefixo do serviço	Ações
codeguru-profiler	codeguru-profiler:AddNotificationChannels
	codeguru-profiler:BatchGetFrameMetricData
	codeguru-profiler:ConfigureAgent
	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
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
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:ListWebhooks
	codepipeline:PollForJobs
	codepipeline:PollForThirdPartyJobs

Prefixo do serviço	Ações
	codepipeline:PutActionRevision codepipeline:PutApprovalResult codepipeline:PutJobFailureResult codepipeline:PutJobSuccessResult codepipeline:PutThirdPartyJobFailureResult codepipeline:PutThirdPartyJobSuccessResult codepipeline:PutWebhook codepipeline:RegisterWebhookWithThirdParty codepipeline:StartPipelineExecution codepipeline:StopPipelineExecution codepipeline:UpdateActionType codepipeline:UpdatePipeline

Prefixo do serviço	Ações
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
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
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
cognito-idp	cognito-idp:AddCustomAttributes
	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:AdminInitiateAuth
	cognito-idp:AdminLinkProviderForUser
	cognito-idp:AdminListDevices
	cognito-idp:AdminListGroupsForUser
	cognito-idp:AdminListUserAuthEvents
	cognito-idp:AdminRemoveUserFromGroup
	cognito-idp:AdminResetUserPassword
	cognito-idp:AdminRespondToAuthChallenge
	cognito-idp:AdminSetUserMFAPreference

Prefixo do serviço	Ações
	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:CreateResourceServer
	cognito-idp:CreateUserImportJob
	cognito-idp:CreateUserPool
	cognito-idp:CreateUserPoolClient
	cognito-idp:CreateUserPoolDomain
	cognito-idp:DeleteGroup
	cognito-idp:DeleteIdentityProvider
	cognito-idp:DeleteResourceServer

Prefixo do serviço	Ações
	cognito-idp:DeleteUser
	cognito-idp:DeleteUserAttributes
	cognito-idp:DeleteUserPool
	cognito-idp:DeleteUserPoolClient
	cognito-idp:DeleteUserPoolDomain
	cognito-idp:DescribeIdentityProvider
	cognito-idp:DescribeResourceServer
	cognito-idp:DescribeRiskConfiguration
	cognito-idp:DescribeUserImportJob
	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:GetSigningCertificate
	cognito-idp:GetUICustomization

Prefixo do serviço	Ações
	cognito-idp:GetUser
	cognito-idp:GetUserAttributeVerificationCode
	cognito-idp:GetUserPoolMfaConfig
	cognito-idp:GlobalSignOut
	cognito-idp:InitiateAuth
	cognito-idp:ListDevices
	cognito-idp:ListGroups
	cognito-idp:ListIdentityProviders
	cognito-idp:ListResourceServers
	cognito-idp:ListUserImportJobs
	cognito-idp:ListUserPoolClients
	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

Prefixo do serviço	Ações
	cognito-idp:SetUserPoolMfaConfig
	cognito-idp:SetUserSettings
	cognito-idp:SignUp
	cognito-idp:StartUserImportJob
	cognito-idp:StopUserImportJob
	cognito-idp:UpdateAuthEventFeedback
	cognito-idp:UpdateDeviceStatus
	cognito-idp:UpdateGroup
	cognito-idp:UpdateIdentityProvider
	cognito-idp:UpdateResourceServer
	cognito-idp:UpdateUserAttributes
	cognito-idp:UpdateUserPool
	cognito-idp:UpdateUserPoolClient
	cognito-idp:UpdateUserPoolDomain
	cognito-idp:VerifySoftwareToken
	cognito-idp:VerifyUserAttribute

Prefixo do serviço	Ações
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
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
	comprehendmedical:StopICD10CMInferenceJob comprehendmedical:StopPHIDetectionJob comprehendmedical:StopRxNormInferenceJob comprehendmedical:StopSNOMEDCTInferenceJob

Prefixo do serviço	Ações
compute-optimizer	compute-optimizer:DeleteRecommendationPreferences compute-optimizer:DescribeRecommendationExportJobs compute-optimizer:ExportAutoScalingGroupRecommendations compute-optimizer:ExportEBSVolumeRecommendations compute-optimizer:ExportEC2InstanceRecommendations compute-optimizer:ExportECSServiceRecommendations compute-optimizer:ExportLambdaFunctionRecommendations compute-optimizer:ExportLicenseRecommendations compute-optimizer:GetEC2RecommendationProjectedMetrics compute-optimizer:GetECSServiceRecommendationProjectedMetrics compute-optimizer:GetEffectiveRecommendationPreferences compute-optimizer:GetEnrollmentStatus compute-optimizer:GetEnrollmentStatusesForOrganization compute-optimizer:GetRecommendationPreferences compute-optimizer:GetRecommendationSummaries compute-optimizer:PutRecommendationPreferences compute-optimizer:UpdateEnrollmentStatus

Prefixo do serviço	Ações
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
	config:DescribeComplianceByResource
	config:DescribeConfigRuleEvaluationStatus
	config:DescribeConfigRules
	config:DescribeConfigurationAggregators
	config:DescribeConfigurationAggregatorSourcesStatus
	config:DescribeConfigurationRecorders
	config:DescribeConfigurationRecorderStatus
	config:DescribeConformancePackCompliance
	config:DescribeConformancePacks
	config:DescribeConformancePackStatus
	config:DescribeDeliveryChannels
	config:DescribeDeliveryChannelStatus
	config:DescribeOrganizationConfigRules
	config:DescribeOrganizationConfigRuleStatuses
	config:DescribeOrganizationConformancePacks
	config:DescribeOrganizationConformancePackStatuses
	config:DescribePendingAggregationRequests
	config:DescribeRemediationConfigurations
	config:DescribeRemediationExceptions
	config:DescribeRemediationExecutionStatus
	config:DescribeRetentionConfigurations

Prefixo do serviço	Ações
	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:ListConformancePackComplianceScores
	config:ListDiscoveredResources
	config:ListResourceEvaluations
	config:ListStoredQueries
	config:PutConfigRule
	config:PutConfigurationAggregator
	config:PutConfigurationRecorder

Prefixo do serviço	Ações
	config:PutConformancePack
	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
conectar	connect:ActivateEvaluationForm connect:AssociateApprovedOrigin connect:AssociateBot connect:AssociateDefaultVocabulary connect:AssociateFlow connect:AssociateInstanceStorageConfig connect:AssociateLambdaFunction connect:AssociateLexBot connect:AssociatePhoneNumberContactFlow connect:AssociateQueueQuickConnects connect:AssociateRoutingProfileQueues connect:AssociateSecurityKey connect:AssociateUserProficiencies connect:BatchGetFlowAssociation connect:BatchPutContact connect:ClaimPhoneNumber connect:CreateAgentStatus connect:CreateContactFlow connect:CreateContactFlowModule connect:CreateEvaluationForm connect:CreateHoursOfOperation

Prefixo do serviço	Ações
	connect:CreateInstance connect:CreateIntegrationAssociation connect:CreateParticipant connect:CreatePersistentContactAssociation connect:CreatePredefinedAttribute connect:CreatePrompt 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:DeactivateEvaluationForm connect:DeleteContactEvaluation

Prefixo do serviço	Ações
	connect:DeleteContactFlow connect:DeleteContactFlowModule connect:DeleteEvaluationForm connect:DeleteHoursOfOperation connect:DeleteInstance connect:DeleteIntegrationAssociation connect:DeletePredefinedAttribute connect:DeletePrompt connect:DeleteQueue connect:DeleteQuickConnect connect:DeleteRoutingProfile connect:DeleteRule connect:DeleteSecurityProfile connect:DeleteTaskTemplate connect:DeleteTrafficDistributionGroup connect:DeleteUseCase connect:DeleteUser connect:DeleteUserHierarchyGroup connect:DeleteView connect:DeleteVocabulary connect:DescribeAgentStatus

Prefixo do serviço	Ações
	connect:DescribeContactEvaluation connect:DescribeContactFlow connect:DescribeContactFlowModule connect:DescribeEvaluationForm connect:DescribeInstanceAttribute connect:DescribeInstanceStorageConfig connect:DescribePhoneNumber connect:DescribeRule connect:DescribeTrafficDistributionGroup connect:DescribeUserHierarchyGroup connect:DescribeUserHierarchyStructure connect:DescribeView connect:DescribeVocabulary connect:DisassociateApprovedOrigin connect:DisassociateBot connect:DisassociateFlow connect:DisassociateInstanceStorageConfig connect:DisassociateLambdaFunction connect:DisassociateLexBot connect:DisassociatePhoneNumberContactFlow connect:DisassociateQueueQuickConnects

Prefixo do serviço	Ações
	connect:DisassociateRoutingProfileQueues connect:DisassociateSecurityKey connect:DisassociateUserProficiencies connect:DismissUserContact connect:GetContactAttributes connect:GetCurrentMetricData connect:GetCurrentUserData connect:GetFederationToken connect:GetFlowAssociation connect:GetMetricData connect:GetMetricDataV2 connect:GetPromptFile connect:GetTaskTemplate connect:GetTrafficDistribution connect:ImportPhoneNumber connect:ListApprovedOrigins connect:ListBots connect:ListContactEvaluations connect:ListContactFlowModules connect:ListContactFlows connect:ListContactReferences

Prefixo do serviço	Ações
	connect:ListDefaultVocabularies connect:ListEvaluationForms connect:ListEvaluationFormVersions connect:ListFlowAssociations connect:ListHoursOfOperations connect:ListInstanceAttributes connect:ListInstanceStorageConfigs connect:ListIntegrationAssociations connect:ListLambdaFunctions connect:ListLexBots connect:ListPhoneNumbers connect:ListPhoneNumbersV2 connect:ListPredefinedAttributes connect:ListPrompts connect:ListQueueQuickConnects connect:ListQueues connect:ListQuickConnects connect:ListRealtimeContactAnalysisSegmentsV2 connect:ListRoutingProfileQueues connect:ListRoutingProfiles connect:ListRules

Prefixo do serviço	Ações
	connect:ListSecurityKeys connect:ListSecurityProfileApplications connect:ListSecurityProfilePermissions connect:ListSecurityProfiles connect:ListTaskTemplates connect:ListTrafficDistributionGroups connect:ListUseCases connect:ListUserHierarchyGroups connect:ListUserProficiencies connect:ListUsers connect:ListViews connect:ListViewVersions connect:MonitorContact connect:PauseContact connect:PutUserStatus connect:ReleasePhoneNumber connect:ReplicateInstance connect:ResumeContact connect:ResumeContactRecording connect:SearchAvailablePhoneNumbers connect:SearchContacts

Prefixo do serviço	Ações
	connect:SearchHoursOfOperations connect:SearchPredefinedAttributes connect:SearchPrompts connect:SearchQueues connect:SearchQuickConnects connect:SearchRoutingProfiles connect:SearchSecurityProfiles connect:SearchVocabularies connect:SendChatIntegrationEvent connect:StartChatContact connect:StartContactEvaluation connect:StartContactRecording connect:StartContactStreaming connect:StartOutboundVoiceContact connect:StartTaskContact connect:StartWebRTCContact connect:StopContact connect:StopContactRecording connect:StopContactStreaming connect:SubmitContactEvaluation connect:SuspendContactRecording

Prefixo do serviço	Ações
	connect:TransferContact connect:UpdateAgentStatus connect:UpdateContact connect:UpdateContactAttributes connect:UpdateContactEvaluation connect:UpdateContactFlowContent connect:UpdateContactFlowMetadata connect:UpdateContactFlowModuleContent connect:UpdateContactFlowModuleMetadata connect:UpdateContactFlowName connect:UpdateContactRoutingData connect:UpdateContactSchedule connect:UpdateEvaluationForm connect:UpdateHoursOfOperation connect:UpdateInstanceAttribute connect:UpdateInstanceStorageConfig connect:UpdateParticipantRoleConfig connect:UpdatePhoneNumber connect:UpdatePhoneNumberMetadata connect:UpdatePredefinedAttribute connect:UpdatePrompt

Prefixo do serviço	Ações
	connect:UpdateQueueHoursOfOperation connect:UpdateQueueMaxContacts connect:UpdateQueueName connect:UpdateQueueOutboundCallerConfig 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 connect:UpdateUserHierarchyGroupName connect:UpdateUserHierarchyStructure connect:UpdateUserIdentityInfo connect:UpdateUserPhoneConfig

Prefixo do serviço	Ações
	connect:UpdateUserProficiencies connect:UpdateUserRoutingProfile connect:UpdateUserSecurityProfiles connect:UpdateViewContent connect:UpdateViewMetadata
cur	cur:DeleteReportDefinition cur:DescribeReportDefinitions cur:ModifyReportDefinition cur:PutReportDefinition

Prefixo do serviço	Ações
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
	databrew:ListDatasets databrew:ListJobRuns databrew:ListJobs databrew:ListProjects databrew:ListRecipes databrew:ListRecipeVersions 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
dataexchange	dataexchange:CancelJob
	dataexchange:CreateDataSet
	dataexchange:CreateEventAction
	dataexchange:CreateJob
	dataexchange:CreateRevision
	dataexchange:DeleteAsset
	dataexchange:DeleteEventAction
	dataexchange:DeleteRevision
	dataexchange:GetEventAction
	dataexchange:GetJob
	dataexchange:ListDataSetRevisions
	dataexchange:ListDataSets
	dataexchange:ListEventActions
	dataexchange:ListJobs
	dataexchange:ListRevisionAssets
	dataexchange:RevokeRevision
	dataexchange:SendDataSetNotification
	dataexchange:StartJob
	dataexchange:UpdateAsset
	dataexchange:UpdateDataSet
	dataexchange:UpdateEventAction

Prefixo do serviço	Ações
	dataexchange:UpdateRevision
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
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
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
	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
	devicefarm:ListNetworkProfiles devicefarm:ListOfferingPromotions devicefarm:ListOfferings devicefarm:ListOfferingTransactions 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
	devicefarm:StopRemoteAccessSession devicefarm:StopRun devicefarm:UpdateDeviceInstance devicefarm:UpdateDevicePool devicefarm:UpdateInstanceProfile devicefarm:UpdateNetworkProfile devicefarm:UpdateProject devicefarm:UpdateTestGridProject devicefarm:UpdateUpload devicefarm:UpdateVPCEConfiguration

Prefixo do serviço	Ações
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
	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
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
	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
	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
dlm	dlm:CreateLifecyclePolicy
	dlm:DeleteLifecyclePolicy
	dlm:GetLifecyclePolicies
	dlm:GetLifecyclePolicy
	dlm:UpdateLifecyclePolicy

Prefixo do serviço	Ações
dms	dms:ApplyPendingMaintenanceAction dms:BatchStartRecommendations 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:DeleteDataProvider dms:DeleteEndpoint dms:DeleteEventSubscription dms:DeleteFleetAdvisorCollector dms:DeleteFleetAdvisorDatabases dms:DeleteInstanceProfile dms:DeleteMigrationProject

Prefixo do serviço	Ações
	dms:DeleteReplicationConfig dms:DeleteReplicationInstance dms:DeleteReplicationSubnetGroup dms:DeleteReplicationTask dms:DeleteReplicationTaskAssessmentRun dms:DescribeAccountAttributes dms:DescribeApplicableIndividualAssessments dms:DescribeCertificates dms:DescribeConnections dms:DescribeEndpoints dms:DescribeEndpointSettings dms:DescribeEndpointTypes dms:DescribeEngineVersions dms:DescribeEventCategories dms:DescribeEvents dms:DescribeEventSubscriptions dms:DescribeFleetAdvisorCollectors dms:DescribeFleetAdvisorDatabases dms:DescribeFleetAdvisorLsaAnalysis dms:DescribeFleetAdvisorSchemaObjectSummary dms:DescribeFleetAdvisorSchemas

Prefixo do serviço	Ações
	dms:DescribeMetadataModelImports dms:DescribeOrderableReplicationInstances dms:DescribePendingMaintenanceActions dms:DescribeRecommendationLimitations dms:DescribeRecommendations dms:DescribeRefreshSchemasStatus dms:DescribeReplicationConfigs dms:DescribeReplicationInstances dms:DescribeReplicationInstanceTaskLogs dms:DescribeReplications dms:DescribeReplicationSubnetGroups dms:DescribeReplicationTableStatistics dms:DescribeReplicationTaskAssessmentResults dms:DescribeReplicationTaskAssessmentRuns dms:DescribeReplicationTaskIndividualAssessments dms:DescribeReplicationTasks dms:DescribeSchemas dms:DescribeTableStatistics dms:ExportMetadataModelAssessment dms:GetMetadataModel dms:ImportCertificate

Prefixo do serviço	Ações
	dms:ListMetadataModelAssessmentActionItems dms:ModifyEndpoint dms:ModifyEventSubscription dms:ModifyReplicationConfig dms:ModifyReplicationInstance dms:ModifyReplicationSubnetGroup dms:ModifyReplicationTask dms:MoveReplicationTask dms:RebootReplicationInstance dms:RefreshSchemas dms:ReloadReplicationTables dms:ReloadTables dms:RunFleetAdvisorLsaAnalysis dms:StartMetadataModelAssessment dms:StartMetadataModelConversion dms:StartMetadataModelExportToTarget dms:StartRecommendations dms:StartReplication dms:StartReplicationTask dms:StartReplicationTaskAssessment dms:StopReplicationTask

Prefixo do serviço	Ações
	dms:TestConnection dms:UpdateSubscriptionsToEventBridge
docdb-elastic	docdb-elastic:CopyClusterSnapshot docdb-elastic:DeleteCluster docdb-elastic:DeleteClusterSnapshot docdb-elastic:GetCluster docdb-elastic:GetClusterSnapshot docdb-elastic:ListClusters docdb-elastic:ListClusterSnapshots docdb-elastic:RestoreClusterFromSnapshot docdb-elastic:StartCluster docdb-elastic:StopCluster docdb-elastic:UpdateCluster

Prefixo do serviço	Ações
ds	ds:AcceptSharedDirectory ds:AddIpRoutes ds:AddRegion ds:CancelSchemaExtension ds:ConnectDirectory ds:CreateAlias ds:CreateComputer ds:CreateConditionalForwarder ds:CreateDirectory ds:CreateLogSubscription ds:CreateMicrosoftAD ds:CreateSnapshot ds:CreateTrust ds:DeleteConditionalForwarder ds:DeleteDirectory ds:DeleteLogSubscription ds:DeleteSnapshot ds:DeleteTrust ds:DeregisterCertificate ds:DeregisterEventTopic ds:DescribeCertificate

Prefixo do serviço	Ações
	ds:DescribeClientAuthenticationSettings ds:DescribeConditionalForwarders ds:DescribeDirectories ds:DescribeDomainControllers ds:DescribeEventTopics ds:DescribeLDAPSSettings ds:DescribeRegions ds:DescribeSettings ds:DescribeSharedDirectories ds:DescribeSnapshots ds:DescribeTrusts ds:DescribeUpdateDirectory ds:DisableClientAuthentication ds:DisableLDAPS ds:DisableRadius ds:DisableSso ds:EnableClientAuthentication ds:EnableLDAPS ds:EnableRadius ds:EnableSso ds:GetDirectoryLimits

Prefixo do serviço	Ações
	ds:GetSnapshotLimits ds:ListCertificates ds:ListIpRoutes ds:ListLogSubscriptions ds:ListSchemaExtensions ds:RegisterCertificate ds:RegisterEventTopic ds:RejectSharedDirectory ds:RemoveIpRoutes ds:RemoveRegion ds:ResetUserPassword ds:RestoreFromSnapshot ds:ShareDirectory ds:StartSchemaExtension ds:UnshareDirectory ds:UpdateConditionalForwarder ds:UpdateDirectorySetup ds:UpdateNumberOfDomainControllers ds:UpdateRadius ds:UpdateSettings ds:UpdateTrust

Prefixo do serviço	Ações
	ds:VerifyTrust

Prefixo do serviço	Ações
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
	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
ebs	ebs:CompleteSnapshot ebs:StartSnapshot

Prefixo do serviço	Ações
ec2	ec2:AcceptAddressTransfer
	ec2:AcceptReservedInstancesExchangeQuote
	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:AssociateClientVpnTargetNetwork
	ec2:AssociateDhcpOptions
	ec2:AssociateEnclaveCertificateIamRole
	ec2:AssociateIamInstanceProfile
	ec2:AssociateInstanceEventWindow

Prefixo do serviço	Ações
	ec2:AssociateIpamByoasn
	ec2:AssociateIpamResourceDiscovery
	ec2:AssociateNatGatewayAddress
	ec2:AssociateRouteTable
	ec2:AssociateSubnetCidrBlock
	ec2:AssociateTransitGatewayMulticastDomain
	ec2:AssociateTransitGatewayPolicyTable
	ec2:AssociateTransitGatewayRouteTable
	ec2:AssociateTrunkInterface
	ec2:AssociateVpcCidrBlock
	ec2:AttachClassicLinkVpc
	ec2:AttachInternetGateway
	ec2:AttachNetworkInterface
	ec2:AttachVerifiedAccessTrustProvider
	ec2:AttachVolume
	ec2:AttachVpnGateway
	ec2:AuthorizeClientVpnIngress
	ec2:AuthorizeSecurityGroupEgress
	ec2:AuthorizeSecurityGroupIngress
	ec2:BundleInstance
	ec2:CancelBundleTask

Prefixo do serviço	Ações
	ec2:CancelCapacityReservation
	ec2:CancelCapacityReservationFleets
	ec2:CancelConversionTask
	ec2:CancelExportTask
	ec2:CancelImageLaunchPermission
	ec2:CancelImportTask
	ec2:CancelReservedInstancesListing
	ec2:CancelSpotFleetRequests
	ec2:CancelSpotInstanceRequests
	ec2:ConfirmProductInstance
	ec2:CopyFpgaImage
	ec2:CopyImage
	ec2:CopySnapshot
	ec2:CreateCapacityReservation
	ec2:CreateCapacityReservationFleet
	ec2:CreateCarrierGateway
	ec2:CreateClientVpnEndpoint
	ec2:CreateClientVpnRoute
	ec2:CreateCoipCidr
	ec2:CreateCoipPool
	ec2:CreateCustomerGateway

Prefixo do serviço	Ações
	ec2:CreateDefaultSubnet
	ec2:CreateDefaultVpc
	ec2:CreateDhcpOptions
	ec2:CreateEgressOnlyInternetGateway
	ec2:CreateFleet
	ec2:CreateFlowLogs
	ec2:CreateFpgaImage
	ec2:CreateImage
	ec2:CreateInstanceConnectEndpoint
	ec2:CreateInstanceEventWindow
	ec2:CreateInstanceExportTask
	ec2:CreateInternetGateway
	ec2:CreateIpam
	ec2:CreateIpamPool
	ec2:CreateIpamResourceDiscovery
	ec2:CreateIpamScope
	ec2:CreateKeyPair
	ec2:CreateLaunchTemplate
	ec2:CreateLaunchTemplateVersion
	ec2:CreateLocalGatewayRoute
	ec2:CreateLocalGatewayRouteTable

Prefixo do serviço	Ações
	ec2:CreateLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:CreateLocalGatewayRouteTableVpcAssociation
	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:CreateRouteTable
	ec2:CreateSecurityGroup
	ec2:CreateSnapshot
	ec2:CreateSnapshots

Prefixo do serviço	Ações
	ec2:CreateSpotDatafeedSubscription
	ec2:CreateStoreImageTask
	ec2:CreateSubnet
	ec2:CreateSubnetCidrReservation
	ec2:CreateTrafficMirrorFilter
	ec2:CreateTrafficMirrorFilterRule
	ec2:CreateTrafficMirrorSession
	ec2:CreateTrafficMirrorTarget
	ec2:CreateTransitGateway
	ec2:CreateTransitGatewayConnect
	ec2:CreateTransitGatewayConnectPeer
	ec2:CreateTransitGatewayMulticastDomain
	ec2:CreateTransitGatewayPeeringAttachment
	ec2:CreateTransitGatewayPolicyTable
	ec2:CreateTransitGatewayPrefixListReference
	ec2:CreateTransitGatewayRoute
	ec2:CreateTransitGatewayRouteTable
	ec2:CreateTransitGatewayRouteTableAnnouncement
	ec2:CreateTransitGatewayVpcAttachment
	ec2:CreateVerifiedAccessEndpoint
	ec2:CreateVerifiedAccessGroup

Prefixo do serviço	Ações
	ec2:CreateVerifiedAccessInstance
	ec2:CreateVerifiedAccessTrustProvider
	ec2:CreateVolume
	ec2:CreateVpc
	ec2:CreateVpcEndpoint
	ec2:CreateVpcEndpointConnectionNotification
	ec2:CreateVpcEndpointServiceConfiguration
	ec2:CreateVpcPeeringConnection
	ec2:CreateVpnConnection
	ec2:CreateVpnConnectionRoute
	ec2:CreateVpnGateway
	ec2:DeleteCarrierGateway
	ec2:DeleteClientVpnEndpoint
	ec2:DeleteClientVpnRoute
	ec2:DeleteCoipCidr
	ec2:DeleteCoipPool
	ec2:DeleteCustomerGateway
	ec2:DeleteDhcpOptions
	ec2:DeleteEgressOnlyInternetGateway
	ec2:DeleteFleets
	ec2:DeleteFlowLogs

Prefixo do serviço	Ações
	ec2:DeleteFpgaImage
	ec2:DeleteInstanceConnectEndpoint
	ec2:DeleteInstanceEventWindow
	ec2:DeleteInternetGateway
	ec2:DeleteIpam
	ec2:DeleteIpamPool
	ec2:DeleteIpamResourceDiscovery
	ec2:DeleteIpamScope
	ec2:DeleteKeyPair
	ec2:DeleteLaunchTemplate
	ec2:DeleteLaunchTemplateVersions
	ec2:DeleteLocalGatewayRoute
	ec2:DeleteLocalGatewayRouteTable
	ec2:DeleteLocalGatewayRouteTableVirtualInterfaceGroupAssociation
	ec2:DeleteLocalGatewayRouteTableVpcAssociation
	ec2:DeleteManagedPrefixList
	ec2:DeleteNatGateway
	ec2:DeleteNetworkAcl
	ec2:DeleteNetworkAclEntry
	ec2:DeleteNetworkInsightsAccessScope

Prefixo do serviço	Ações
	ec2:DeleteNetworkInsightsAccessScopeAnalysis
	ec2:DeleteNetworkInsightsAnalysis
	ec2:DeleteNetworkInsightsPath
	ec2:DeleteNetworkInterface
	ec2:DeleteNetworkInterfacePermission
	ec2:DeletePlacementGroup
	ec2:DeletePublicIpv4Pool
	ec2:DeleteQueuedReservedInstances
	ec2:DeleteRoute
	ec2:DeleteRouteTable
	ec2:DeleteSecurityGroup
	ec2:DeleteSnapshot
	ec2:DeleteSpotDatafeedSubscription
	ec2:DeleteSubnet
	ec2:DeleteSubnetCidrReservation
	ec2:DeleteTrafficMirrorFilter
	ec2:DeleteTrafficMirrorFilterRule
	ec2:DeleteTrafficMirrorSession
	ec2:DeleteTrafficMirrorTarget
	ec2:DeleteTransitGateway
	ec2:DeleteTransitGatewayConnect

Prefixo do serviço	Ações
	ec2:DeleteTransitGatewayConnectPeer
	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:DeleteVpcEndpointConnectionNotifications
	ec2:DeleteVpcEndpoints
	ec2:DeleteVpcEndpointServiceConfigurations
	ec2:DeleteVpcPeeringConnection
	ec2:DeleteVpnConnection
	ec2:DeleteVpnConnectionRoute

Prefixo do serviço	Ações
	ec2:DeleteVpnGateway
	ec2:DeprovisionByoipCidr
	ec2:DeprovisionIpamByoasn
	ec2:DeprovisionIpamPoolCidr
	ec2:DeprovisionPublicIpv4PoolCidr
	ec2:DeregisterImage
	ec2:DeregisterInstanceEventNotificationAttributes
	ec2:DeregisterTransitGatewayMulticastGroupMembers
	ec2:DeregisterTransitGatewayMulticastGroupSources
	ec2:DescribeAccountAttributes
	ec2:DescribeAddresses
	ec2:DescribeAddressesAttribute
	ec2:DescribeAddressTransfers
	ec2:DescribeAggregateIdFormat
	ec2:DescribeAvailabilityZones
	ec2:DescribeAwsNetworkPerformanceMetricSubscriptions
	ec2:DescribeBundleTasks
	ec2:DescribeByoipCidrs
	ec2:DescribeCapacityReservationFleets
	ec2:DescribeCapacityReservations
	ec2:DescribeCarrierGateways

Prefixo do serviço	Ações
	ec2:DescribeClassicLinkInstances
	ec2:DescribeClientVpnAuthorizationRules
	ec2:DescribeClientVpnConnections
	ec2:DescribeClientVpnEndpoints
	ec2:DescribeClientVpnRoutes
	ec2:DescribeClientVpnTargetNetworks
	ec2:DescribeCoipPools
	ec2:DescribeConversionTasks
	ec2:DescribeCustomerGateways
	ec2:DescribeDhcpOptions
	ec2:DescribeEgressOnlyInternetGateways
	ec2:DescribeElasticGpus
	ec2:DescribeExportImageTasks
	ec2:DescribeExportTasks
	ec2:DescribeFastLaunchImages
	ec2:DescribeFastSnapshotRestores
	ec2:DescribeFleetHistory
	ec2:DescribeFleetInstances
	ec2:DescribeFleets
	ec2:DescribeFlowLogs
	ec2:DescribeFpgaImageAttribute

Prefixo do serviço	Ações
	ec2:DescribeFpgaImages
	ec2:DescribeHostReservationOfferings
	ec2:DescribeHostReservations
	ec2:DescribeHosts
	ec2:DescribeIamInstanceProfileAssociations
	ec2:DescribeIdentityIdFormat
	ec2:DescribeIdFormat
	ec2:DescribeImageAttribute
	ec2:DescribeImages
	ec2:DescribeImportImageTasks
	ec2:DescribeImportSnapshotTasks
	ec2:DescribeInstanceAttribute
	ec2:DescribeInstanceConnectEndpoints
	ec2:DescribeInstanceCreditSpecifications
	ec2:DescribeInstanceEventNotificationAttributes
	ec2:DescribeInstanceEventWindows
	ec2:DescribeInstances
	ec2:DescribeInstanceStatus
	ec2:DescribeInstanceTopology
	ec2:DescribeInstanceTypeOfferings
	ec2:DescribeInstanceTypes

Prefixo do serviço	Ações
	ec2:DescribeInternetGateways
	ec2:DescribeIpamByoasn
	ec2:DescribeIpamPools
	ec2:DescribeIpamResourceDiscoveries
	ec2:DescribeIpamResourceDiscoveryAssociations
	ec2:DescribeIpams
	ec2:DescribeIpamScopes
	ec2:DescribeIpv6Pools
	ec2:DescribeKeyPairs
	ec2:DescribeLaunchTemplates
	ec2:DescribeLaunchTemplateVersions
	ec2:DescribeLocalGatewayRouteTables
	ec2:DescribeLocalGatewayRouteTableVirtualInterfaceGroupAssociations
	ec2:DescribeLocalGatewayRouteTableVpcAssociations
	ec2:DescribeLocalGateways
	ec2:DescribeLocalGatewayVirtualInterfaceGroups
	ec2:DescribeLocalGatewayVirtualInterfaces
	ec2:DescribeLockedSnapshots
	ec2:DescribeMacHosts
	ec2:DescribeManagedPrefixLists

Prefixo do serviço	Ações
	ec2:DescribeMovingAddresses
	ec2:DescribeNatGateways
	ec2:DescribeNetworkAcls
	ec2:DescribeNetworkInsightsAccessScopeAnalyses
	ec2:DescribeNetworkInsightsAccessScopes
	ec2:DescribeNetworkInsightsAnalyses
	ec2:DescribeNetworkInsightsPaths
	ec2:DescribeNetworkInterfaceAttribute
	ec2:DescribeNetworkInterfacePermissions
	ec2:DescribeNetworkInterfaces
	ec2:DescribePlacementGroups
	ec2:DescribePrefixLists
	ec2:DescribePrincipalIdFormat
	ec2:DescribePublicIpv4Pools
	ec2:DescribeRegions
	ec2:DescribeReplaceRootVolumeTasks
	ec2:DescribeReservedInstances
	ec2:DescribeReservedInstancesListings
	ec2:DescribeReservedInstancesModifications
	ec2:DescribeReservedInstancesOfferings
	ec2:DescribeRouteTables

Prefixo do serviço	Ações
	ec2:DescribeScheduledInstanceAvailability
	ec2:DescribeScheduledInstances
	ec2:DescribeSecurityGroupReferences
	ec2:DescribeSecurityGroupRules
	ec2:DescribeSecurityGroups
	ec2:DescribeSnapshotAttribute
	ec2:DescribeSnapshots
	ec2:DescribeSnapshotTierStatus
	ec2:DescribeSpotDatafeedSubscription
	ec2:DescribeSpotFleetInstances
	ec2:DescribeSpotFleetRequestHistory
	ec2:DescribeSpotFleetRequests
	ec2:DescribeSpotInstanceRequests
	ec2:DescribeSpotPriceHistory
	ec2:DescribeStaleSecurityGroups
	ec2:DescribeStoreImageTasks
	ec2:DescribeSubnets
	ec2:DescribeTrafficMirrorFilters
	ec2:DescribeTrafficMirrorSessions
	ec2:DescribeTrafficMirrorTargets
	ec2:DescribeTransitGatewayAttachments

Prefixo do serviço	Ações
	ec2:DescribeTransitGatewayConnectPeers
	ec2:DescribeTransitGatewayConnects
	ec2:DescribeTransitGatewayMulticastDomains
	ec2:DescribeTransitGatewayPeeringAttachments
	ec2:DescribeTransitGatewayPolicyTables
	ec2:DescribeTransitGatewayRouteTableAnnouncements
	ec2:DescribeTransitGatewayRouteTables
	ec2:DescribeTransitGateways
	ec2:DescribeTransitGatewayVpcAttachments
	ec2:DescribeTrunkInterfaceAssociations
	ec2:DescribeVerifiedAccessEndpoints
	ec2:DescribeVerifiedAccessGroups
	ec2:DescribeVerifiedAccessInstanceLoggingConfigurations
	ec2:DescribeVerifiedAccessInstances
	ec2:DescribeVerifiedAccessTrustProviders
	ec2:DescribeVolumeAttribute
	ec2:DescribeVolumes
	ec2:DescribeVolumesModifications
	ec2:DescribeVolumeStatus
	ec2:DescribeVpcAttribute
	ec2:DescribeVpcClassicLink

Prefixo do serviço	Ações
	ec2:DescribeVpcClassicLinkDnsSupport
	ec2:DescribeVpcEndpointConnectionNotifications
	ec2:DescribeVpcEndpointConnections
	ec2:DescribeVpcEndpoints
	ec2:DescribeVpcEndpointServiceConfigurations
	ec2:DescribeVpcEndpointServicePermissions
	ec2:DescribeVpcEndpointServices
	ec2:DescribeVpcPeeringConnections
	ec2:DescribeVpcs
	ec2:DescribeVpnConnections
	ec2:DescribeVpnGateways
	ec2:DetachClassicLinkVpc
	ec2:DetachInternetGateway
	ec2:DetachNetworkInterface
	ec2:DetachVerifiedAccessTrustProvider
	ec2:DetachVolume
	ec2:DetachVpnGateway
	ec2:DisableAddressTransfer
	ec2:DisableAwsNetworkPerformanceMetricSubscription
	ec2:DisableEbsEncryptionByDefault
	ec2:DisableFastLaunch

Prefixo do serviço	Ações
	ec2:DisableFastSnapshotRestores
	ec2:DisableImage
	ec2:DisableImageBlockPublicAccess
	ec2:DisableImageDeprecation
	ec2:DisableIpamOrganizationAdminAccount
	ec2:DisableSerialConsoleAccess
	ec2:DisableSnapshotBlockPublicAccess
	ec2:DisableTransitGatewayRouteTablePropagation
	ec2:DisableVgwRoutePropagation
	ec2:DisableVpcClassicLink
	ec2:DisableVpcClassicLinkDnsSupport
	ec2:DisassociateAddress
	ec2:DisassociateClientVpnTargetNetwork
	ec2:DisassociateEnclaveCertificateIamRole
	ec2:DisassociateIamInstanceProfile
	ec2:DisassociateInstanceEventWindow
	ec2:DisassociateIpamByoasn
	ec2:DisassociateIpamResourceDiscovery
	ec2:DisassociateNatGatewayAddress
	ec2:DisassociateRouteTable
	ec2:DisassociateSubnetCidrBlock

Prefixo do serviço	Ações
	ec2:DisassociateTransitGatewayMulticastDomain
	ec2:DisassociateTransitGatewayPolicyTable
	ec2:DisassociateTransitGatewayRouteTable
	ec2:DisassociateTrunkInterface
	ec2:DisassociateVpcCidrBlock
	ec2:EnableAddressTransfer
	ec2:EnableAwsNetworkPerformanceMetricSubscription
	ec2:EnableEbsEncryptionByDefault
	ec2:EnableFastLaunch
	ec2:EnableFastSnapshotRestores
	ec2:EnableImage
	ec2:EnableImageBlockPublicAccess
	ec2:EnableImageDeprecation
	ec2:EnableIpamOrganizationAdminAccount
	ec2:EnableReachabilityAnalyzerOrganizationSharing
	ec2:EnableSerialConsoleAccess
	ec2:EnableSnapshotBlockPublicAccess
	ec2:EnableTransitGatewayRouteTablePropagation
	ec2:EnableVgwRoutePropagation
	ec2:EnableVolumeIO
	ec2:EnableVpcClassicLink

Prefixo do serviço	Ações
	ec2:EnableVpcClassicLinkDnsSupport
	ec2:ExportClientVpnClientCertificateRevocationList
	ec2:ExportClientVpnClientConfiguration
	ec2:ExportImage
	ec2:ExportTransitGatewayRoutes
	ec2:GetAssociatedEnclaveCertificateIamRoles
	ec2:GetAssociatedIpv6PoolCidrs
	ec2:GetAwsNetworkPerformanceData
	ec2:GetCapacityReservationUsage
	ec2:GetCoipPoolUsage
	ec2:GetConsoleOutput
	ec2:GetConsoleScreenshot
	ec2:GetDefaultCreditSpecification
	ec2:GetEbsDefaultKmsKeyId
	ec2:GetEbsEncryptionByDefault
	ec2:GetFlowLogsIntegrationTemplate
	ec2:GetGroupsForCapacityReservation
	ec2:GetHostReservationPurchasePreview
	ec2:GetImageBlockPublicAccessState
	ec2:GetInstanceMetadataDefaults
	ec2:GetInstanceTypesFromInstanceRequirements

Prefixo do serviço	Ações
	ec2:GetInstanceUefiData
	ec2:GetIpamAddressHistory
	ec2:GetIpamDiscoveredAccounts
	EC2: Obtenha endereços públicos descobertos pelo IPAM
	ec2:GetIpamDiscoveredResourceCidrs
	ec2:GetIpamPoolAllocations
	ec2:GetIpamPoolCidrs
	ec2:GetIpamResourceCidrs
	ec2:GetLaunchTemplateData
	ec2:GetManagedPrefixListAssociations
	ec2:GetManagedPrefixListEntries
	ec2:GetNetworkInsightsAccessScopeAnalysisFindings
	ec2:GetNetworkInsightsAccessScopeContent
	ec2:GetPasswordData
	ec2:GetReservedInstancesExchangeQuote
	ec2:GetSecurityGroupsForVpc
	ec2:GetSerialConsoleAccessStatus
	ec2:GetSnapshotBlockPublicAccessState
	ec2:GetSpotPlacementScores
	ec2:GetSubnetCidrReservations
	ec2:GetTransitGatewayAttachmentPropagations

Prefixo do serviço	Ações
	ec2:GetTransitGatewayMulticastDomainAssociations
	ec2:GetTransitGatewayPolicyTableAssociations
	ec2:GetTransitGatewayPolicyTableEntries
	ec2:GetTransitGatewayPrefixListReferences
	ec2:GetTransitGatewayRouteTableAssociations
	ec2:GetTransitGatewayRouteTablePropagations
	ec2:GetVerifiedAccessEndpointPolicy
	ec2:GetVerifiedAccessGroupPolicy
	ec2:GetVpnConnectionDeviceSampleConfiguration
	ec2:GetVpnConnectionDeviceTypes
	ec2:GetVpnTunnelReplacementStatus
	ec2:ImportClientVpnClientCertificateRevocationList
	ec2:ImportImage
	ec2:ImportInstance
	ec2:ImportKeyPair
	ec2:ImportSnapshot
	ec2:ImportVolume
	ec2:ListImagesInRecycleBin
	ec2:ListSnapshotsInRecycleBin
	ec2:LockSnapshot
	ec2:ModifyAddressAttribute

Prefixo do serviço	Ações
	ec2:ModifyAvailabilityZoneGroup
	ec2:ModifyCapacityReservation
	ec2:ModifyCapacityReservationFleet
	ec2:ModifyClientVpnEndpoint
	ec2:ModifyDefaultCreditSpecification
	ec2:ModifyEbsDefaultKmsKeyId
	ec2:ModifyFleet
	ec2:ModifyFpgaImageAttribute
	ec2:ModifyHosts
	ec2:ModifyIdentityIdFormat
	ec2:ModifyIdFormat
	ec2:ModifyImageAttribute
	ec2:ModifyInstanceAttribute
	ec2:ModifyInstanceCapacityReservationAttributes
	ec2:ModifyInstanceCreditSpecification
	ec2:ModifyInstanceEventStartTime
	ec2:ModifyInstanceEventWindow
	ec2:ModifyInstanceMaintenanceOptions
	ec2:ModifyInstanceMetadataDefaults
	ec2:ModifyInstanceMetadataOptions
	ec2:ModifyInstancePlacement

Prefixo do serviço	Ações
	ec2:ModifyIpam
	ec2:ModifyIpamPool
	ec2:ModifyIpamResourceCidr
	ec2:ModifyIpamResourceDiscovery
	ec2:ModifyIpamScope
	ec2:ModifyLaunchTemplate
	ec2:ModifyLocalGatewayRoute
	ec2:ModifyManagedPrefixList
	ec2:ModifyNetworkInterfaceAttribute
	ec2:ModifyPrivateDnsNameOptions
	ec2:ModifyReservedInstances
	ec2:ModifySecurityGroupRules
	ec2:ModifySnapshotAttribute
	ec2:ModifySnapshotTier
	ec2:ModifySpotFleetRequest
	ec2:ModifySubnetAttribute
	ec2:ModifyTrafficMirrorFilterNetworkServices
	ec2:ModifyTrafficMirrorFilterRule
	ec2:ModifyTrafficMirrorSession
	ec2:ModifyTransitGateway
	ec2:ModifyTransitGatewayPrefixListReference

Prefixo do serviço	Ações
	ec2:ModifyTransitGatewayVpcAttachment
	ec2:ModifyVerifiedAccessEndpoint
	ec2:ModifyVerifiedAccessEndpointPolicy
	ec2:ModifyVerifiedAccessGroup
	ec2:ModifyVerifiedAccessGroupPolicy
	ec2:ModifyVerifiedAccessInstance
	ec2:ModifyVerifiedAccessInstanceLoggingConfiguration
	ec2:ModifyVerifiedAccessTrustProvider
	ec2:ModifyVolume
	ec2:ModifyVolumeAttribute
	ec2:ModifyVpcAttribute
	ec2:ModifyVpcEndpoint
	ec2:ModifyVpcEndpointConnectionNotification
	ec2:ModifyVpcEndpointServiceConfiguration
	ec2:ModifyVpcEndpointServicePayerResponsibility
	ec2:ModifyVpcEndpointServicePermissions
	ec2:ModifyVpcPeeringConnectionOptions
	ec2:ModifyVpcTenancy
	ec2:ModifyVpnConnection
	ec2:ModifyVpnConnectionOptions
	ec2:ModifyVpnTunnelCertificate

Prefixo do serviço	Ações
	ec2:ModifyVpnTunnelOptions
	ec2:MonitorInstances
	ec2:MoveAddressToVpc
	ec2:MoveByoipCidrToIpam
	ec2:ProvisionByoipCidr
	ec2:ProvisionIpamByoasn
	ec2:ProvisionIpamPoolCidr
	ec2:ProvisionPublicIpv4PoolCidr
	ec2:PurchaseHostReservation
	ec2:PurchaseReservedInstancesOffering
	ec2:PurchaseScheduledInstances
	ec2:RebootInstances
	ec2:RegisterImage
	ec2:RegisterInstanceEventNotificationAttributes
	ec2:RegisterTransitGatewayMulticastGroupMembers
	ec2:RegisterTransitGatewayMulticastGroupSources
	ec2:RejectTransitGatewayMulticastDomainAssociations
	ec2:RejectTransitGatewayPeeringAttachment
	ec2:RejectTransitGatewayVpcAttachment
	ec2:RejectVpcEndpointConnections
	ec2:RejectVpcPeeringConnection

Prefixo do serviço	Ações
	ec2:ReleaseAddress
	ec2:ReleaseHosts
	ec2:ReleaseIpamPoolAllocation
	ec2:ReplaceIamInstanceProfileAssociation
	ec2:ReplaceNetworkAclAssociation
	ec2:ReplaceNetworkAclEntry
	ec2:ReplaceRoute
	ec2:ReplaceRouteTableAssociation
	ec2:ReplaceTransitGatewayRoute
	ec2:ReplaceVpnTunnel
	ec2:ReportInstanceStatus
	ec2:RequestSpotFleet
	ec2:RequestSpotInstances
	ec2:ResetAddressAttribute
	ec2:ResetEbsDefaultKmsKeyId
	ec2:ResetFpgaImageAttribute
	ec2:ResetImageAttribute
	ec2:ResetInstanceAttribute
	ec2:ResetNetworkInterfaceAttribute
	ec2:ResetSnapshotAttribute
	ec2:RestoreAddressToClassic

Prefixo do serviço	Ações
	ec2:RestoreImageFromRecycleBin
	ec2:RestoreManagedPrefixListVersion
	ec2:RestoreSnapshotFromRecycleBin
	ec2:RestoreSnapshotTier
	ec2:RevokeClientVpnIngress
	ec2:RevokeSecurityGroupEgress
	ec2:RevokeSecurityGroupIngress
	ec2:RunInstances
	ec2:RunScheduledInstances
	ec2:SearchLocalGatewayRoutes
	ec2:SearchTransitGatewayMulticastGroups
	ec2:SearchTransitGatewayRoutes
	ec2:SendDiagnosticInterrupt
	ec2:StartInstances
	ec2:StartNetworkInsightsAccessScopeAnalysis
	ec2:StartNetworkInsightsAnalysis
	ec2:StartVpcEndpointServicePrivateDnsVerification
	ec2:StopInstances
	ec2:TerminateClientVpnConnections
	ec2:TerminateInstances
	ec2:UnassignIpv6Addresses

Prefixo do serviço	Ações
	ec2:UnassignPrivateIpAddresses ec2:UnassignPrivateNatGatewayAddress ec2:UnlockSnapshot ec2:UnmonitorInstances ec2:UpdateSecurityGroupRuleDescriptionsEgress ec2:UpdateSecurityGroupRuleDescriptionsIngress ec2:WithdrawByoipCidr

Prefixo do serviço	Ações
ecr	ecr:BatchCheckLayerAvailability ecr:BatchDeleteImage ecr:BatchGetImage ecr:BatchGetRepositoryScanningConfiguration ecr:CompleteLayerUpload ecr:CreatePullThroughCacheRule ecr:CreateRepository ecr:CreateRepositoryCreationTemplate ecr:DeleteLifecyclePolicy ecr:DeletePullThroughCacheRule ecr:DeleteRegistryPolicy ecr:DeleteRepository ecr:DeleteRepositoryCreationTemplate ecr:DeleteRepositoryPolicy ecr:DescribeImageReplicationStatus ecr:DescribeImages ecr:DescribeImageScanFindings ecr:DescribePullThroughCacheRules ecr:DescribeRegistry ecr:DescribeRepositories ecr:GetAuthorizationToken

Prefixo do serviço	Ações
	ecr:GetDownloadUrlForLayer ecr:GetLifecyclePolicy ecr:GetLifecyclePolicyPreview ecr:GetRegistryPolicy ecr:GetRegistryScanningConfiguration ecr:GetRepositoryPolicy ecr:InitiateLayerUpload ecr:ListImages ecr:PutImage ecr:PutImageScanningConfiguration ecr:PutRegistryPolicy ecr:PutRegistryScanningConfiguration ecr:PutReplicationConfiguration ecr:StartImageScan ecr:StartLifecyclePolicyPreview ecr:UpdatePullThroughCacheRule ecr:UploadLayerPart ecr:ValidatePullThroughCacheRule

Prefixo do serviço	Ações
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
ecs	ecs:CreateCapacityProvider ecs:CreateCluster ecs:CreateService ecs:CreateTaskSet ecs:DeleteAccountSetting ecs:DeleteAttributes ecs:DeleteCapacityProvider ecs:DeleteCluster ecs:DeleteService ecs:DeleteTaskDefinitions ecs:DeleteTaskSet ecs:DeregisterContainerInstance ecs:DeregisterTaskDefinition ecs:DescribeCapacityProviders ecs:DescribeClusters ecs:DescribeContainerInstances ecs:DescribeServices ecs:DescribeTaskDefinition ecs:DescribeTasks ecs:DescribeTaskSets ecs:DiscoverPollEndpoint

Prefixo do serviço	Ações
	ecs:ExecuteCommand ecs:GetTaskProtection ecs:ListAccountSettings ecs:ListAttributes ecs:ListClusters ecs:ListContainerInstances ecs:ListServices ecs:ListServicesByNamespace ecs:ListTaskDefinitionFamilies ecs:ListTaskDefinitions ecs:ListTasks ecs:PutAccountSetting ecs:PutAccountSettingDefault ecs:PutAttributes ecs:PutClusterCapacityProviders ecs:RegisterContainerInstance ecs:RegisterTaskDefinition ecs:RunTask ecs:StartTask ecs:StopTask ecs:SubmitAttachmentStateChanges

Prefixo do serviço	Ações
	ecs:SubmitContainerStateChange ecs:SubmitTaskStateChange ecs:UpdateCapacityProvider ecs:UpdateCluster ecs:UpdateClusterSettings ecs:UpdateContainerAgent ecs:UpdateContainerInstancesState ecs:UpdateService ecs:UpdateServicePrimaryTaskSet ecs:UpdateTaskProtection ecs:UpdateTaskSet

Prefixo do serviço	Ações
eks	eks:AssociateAccessPolicy
	eks:AssociateEncryptionConfig
	eks:AssociateIdentityProviderConfig
	eks:CreateAccessEntry
	eks:CreateAddon
	eks:CreateCluster
	eks:CreateEksAnywhereSubscription
	eks:CreateFargateProfile
	eks:CreateNodegroup
	eks:DeleteAccessEntry
	eks:DeleteAddon
	eks:DeleteCluster
	eks:DeleteEksAnywhereSubscription
	eks:DeleteFargateProfile
	eks:DeleteNodegroup
	eks:DeletePodIdentityAssociation
	eks:DeregisterCluster
	eks:DescribeAccessEntry
	eks:DescribeAddon
	eks:DescribeAddonConfiguration
	eks:DescribeAddonVersions

Prefixo do serviço	Ações
	eks:DescribeCluster
	eks:DescribeEksAnywhereSubscription
	eks:DescribeFargateProfile
	eks:DescribeIdentityProviderConfig
	eks:DescribeInsight
	eks:DescribeNodegroup
	eks:DescribePodIdentityAssociation
	eks:DescribeUpdate
	eks:DisassociateAccessPolicy
	eks:DisassociateIdentityProviderConfig
	eks:ListAccessEntries
	eks:ListAccessPolicies
	eks:ListAddons
	eks:ListAssociatedAccessPolicies
	eks:ListClusters
	eks:ListEksAnywhereSubscriptions
	eks:ListFargateProfiles
	eks:ListIdentityProviderConfigs
	eks:ListInsights
	eks:ListNodegroups
	eks:ListPodIdentityAssociations

Prefixo do serviço	Ações
	eks:ListUpdates eks:RegisterCluster eks:UpdateAccessEntry eks:UpdateAddon eks:UpdateClusterConfig eks:UpdateClusterVersion eks:UpdateEksAnywhereSubscription eks:UpdateNodegroupConfig eks:UpdateNodegroupVersion eks:UpdatePodIdentityAssociation
elastic-inference	elastic-inference:DescribeAcceleratorOfferings elastic-inference:DescribeAccelerators elastic-inference:DescribeAcceleratorTypes

Prefixo do serviço	Ações
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
	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
	elasticache:DescribeServerlessCaches
	elasticache:DescribeServerlessCacheSnapshots
	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
	elasticache:ModifyUserGroup elasticache:PurchaseReservedCacheNodesOffering elasticache:RebalanceSlotsInGlobalReplicationGroup elasticache:RebootCacheCluster elasticache:ResetCacheParameterGroup elasticache:RevokeCacheSecurityGroupIngress elasticache:StartMigration elasticache:TestFailover elasticache:TestMigration

Prefixo do serviço	Ações
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:DescribeApplications elasticbeanstalk:DescribeApplicationVersions elasticbeanstalk:DescribeConfigurationOptions elasticbeanstalk:DescribeConfigurationSettings

Prefixo do serviço	Ações
	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
	elasticbeanstalk:UpdateConfigurationTemplate elasticbeanstalk:UpdateEnvironment elasticbeanstalk:ValidateConfigurationSettings

Prefixo do serviço	Ações
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:DescribeMountTargets
	elasticfilesystem:DescribeMountTargetSecurityGroups
	elasticfilesystem:DescribeReplicationConfigurations
	elasticfilesystem:ModifyMountTargetSecurityGroups
	elasticfilesystem:PutAccountPreferences
	elasticfilesystem:PutBackupPolicy

Prefixo do serviço	Ações
	elasticfilesystem:PutFileSystemPolicy elasticfilesystem:PutLifecycleConfiguration elasticfilesystem:UpdateFileSystem elasticfilesystem:UpdateFileSystemProtection

Prefixo do serviço	Ações
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:DeleteTargetGroup elasticloadbalancing:DeleteTrustStore

Prefixo do serviço	Ações
	elasticloadbalancing:DeregisterInstancesFromLoadBalancer elasticloadbalancing:DeregisterTargets elasticloadbalancing:DescribeAccountLimits elasticloadbalancing:DescribeInstanceHealth 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 elasticloadbalancing:DetachLoadBalancerFromSubnets elasticloadbalancing:DisableAvailabilityZonesForLoadBalancer elasticloadbalancing:EnableAvailabilityZonesForLoadBalancer

Prefixo do serviço	Ações
	elasticloadbalancing:GetTrustStoreCaCertificatesBundle elasticloadbalancing:GetTrustStoreRevocationContent 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 elasticloadbalancing:SetLoadBalancerPoliciesOfListener elasticloadbalancing:SetRulePriorities elasticloadbalancing:SetSecurityGroups elasticloadbalancing:SetSubnets

Prefixo do serviço	Ações
elastictranscoder	elastictranscoder:CancelJob
	elastictranscoder:CreateJob
	elastictranscoder:CreatePipeline
	elastictranscoder:CreatePreset
	elastictranscoder:DeletePipeline
	elastictranscoder:DeletePreset
	elastictranscoder:ListJobsByPipeline
	elastictranscoder:ListJobsByStatus
	elastictranscoder:ListPipelines
	elastictranscoder:ListPresets
	elastictranscoder:ReadJob
	elastictranscoder:ReadPipeline
	elastictranscoder:ReadPreset
	elastictranscoder:TestRole
	elastictranscoder:UpdatePipeline
	elastictranscoder:UpdatePipelineNotifications
	elastictranscoder:UpdatePipelineStatus

Prefixo do serviço	Ações
emr-containers	emr-containers:CancelJobRun emr-containers:CreateJobTemplate emr-containers:CreateManagedEndpoint emr-containers:CreateVirtualCluster emr-containers:DeleteJobTemplate emr-containers:DeleteManagedEndpoint emr-containers:DeleteVirtualCluster emr-containers:DescribeJobRun emr-containers:DescribeJobTemplate emr-containers:DescribeManagedEndpoint emr-containers:DescribeVirtualCluster emr-containers:GetManagedEndpointSessionCredentials emr-containers:ListJobRuns emr-containers:ListJobTemplates emr-containers:ListManagedEndpoints emr-containers:ListVirtualClusters emr-containers:StartJobRun

Prefixo do serviço	Ações
emr-serverless	emr-serverless:CancelJobRun
	emr-serverless:CreateApplication
	emr-serverless:DeleteApplication
	emr-serverless:GetApplication
	emr-serverless:GetDashboardForJobRun
	emr-serverless:GetJobRun
	emr-serverless:ListApplications
	emr-serverless:ListJobRuns
	emr-serverless:StartApplication
	emr-serverless:StartJobRun
	emr-serverless:StopApplication
	emr-serverless:UpdateApplication

Prefixo do serviço	Ações
es	es:AcceptInboundConnection es:AcceptInboundCrossClusterSearchConnection es:AssociatePackage es:AuthorizeVpcEndpointAccess es:CancelElasticsearchServiceSoftwareUpdate es:CancelServiceSoftwareUpdate es:CreateDomain es:CreateElasticsearchDomain es:CreateOutboundConnection es:CreateOutboundCrossClusterSearchConnection es:CreatePackage es:CreateVpcEndpoint es:DeleteDomain es:DeleteElasticsearchDomain es:DeleteElasticsearchServiceRole es:DeleteInboundConnection es:DeleteInboundCrossClusterSearchConnection es:DeleteOutboundConnection es:DeleteOutboundCrossClusterSearchConnection es:DeletePackage es:DeleteVpcEndpoint

Prefixo do serviço	Ações
	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 es:DescribeReservedElasticsearchInstances es:DescribeReservedInstanceOfferings

Prefixo do serviço	Ações
	es:DescribeReservedInstances es:DescribeVpcEndpoints es:DissociatePackage 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:ListInstanceTypeDetails es:ListPackagesForDomain es:ListScheduledActions es:ListVersions es:ListVpcEndpointAccess es:ListVpcEndpoints

Prefixo do serviço	Ações
	es:ListVpcEndpointsForDomain es:PurchaseReservedElasticsearchInstanceOffering es:PurchaseReservedInstanceOffering es:RejectInboundConnection es:RejectInboundCrossClusterSearchConnection es:RevokeVpcEndpointAccess es:StartDomainMaintenance es:StartElasticsearchServiceSoftwareUpdate es:StartServiceSoftwareUpdate es:UpdateDataSource es:UpdateDomainConfig es:UpdateElasticsearchDomainConfig es:UpdatePackage es:UpdateScheduledAction es:UpdateVpcEndpoint es:UpgradeDomain es:UpgradeElasticsearchDomain

Prefixo do serviço	Ações
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
	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
	events:PutTargets
	events:RemovePermission
	events:RemoveTargets
	events:StartReplay
	events:TestEventPattern
	events:UpdateApiDestination
	events:UpdateArchive
	events:UpdateConnection
	events:UpdateEndpoint

Prefixo do serviço	Ações
evidently	evidently:CreateExperiment evidently:CreateFeature evidently:CreateLaunch evidently:CreateProject evidently:CreateSegment evidently:DeleteExperiment evidently:DeleteFeature evidently:DeleteLaunch evidently:DeleteProject evidently:DeleteSegment evidently:GetExperiment evidently:GetExperimentResults evidently:GetFeature evidently:GetLaunch evidently:GetProject evidently:GetSegment evidently:ListExperiments evidently:ListFeatures evidently:ListLaunches evidently:ListProjects evidently:ListSegmentReferences

Prefixo do serviço	Ações
	evidently:ListSegments evidently:StartExperiment evidently:StartLaunch evidently:StopExperiment evidently:StopLaunch evidently:TestSegmentPattern evidently:UpdateExperiment evidently:UpdateFeature evidently:UpdateLaunch evidently:UpdateProject evidently:UpdateProjectDataDelivery

Prefixo do serviço	Ações
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
	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
	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
fis	fis:CreateExperimentTemplate fis:CreateTargetAccountConfiguration fis:DeleteExperimentTemplate fis:DeleteTargetAccountConfiguration fis:GetAction fis:GetExperiment fis:GetExperimentTargetAccountConfiguration fis:GetExperimentTemplate fis:GetTargetAccountConfiguration fis:GetTargetResourceType fis:ListActions fis:ListExperimentResolvedTargets fis:ListExperiments fis:ListExperimentTargetAccountConfigurations fis:ListExperimentTemplates fis:ListTargetAccountConfigurations fis:ListTargetResourceTypes fis:StartExperiment fis:StopExperiment fis:UpdateExperimentTemplate fis:UpdateTargetAccountConfiguration

Prefixo do serviço	Ações
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
	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
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:DeleteEventsByEventType
	frauddetector:DeleteEventType
	frauddetector:DeleteExternalModel

Prefixo do serviço	Ações
	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:GetDetectors
	frauddetector:GetDetectorVersion
	frauddetector:GetEntityTypes
	frauddetector:GetEvent
	frauddetector:GetEventPrediction
	frauddetector:GetEventPredictionMetadata
	frauddetector:GetEventTypes
	frauddetector:GetExternalModels
	frauddetector:GetKMSEncryptionKey

Prefixo do serviço	Ações
	frauddetector:GetLabels
	frauddetector:GetListElements
	frauddetector:GetListsMetadata
	frauddetector:GetModels
	frauddetector:GetModelVersion
	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
	frauddetector:UpdateList frauddetector:UpdateModel frauddetector:UpdateModelVersion frauddetector:UpdateModelVersionStatus frauddetector:UpdateRuleMetadata frauddetector:UpdateRuleVersion frauddetector:UpdateVariable

Prefixo do serviço	Ações
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
	fsx:DescribeFileSystemAliases fsx:DescribeFileSystems fsx:DescribeSharedVpcConfiguration fsx:DescribeSnapshots fsx:DescribeStorageVirtualMachines fsx:DescribeVolumes 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
gamelift	gamelift:AcceptMatch
	gamelift:ClaimGameServer
	gamelift:CreateAlias
	gamelift:CreateBuild
	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:DeleteFleet
	gamelift:DeleteFleetLocations

Prefixo do serviço	Ações
	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:DescribeEC2InstanceLimits
	gamelift:DescribeFleetAttributes
	gamelift:DescribeFleetCapacity
	gamelift:DescribeFleetEvents
	gamelift:DescribeFleetLocationAttributes
	gamelift:DescribeFleetLocationCapacity
	gamelift:DescribeFleetLocationUtilization

Prefixo do serviço	Ações
	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
	gamelift:DescribeScript
	gamelift:DescribeVpcPeeringAuthorizations
	gamelift:DescribeVpcPeeringConnections
	gamelift:GetComputeAccess
	gamelift:GetComputeAuthToken

Prefixo do serviço	Ações
	gamelift:GetGameSessionLogUrl
	gamelift:GetInstanceAccess
	gamelift:ListAliases
	gamelift:ListBuilds
	gamelift:ListCompute
	gamelift:ListFleets
	gamelift:ListGameServerGroups
	gamelift:ListGameServers
	gamelift:ListLocations
	gamelift:ListScripts
	gamelift:PutScalingPolicy
	gamelift:RegisterCompute
	gamelift:RegisterGameServer
	gamelift:RequestUploadCredentials
	gamelift:ResolveAlias
	gamelift:ResumeGameServerGroup
	gamelift:SearchGameSessions
	gamelift:StartFleetActions
	gamelift:StartGameSessionPlacement
	gamelift:StartMatchBackfill
	gamelift:StartMatchmaking

Prefixo do serviço	Ações
	gamelift:StopFleetActions gamelift:StopGameSessionPlacement gamelift:StopMatchmaking gamelift:SuspendGameServerGroup gamelift:UpdateAlias gamelift:UpdateBuild gamelift:UpdateFleetAttributes gamelift:UpdateFleetCapacity gamelift:UpdateFleetPortSettings gamelift:UpdateGameServer gamelift:UpdateGameServerGroup gamelift:UpdateGameSession gamelift:UpdateGameSessionQueue gamelift:UpdateMatchmakingConfiguration gamelift:UpdateRuntimeConfiguration gamelift:UpdateScript gamelift:ValidateMatchmakingRuleSet

Prefixo do serviço	Ações
geo	geo:AssociateTrackerConsumer geo:BatchDeleteDevicePositionHistory geo:BatchDeleteGeofence geo:BatchEvaluateGeofences geo:BatchGetDevicePosition geo:BatchPutGeofence geo:BatchUpdateDevicePosition geo:CalculateRoute geo:CalculateRouteMatrix geo:CreateGeofenceCollection geo:CreateMap geo:CreatePlaceIndex geo:CreateRouteCalculator geo:CreateTracker geo:DeleteGeofenceCollection geo:DeleteKey geo:DeleteMap geo:DeletePlaceIndex geo:DeleteRouteCalculator geo:DeleteTracker geo:DescribeGeofenceCollection

Prefixo do serviço	Ações
	geo:DescribeKey geo:DescribeMap geo:DescribePlaceIndex geo:DescribeRouteCalculator geo:DescribeTracker geo:DisassociateTrackerConsumer geo:GetDevicePosition geo:GetDevicePositionHistory geo:GetGeofence geo:GetMapGlyphs geo:GetMapSprites geo:GetMapStyleDescriptor geo:GetMapTile geo:GetPlace geo:ListDevicePositions geo:ListGeofenceCollections geo:ListGeofences geo:ListKeys geo:ListMaps geo:ListPlaceIndexes geo:ListRouteCalculators

Prefixo do serviço	Ações
	geo:ListTrackerConsumers geo:ListTrackers geo:PutGeofence geo:SearchPlaceIndexForPosition geo:SearchPlaceIndexForSuggestions geo:SearchPlaceIndexForText geo:UpdateGeofenceCollection geo:UpdateKey geo:UpdateMap geo:UpdatePlaceIndex geo:UpdateRouteCalculator geo:UpdateTracker

Prefixo do serviço	Ações
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
	glacier:ListParts glacier:ListProvisionedCapacity glacier:ListVaults glacier:PurchaseProvisionedCapacity glacier:SetDataRetrievalPolicy glacier:SetVaultAccessPolicy glacier:SetVaultNotifications glacier:UploadArchive glacier:UploadMultipartPart

Prefixo do serviço	Ações
grafana	grafana:AssociateLicense grafana:CreateWorkspace grafana:CreateWorkspaceApiKey grafana:DeleteWorkspace grafana:DeleteWorkspaceApiKey grafana:DescribeWorkspace grafana:DescribeWorkspaceAuthentication grafana:DescribeWorkspaceConfiguration grafana:DisassociateLicense grafana:ListPermissions grafana:ListVersions grafana:ListWorkspaces grafana:UpdatePermissions grafana:UpdateWorkspace grafana:UpdateWorkspaceAuthentication grafana:UpdateWorkspaceConfiguration

Prefixo do serviço	Ações
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
	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
	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
	greengrass:GetServiceRoleForAccount
	greengrass:GetSubscriptionDefinition
	greengrass:GetSubscriptionDefinitionVersion
	greengrass:GetThingRuntimeConfiguration
	greengrass:ListBulkDeploymentDetailedReports
	greengrass:ListBulkDeployments
	greengrass:ListClientDevicesAssociatedWithCoreDevice
	greengrass:ListComponents
	greengrass:ListComponentVersions
	greengrass:ListConnectorDefinitions
	greengrass:ListConnectorDefinitionVersions
	greengrass:ListCoreDefinitions
	greengrass:ListCoreDefinitionVersions
	greengrass:ListCoreDevices
	greengrass:ListDeployments
	greengrass:ListDeviceDefinitions
	greengrass:ListDeviceDefinitionVersions
	greengrass:ListEffectiveDeployments
	greengrass:ListFunctionDefinitions
	greengrass:ListFunctionDefinitionVersions
	greengrass:ListGroupCertificateAuthorities

Prefixo do serviço	Ações
	greengrass:ListGroups
	greengrass:ListGroupVersions
	greengrass:ListInstalledComponents
	greengrass:ListLoggerDefinitions
	greengrass:ListLoggerDefinitionVersions
	greengrass:ListResourceDefinitions
	greengrass:ListResourceDefinitionVersions
	greengrass:ListSubscriptionDefinitions
	greengrass:ListSubscriptionDefinitionVersions
	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
	greengrass:UpdateSubscriptionDefinition greengrass:UpdateThingRuntimeConfiguration

Prefixo do serviço	Ações
groundstation	groundstation:CancelContact groundstation:CreateConfig groundstation:CreateDataflowEndpointGroup groundstation:CreateEphemeris groundstation:CreateMissionProfile groundstation:DeleteConfig groundstation:DeleteDataflowEndpointGroup groundstation:DeleteEphemeris groundstation:DeleteMissionProfile groundstation:DescribeContact groundstation:DescribeEphemeris groundstation:GetConfig groundstation:GetDataflowEndpointGroup groundstation:GetMinuteUsage groundstation:GetMissionProfile groundstation:GetSatellite groundstation:ListConfigs groundstation:ListContacts groundstation:ListDataflowEndpointGroups groundstation:ListEphemerides groundstation:ListGroundStations

Prefixo do serviço	Ações
	groundstation:ListMissionProfiles groundstation:ListSatellites groundstation:RegisterAgent groundstation:ReserveContact groundstation:UpdateAgentStatus groundstation:UpdateConfig groundstation:UpdateEphemeris groundstation:UpdateMissionProfile

Prefixo do serviço	Ações
guardduty	guardduty:AcceptAdministratorInvitation
	guardduty:AcceptInvitation
	guardduty:ArchiveFindings
	guardduty:CreateDetector
	guardduty:CreateFilter
	guardduty:CreateIPSet
	guardduty:CreateMembers
	guardduty:CreatePublishingDestination
	guardduty:CreateSampleFindings
	guardduty:CreateThreatIntelSet
	guardduty:DeclineInvitations
	guardduty:DeleteDetector
	guardduty:DeleteFilter
	guardduty:DeleteInvitations
	guardduty:DeleteIPSet
	guardduty:DeleteMembers
	guardduty:DeletePublishingDestination
	guardduty:DeleteThreatIntelSet
	guardduty:DescribeMalwareScans
	guardduty:DescribeOrganizationConfiguration
	guardduty:DescribePublishingDestination

Prefixo do serviço	Ações
	guardduty:DisableOrganizationAdminAccount
	guardduty:DisassociateFromAdministratorAccount
	guardduty:DisassociateFromMasterAccount
	guardduty:DisassociateMembers
	guardduty:EnableOrganizationAdminAccount
	guardduty:GetAdministratorAccount
	guardduty:GetCoverageStatistics
	guardduty:GetDetector
	guardduty:GetFilter
	guardduty:GetFindings
	guardduty:GetFindingsStatistics
	guardduty:GetInvitationsCount
	guardduty:GetIPSet
	guardduty:GetMalwareScanSettings
	guardduty:GetMasterAccount
	guardduty:GetMemberDetectors
	guardduty:GetMembers
	guardduty:GetOrganizationStatistics
	guardduty:GetRemainingFreeTrialDays
	guardduty:GetThreatIntelSet
	guardduty:GetUsageStatistics

Prefixo do serviço	Ações
	guardduty:InviteMembers
	guardduty:ListCoverage
	guardduty:ListDetectors
	guardduty:ListFilters
	guardduty:ListFindings
	guardduty:ListInvitations
	guardduty:ListIPSets
	guardduty:ListMembers
	guardduty:ListOrganizationAdminAccounts
	guardduty:ListPublishingDestinations
	guardduty:ListThreatIntelSets
	guardduty:SendSecurityTelemetry
	guardduty:StartMalwareScan
	guardduty:StartMonitoringMembers
	guardduty:StopMonitoringMembers
	guardduty:UnarchiveFindings
	guardduty:UpdateDetector
	guardduty:UpdateFilter
	guardduty:UpdateFindingsFeedback
	guardduty:UpdateIPSet
	guardduty:UpdateMalwareScanSettings

Prefixo do serviço	Ações
	guardduty:UpdateMemberDetectors guardduty:UpdateOrganizationConfiguration guardduty:UpdatePublishingDestination guardduty:UpdateThreatIntelSet
healthlake	healthlake:CreateFHIRDatastore healthlake:CreateResource healthlake:DeleteFHIRDatastore healthlake:DeleteResource healthlake:DescribeFHIRDatastore healthlake:DescribeFHIRExportJob healthlake:DescribeFHIRImportJob healthlake:GetCapabilities healthlake:ListFHIRDatastores healthlake:ListFHIRExportJobs healthlake:ListFHIRImportJobs healthlake:ReadResource healthlake:SearchWithGet healthlake:SearchWithPost healthlake:StartFHIRExportJob healthlake:StartFHIRImportJob healthlake:UpdateResource

Prefixo do serviço	Ações
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
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
	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:DeleteServerCertificate iam:DeleteServiceLinkedRole iam:DeleteServiceSpecificCredential iam:DeleteSigningCertificate iam:DeleteSSHPublicKey

Prefixo do serviço	Ações
	iam:DeleteUser iam:DeleteUserPermissionsBoundary iam:DeleteUserPolicy iam:DeleteVirtualMFADevice iam:DetachGroupPolicy iam:DetachRolePolicy iam:DetachUserPolicy iam:EnableMFADevice iam:GenerateCredentialReport iam:GenerateOrganizationsAccessReport iam:GenerateServiceLastAccessedDetails iam:GetAccessKeyLastUsed iam:GetAccountAuthorizationDetails iam:GetAccountEmailAddress iam:GetAccountName iam:GetAccountPasswordPolicy iam:GetAccountSummary iam:GetCloudFrontPublicKey iam:GetContextKeysForCustomPolicy iam:GetContextKeysForPrincipalPolicy iam:GetCredentialReport

Prefixo do serviço	Ações
	iam:GetGroup iam:GetGroupPolicy iam:GetInstanceProfile iam:GetLoginProfile iam:GetMFADevice iam:GetOpenIDConnectProvider iam:GetOrganizationsAccessReport iam:GetPolicy iam:GetPolicyVersion iam:GetRole iam:GetRolePolicy iam:GetSAMLProvider iam:GetServerCertificate iam:GetServiceLastAccessedDetails iam:GetServiceLastAccessedDetailsWithEntities iam:GetServiceLinkedRoleDeletionStatus iam:GetSSHPublicKey iam:GetUser iam:GetUserPolicy iam:ListAccessKeys iam:ListAccountAliases

Prefixo do serviço	Ações
	iam:ListAttachedGroupPolicies iam:ListAttachedRolePolicies iam:ListAttachedUserPolicies iam:ListCloudFrontPublicKeys iam:ListEntitiesForPolicy iam:ListGroupPolicies iam:ListGroups iam:ListGroupsForUser iam:ListInstanceProfiles iam:ListInstanceProfilesForRole iam:ListMFADevices iam:ListOpenIDConnectProviders iam:ListPolicies iam:ListPoliciesGrantingServiceAccess iam:ListPolicyVersions iam:ListRolePolicies iam:ListRoles iam:ListSAMLProviders iam:ListServerCertificates iam:ListServiceSpecificCredentials iam:ListSigningCertificates

Prefixo do serviço	Ações
	iam:ListSSHPublicKeys iam:ListSTSRegionalEndpointsStatus iam:ListUserPolicies iam:ListUsers iam:ListVirtualMFADevices iam:PutGroupPolicy iam:PutRolePermissionsBoundary iam:PutRolePolicy iam:PutUserPermissionsBoundary iam:PutUserPolicy iam:RemoveClientIDFromOpenIDConnectProvider iam:RemoveRoleFromInstanceProfile iam:RemoveUserFromGroup iam:ResetServiceSpecificCredential iam:ResyncMFADevice iam:SetDefaultPolicyVersion iam:SetSecurityTokenServicePreferences iam:SetSTSRegionalEndpointStatus iam:SimulateCustomPolicy iam:SimulatePrincipalPolicy iam:UpdateAccessKey

Prefixo do serviço	Ações
	iam:UpdateAccountEmailAddress iam:UpdateAccountName iam:UpdateAccountPasswordPolicy iam:UpdateAssumeRolePolicy iam:UpdateCloudFrontPublicKey iam:UpdateGroup iam:UpdateLoginProfile iam:UpdateOpenIDConnectProviderThumbprint iam:UpdateRole iam:UpdateRoleDescription iam:UpdateSAMLProvider iam:UpdateServerCertificate iam:UpdateServiceSpecificCredential iam:UpdateSigningCertificate iam:UpdateSSHPublicKey iam:UpdateUser iam:UploadCloudFrontPublicKey iam:UploadServerCertificate iam:UploadSigningCertificate iam:UploadSSHPublicKey

Prefixo do serviço	Ações
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
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:GetComponentPolicy

Prefixo do serviço	Ações
	imagebuilder:GetContainerRecipePolicy imagebuilder:GetImagePolicy imagebuilder:GetImageRecipePolicy imagebuilder:GetLifecycleExecution imagebuilder:GetLifecyclePolicy imagebuilder:GetWorkflowExecution imagebuilder:GetWorkflowStepExecution imagebuilder:ImportComponent imagebuilder:ImportVmImage imagebuilder:ListComponentBuildVersions imagebuilder:ListComponents imagebuilder:ListContainerRecipes imagebuilder:ListDistributionConfigurations imagebuilder:ListImageBuildVersions imagebuilder:ListImagePackages imagebuilder:ListImagePipelineImages imagebuilder:ListImagePipelines imagebuilder:ListImageRecipes imagebuilder:ListImages imagebuilder:ListImageScanFindingAggregations imagebuilder:ListImageScanFindings

Prefixo do serviço	Ações
	imagebuilder:ListInfrastructureConfigurations imagebuilder:ListLifecycleExecutionResources imagebuilder:ListLifecycleExecutions imagebuilder:ListLifecyclePolicies imagebuilder:ListWaitingWorkflowSteps imagebuilder:ListWorkflowExecutions imagebuilder:ListWorkflows imagebuilder:ListWorkflowStepExecutions imagebuilder:PutComponentPolicy imagebuilder:PutContainerRecipePolicy imagebuilder:PutImagePolicy imagebuilder:PutImageRecipePolicy imagebuilder:SendWorkflowStepAction imagebuilder:StartImagePipelineExecution imagebuilder:StartResourceStateUpdate imagebuilder:UpdateDistributionConfiguration imagebuilder:UpdateImagePipeline imagebuilder:UpdateInfrastructureConfiguration

Prefixo do serviço	Ações
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
	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
inspector2	inspector2:AssociateMember inspector2:BatchGetAccountStatus inspector2:BatchGetCodeSnippet inspector2:BatchGetFindingDetails inspector2:BatchGetFreeTrialInfo inspector2:BatchGetMemberEc2DeepInspectionStatus inspector2:BatchUpdateMemberEc2DeepInspectionStatus inspector2:CancelFindingsReport inspector2:CancelSbomExport inspector2:CreateCisScanConfiguration inspector2:CreateFilter inspector2:CreateFindingsReport inspector2:CreateSbomExport inspector2:DeleteCisScanConfiguration inspector2:DeleteFilter inspector2:DescribeOrganizationConfiguration inspector2:Disable inspector2:DisableDelegatedAdminAccount inspector2:DisassociateMember inspector2:Enable inspector2:EnableDelegatedAdminAccount

Prefixo do serviço	Ações
	inspector2:GetCisScanReport inspector2:GetCisScanResultDetails inspector2:GetConfiguration inspector2:GetDelegatedAdminAccount inspector2:GetEc2DeepInspectionConfiguration inspector2:GetEncryptionKey inspector2:GetFindingsReportStatus inspector2:GetMember inspector2:GetSbomExport inspector2:ListAccountPermissions inspector2:ListCisScanConfigurations inspector2:ListCisScanResultsAggregatedByChecks inspector2:ListCisScanResultsAggregatedByTargetResource inspector2:ListCisScans inspector2:ListCoverage inspector2:ListCoverageStatistics inspector2:ListDelegatedAdminAccounts inspector2:ListFilters inspector2:ListFindingAggregations inspector2:ListFindings inspector2:ListMembers

Prefixo do serviço	Ações
	inspector2:ListUsageTotals inspector2:ResetEncryptionKey inspector2:SearchVulnerabilities inspector2:SendCisSessionHealth inspector2:SendCisSessionTelemetry inspector2:StartCisSession inspector2:StopCisSession inspector2:UpdateCisScanConfiguration inspector2:UpdateConfiguration inspector2:UpdateEc2DeepInspectionConfiguration inspector2:UpdateEncryptionKey inspector2:UpdateFilter inspector2:UpdateOrganizationConfiguration inspector2:UpdateOrgEc2DeepInspectionConfiguration

Prefixo do serviço	Ações
iot	iot:AcceptCertificateTransfer
	iot:AddThingToBillingGroup
	iot:AddThingToThingGroup
	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
	iot:CreateCertificateProvider

Prefixo do serviço	Ações
	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
	iot:CreateSecurityProfile
	iot:CreateStream

Prefixo do serviço	Ações
	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:DeleteCustomMetric
	iot:DeleteDimension
	iot:DeleteDomainConfiguration
	iot:DeleteDynamicThingGroup
	iot:DeleteFleetMetric
	iot:DeleteJob
	iot:DeleteJobExecution
	iot:DeleteJobTemplate
	iot:DeleteMitigationAction

Prefixo do serviço	Ações
	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
	iot:DeprecateThingType
	iot:DescribeAccountAuditConfiguration
	iot:DescribeAuditFinding

Prefixo do serviço	Ações
	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:DescribeEndpoint
	iot:DescribeEventConfigurations
	iot:DescribeFleetMetric
	iot:DescribeIndex
	iot:DescribeJob
	iot:DescribeJobExecution
	iot:DescribeJobTemplate
	iot:DescribeManagedJobTemplate

Prefixo do serviço	Ações
	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:EnableTopicRule
	iot:GetBehaviorModelTrainingSummaries
	iot:GetBucketsAggregation
	iot:GetCardinality
	iot:GetEffectivePolicies

Prefixo do serviço	Ações
	iot:GetJobDocument
	iot:GetLoggingOptions
	iot:GetOTAUpdate
	iot:GetPackage
	iot:GetPackageConfiguration
	iot:GetPackageVersion
	iot:GetPercentiles
	iot:GetPolicy
	iot:GetPolicyVersion
	iot:GetRegistrationCode
	iot:GetStatistics
	iot:GetTopicRule
	iot:GetTopicRuleDestination
	iot:GetV2LoggingOptions
	iot:ListActiveViolations
	iot:ListAttachedPolicies
	iot:ListAuditFindings
	iot:ListAuditMitigationActionsExecutions
	iot:ListAuditMitigationActionsTasks
	iot:ListAuditSuppressions
	iot:ListAuditTasks

Prefixo do serviço	Ações
	iot:ListAuthorizers
	iot:ListBillingGroups
	iot:ListCACertificates
	iot:ListCertificateProviders
	iot:ListCertificates
	iot:ListCertificatesByCA
	iot:ListCustomMetrics
	iot:ListDetectMitigationActionsExecutions
	iot:ListDetectMitigationActionsTasks
	iot:ListDimensions
	iot:ListDomainConfigurations
	iot:ListFleetMetrics
	iot:ListIndices
	iot:ListJobExecutionsForJob
	iot:ListJobExecutionsForThing
	iot:ListJobs
	iot:ListJobTemplates
	iot:ListManagedJobTemplates
	iot:ListMetricValues
	iot:ListMitigationActions
	iot:ListOTAUpdates

Prefixo do serviço	Ações
	iot:ListOutgoingCertificates
	iot:ListPackages
	iot:ListPackageVersions
	iot:ListPolicies
	iot:ListPolicyPrincipals
	iot:ListPolicyVersions
	iot:ListPrincipalPolicies
	iot:ListPrincipalThings
	iot:ListProvisioningTemplates
	iot:ListProvisioningTemplateVersions
	iot:ListRelatedResourcesForAuditFinding
	iot:ListRoleAliases
	iot:ListScheduledAudits
	iot:ListSecurityProfiles
	iot:ListSecurityProfilesForTarget
	iot:ListStreams
	iot:ListTargetsForPolicy
	iot:ListTargetsForSecurityProfile
	iot:ListThingGroups
	iot:ListThingGroupsForThing
	iot:ListThingPrincipals

Prefixo do serviço	Ações
	iot:ListThingRegistrationTaskReports
	iot:ListThingRegistrationTasks
	iot:ListThings
	iot:ListThingsInBillingGroup
	iot:ListThingsInThingGroup
	iot:ListThingTypes
	iot:ListTopicRuleDestinations
	iot:ListTopicRules
	iot:ListV2LoggingLevels
	iot:ListViolationEvents
	iot:PutVerificationStateOnViolation
	iot:RegisterCACertificate
	iot:RegisterCertificate
	iot:RegisterCertificateWithoutCA
	iot:RegisterThing
	iot:RejectCertificateTransfer
	iot:RemoveThingFromBillingGroup
	iot:RemoveThingFromThingGroup
	iot:ReplaceTopicRule
	iot:SearchIndex
	iot:SetDefaultAuthorizer

Prefixo do serviço	Ações
	iot:SetDefaultPolicyVersion
	iot:SetLoggingOptions
	iot:SetV2LoggingLevel
	iot:SetV2LoggingOptions
	iot:StartAuditMitigationActionsTask
	iot:StartDetectMitigationActionsTask
	iot:StartOnDemandAuditTask
	iot:StartThingRegistrationTask
	iot:StopThingRegistrationTask
	iot:TestAuthorization
	iot:TestInvokeAuthorizer
	iot:TransferCertificate
	iot:UpdateAccountAuditConfiguration
	iot:UpdateAuditSuppression
	iot:UpdateAuthorizer
	iot:UpdateBillingGroup
	iot:UpdateCACertificate
	iot:UpdateCertificate
	iot:UpdateCertificateProvider
	iot:UpdateCustomMetric
	iot:UpdateDimension

Prefixo do serviço	Ações
	iot:UpdateDomainConfiguration
	iot:UpdateDynamicThingGroup
	iot:UpdateEventConfigurations
	iot:UpdateFleetMetric
	iot:UpdateIndexingConfiguration
	iot:UpdateJob
	iot:UpdateMitigationAction
	iot:UpdatePackage
	iot:UpdatePackageConfiguration
	iot:UpdatePackageVersion
	iot:UpdateProvisioningTemplate
	iot:UpdateRoleAlias
	iot:UpdateScheduledAudit
	iot:UpdateSecurityProfile
	iot:UpdateStream
	iot:UpdateThing
	iot:UpdateThingGroup
	iot:UpdateThingGroupsForThing
	iot:UpdateTopicRuleDestination
	iot:ValidateSecurityProfileBehaviors

Prefixo do serviço	Ações
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
	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
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
	iotevents:ListAlarmModels iotevents:ListAlarmModelVersions iotevents:ListAlarms iotevents:ListDetectorModels iotevents:ListDetectorModelVersions 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
iotsitewise	iotsitewise:AssociateAssets iotsitewise:AssociateTimeSeriesToAssetProperty iotsitewise:BatchAssociateProjectAssets iotsitewise:BatchDisassociateProjectAssets iotsitewise:BatchGetAssetPropertyValue iotsitewise:BatchGetAssetPropertyValueHistory iotsitewise:BatchPutAssetPropertyValue iotsitewise:CreateAccessPolicy iotsitewise:CreateAsset iotsitewise:CreateAssetModel iotsitewise:CreateAssetModelCompositeModel iotsitewise:CreateBulkImportJob iotsitewise:CreateDashboard iotsitewise:CreateGateway iotsitewise:CreatePortal iotsitewise:CreateProject iotsitewise:DeleteAccessPolicy iotsitewise:DeleteAsset iotsitewise:DeleteAssetModel iotsitewise:DeleteAssetModelCompositeModel iotsitewise:DeleteDashboard

Prefixo do serviço	Ações
	iotsitewise:DeleteGateway iotsitewise:DeletePortal iotsitewise:DeleteProject iotsitewise:DeleteTimeSeries iotsitewise:DescribeAccessPolicy iotsitewise:DescribeAsset iotsitewise:DescribeAssetCompositeModel iotsitewise:DescribeAssetModel iotsitewise:DescribeAssetModelCompositeModel iotsitewise:DescribeAssetProperty iotsitewise:DescribeBulkImportJob iotsitewise:DescribeDashboard iotsitewise:DescribeDefaultEncryptionConfiguration iotsitewise:DescribeGateway iotsitewise:DescribeGatewayCapabilityConfiguration iotsitewise:DescribeLoggingOptions iotsitewise:DescribePortal iotsitewise:DescribeProject iotsitewise:DescribeStorageConfiguration iotsitewise:DescribeTimeSeries iotsitewise:DisassociateAssets

Prefixo do serviço	Ações
	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:ListDashboards iotsitewise:ListGateways iotsitewise:ListPortals iotsitewise:ListProjectAssets iotsitewise:ListProjects iotsitewise:ListTimeSeries iotsitewise:PutDefaultEncryptionConfiguration

Prefixo do serviço	Ações
	iotsitewise:PutLoggingOptions iotsitewise:PutStorageConfiguration iotsitewise:UpdateAccessPolicy iotsitewise:UpdateAsset iotsitewise:UpdateAssetModel iotsitewise:UpdateAssetModelCompositeModel iotsitewise:UpdateAssetProperty iotsitewise:UpdateDashboard iotsitewise:UpdateGateway iotsitewise:UpdateGatewayCapabilityConfiguration iotsitewise:UpdatePortal iotsitewise:UpdateProject

Prefixo do serviço	Ações
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:ListComponents
	iottwinmaker:ListComponentTypes
	iottwinmaker:ListEntities
	iottwinmaker:ListMetadataTransferJobs

Prefixo do serviço	Ações
	iottwinmaker:ListProperties
	iottwinmaker:ListScenes
	iottwinmaker:ListSyncJobs
	iottwinmaker:ListSyncResources
	iottwinmaker:ListWorkspaces
	iottwinmaker:UpdateComponentType
	iottwinmaker:UpdateEntity
	iottwinmaker:UpdatePricingPlan
	iottwinmaker:UpdateScene
	iottwinmaker:UpdateWorkspace

Prefixo do serviço	Ações
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
	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
	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
	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:ListWirelessGateways iotwireless:ListWirelessGatewayTaskDefinitions iotwireless:PutPositionConfiguration iotwireless:PutResourceLogLevel

Prefixo do serviço	Ações
	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
	iotwireless:UpdatePosition iotwireless:UpdateResourceEventConfiguration iotwireless:UpdateResourcePosition iotwireless:UpdateWirelessDevice iotwireless:UpdateWirelessDeviceImportTask iotwireless:UpdateWirelessGateway

Prefixo do serviço	Ações
ivs	ivs:BatchGetChannel
	ivs:BatchGetStreamKey
	ivs:BatchStartViewerSessionRevocation
	ivs:CreateChannel
	ivs:CreateEncoderConfiguration
	ivs:CreateParticipantToken
	ivs:CreatePlaybackRestrictionPolicy
	ivs:CreateRecordingConfiguration
	ivs:CreateStorageConfiguration
	ivs:CreateStreamKey
	ivs:DeleteChannel
	ivs:DeleteEncoderConfiguration
	ivs:DeletePlaybackKeyPair
	ivs:DeletePlaybackRestrictionPolicy
	ivs:DeleteRecordingConfiguration
	ivs:DeleteStorageConfiguration
	ivs:DeleteStreamKey
	ivs:DisconnectParticipant
	ivs:GetChannel
	ivs:GetComposition
	ivs:GetEncoderConfiguration

Prefixo do serviço	Ações
	ivs:GetParticipant
	ivs:GetPlaybackKeyPair
	ivs:GetPlaybackRestrictionPolicy
	ivs:GetRecordingConfiguration
	ivs:GetStorageConfiguration
	ivs:GetStream
	ivs:GetStreamKey
	ivs:GetStreamSession
	ivs:ImportPlaybackKeyPair
	ivs:ListChannels
	ivs:ListCompositions
	ivs:ListEncoderConfigurations
	ivs:ListParticipantEvents
	ivs:ListParticipants
	ivs:ListPlaybackKeyPairs
	ivs:ListPlaybackRestrictionPolicies
	ivs:ListRecordingConfigurations
	ivs:ListStorageConfigurations
	ivs:ListStreamKeys
	ivs:ListStreams
	ivs:ListStreamSessions

Prefixo do serviço	Ações
	ivs:PutMetadata ivs:StartComposition ivs:StartViewerSessionRevocation ivs:StopComposition ivs:StopStream ivs:UpdateChannel 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
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
	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
	kafka:UpdateConfiguration kafka:UpdateConnectivity kafka:UpdateMonitoring 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:ListConnectors kafkaconnect:ListCustomPlugins kafkaconnect:ListWorkerConfigurations kafkaconnect:UpdateConnector

Prefixo do serviço	Ações
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
	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:ListDataSources
	kendra:ListDataSourceSyncJobs
	kendra:ListEntityPersonas
	kendra:ListExperienceEntities
	kendra:ListExperiences

Prefixo do serviço	Ações
	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
kinesis	kinesis:CreateStream kinesis:DecreaseStreamRetentionPeriod kinesis:DeleteStream kinesis:DeregisterStreamConsumer 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:UpdateShardCount kinesis:UpdateStreamMode

Prefixo do serviço	Ações
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:DescribeApplicationSnapshot kinesisanalytics:DescribeApplicationVersion kinesisanalytics:DiscoverInputSchema kinesisanalytics:ListApplications

Prefixo do serviço	Ações
	kinesisanalytics:ListApplicationSnapshots kinesisanalytics:ListApplicationVersions kinesisanalytics:RollbackApplication kinesisanalytics:StartApplication kinesisanalytics:StopApplication kinesisanalytics:UpdateApplication kinesisanalytics:UpdateApplicationMaintenanceConfiguration

Prefixo do serviço	Ações
kms	kms:CancelKeyDeletion kms:ConnectCustomKeyStore kms:CreateAlias kms:CreateCustomKeyStore kms:CreateGrant kms:CreateKey kms:Decrypt kms:DeleteAlias kms:DeleteCustomKeyStore kms:DeleteImportedKeyMaterial kms:DescribeCustomKeyStores kms:DescribeKey kms:DisableKey kms:DisableKeyRotation kms:DisconnectCustomKeyStore kms:EnableKey kms:EnableKeyRotation kms:Encrypt kms:GenerateDataKey kms:GenerateDataKeyPair kms:GenerateDataKeyPairWithoutPlaintext

Prefixo do serviço	Ações
	kms:GenerateDataKeyWithoutPlaintext kms:GenerateMac kms:GenerateRandom kms:GetKeyPolicy kms:GetKeyRotationStatus kms:GetParametersForImport kms:GetPublicKey kms:ImportKeyMaterial kms:ListAliases kms:ListGrants kms:ListKeyPolicies kms:ListKeys kms:ListRetirableGrants kms:ReplicateKey kms:RetireGrant kms:RevokeGrant kms:ScheduleKeyDeletion kms:Sign kms:UpdateAlias kms:UpdateCustomKeyStore kms:UpdateKeyDescription

Prefixo do serviço	Ações
	kms:UpdatePrimaryRegion kms:Verify kms:VerifyMac

Prefixo do serviço	Ações
lambda	lambda:AddLayerVersionPermission
	lambda:AddLayerVersionPermission
	lambda:AddPermission
	lambda:AddPermission
	lambda:AddPermission
	lambda:CreateAlias
	lambda:CreateAlias
	lambda:CreateCodeSigningConfig
	lambda:CreateEventSourceMapping
	lambda:CreateEventSourceMapping
	lambda:CreateFunction
	lambda:CreateFunction
	lambda:CreateFunctionUrlConfig
	lambda:DeleteAlias
	lambda:DeleteAlias
	lambda:DeleteCodeSigningConfig
	lambda:DeleteEventSourceMapping
	lambda:DeleteEventSourceMapping
	lambda:DeleteFunction
	lambda:DeleteFunction
	lambda:DeleteFunctionCodeSigningConfig

Prefixo do serviço	Ações
	lambda:DeleteFunctionConcurrency
	lambda:DeleteFunctionConcurrency
	lambda:DeleteFunctionEventInvokeConfig
	lambda:DeleteFunctionUrlConfig
	lambda:DeleteLayerVersion
	lambda:DeleteLayerVersion
	lambda:DeleteProvisionedConcurrencyConfig
	lambda:GetAccountSettings
	lambda:GetAccountSettings
	lambda:GetAlias
	lambda:GetAlias
	lambda:GetCodeSigningConfig
	lambda:GetEventSourceMapping
	lambda:GetEventSourceMapping
	lambda:GetFunction
	lambda:GetFunction
	lambda:GetFunction
	lambda:GetFunctionCodeSigningConfig
	lambda:GetFunctionConcurrency
	lambda:GetFunctionConfiguration
	lambda:GetFunctionConfiguration

Prefixo do serviço	Ações
	lambda:GetFunctionConfiguration
	lambda:GetFunctionEventInvokeConfig
	lambda:GetFunctionUrlConfig
	lambda:GetLayerVersion
	lambda:GetLayerVersion
	lambda:GetLayerVersion
	lambda:GetLayerVersion
	lambda:GetLayerVersionPolicy
	lambda:GetLayerVersionPolicy
	lambda:GetPolicy
	lambda:GetPolicy
	lambda:GetPolicy
	lambda:GetProvisionedConcurrencyConfig
	lambda:GetRuntimeManagementConfig
	lambda:ListAliases
	lambda:ListAliases
	lambda:ListCodeSigningConfigs
	lambda:ListEventSourceMappings
	lambda:ListEventSourceMappings
	lambda:ListFunctionEventInvokeConfigs
	lambda:ListFunctions

Prefixo do serviço	Ações
	lambda:ListFunctions
	lambda:ListFunctionsByCodeSigningConfig
	lambda:ListFunctionUrlConfigs
	lambda:ListLayers
	lambda:ListLayers
	lambda:ListLayerVersions
	lambda:ListLayerVersions
	lambda:ListProvisionedConcurrencyConfigs
	lambda:ListVersionsByFunction
	lambda:ListVersionsByFunction
	lambda:PublishLayerVersion
	lambda:PublishLayerVersion
	lambda:PublishVersion
	lambda:PublishVersion
	lambda:PutFunctionCodeSigningConfig
	lambda:PutFunctionConcurrency
	lambda:PutFunctionConcurrency
	lambda:PutFunctionEventInvokeConfig
	lambda:PutProvisionedConcurrencyConfig
	lambda:PutRuntimeManagementConfig
	lambda:RemoveLayerVersionPermission

Prefixo do serviço	Ações
	lambda:RemoveLayerVersionPermission
	lambda:RemovePermission
	lambda:RemovePermission
	lambda:RemovePermission
	lambda:UpdateAlias
	lambda:UpdateAlias
	lambda:UpdateCodeSigningConfig
	lambda:UpdateEventSourceMapping
	lambda:UpdateEventSourceMapping
	lambda:UpdateFunctionCode
	lambda:UpdateFunctionCode
	lambda:UpdateFunctionCode
	lambda:UpdateFunctionConfiguration
	lambda:UpdateFunctionConfiguration
	lambda:UpdateFunctionConfiguration
	lambda:UpdateFunctionEventInvokeConfig
	lambda:UpdateFunctionUrlConfig

Prefixo do serviço	Ações
lex	lex:BatchCreateCustomVocabularyItem lex:BatchDeleteCustomVocabularyItem lex:BatchUpdateCustomVocabularyItem lex:BuildBotLocale lex:CreateBotAlias lex:CreateBotVersion lex:CreateExport lex:CreateIntentVersion lex:CreateResourcePolicy lex:CreateSlotTypeVersion lex:CreateTestSetDiscrepancyReport lex:CreateUploadUrl lex:DeleteBot lex:DeleteBotChannelAssociation lex:DeleteExport lex:DeleteImport lex:DeleteIntentVersion lex:DeleteResourcePolicy lex:DeleteSlotTypeVersion lex:DeleteTestSet lex:DeleteUtterances

Prefixo do serviço	Ações
	lex:DescribeBotAlias lex:DescribeBotRecommendation 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 lex:GetBotChannelAssociation lex:GetBotChannelAssociations lex:GetBots lex:GetBotVersions lex:GetBuiltinIntent

Prefixo do serviço	Ações
	lex:GetBuiltinIntents lex:GetBuiltinSlotTypes lex:GetExport lex:GetImport lex:GetIntent lex:GetIntents lex:GetIntentVersions lex:GetMigration lex:GetMigrations lex:GetSlotType lex:GetSlotTypes lex:GetSlotTypeVersions lex:GetTestExecutionArtifactsUrl lex:GetUtterancesView lex:ListBotAliases lex:ListBotRecommendations lex:ListBotResourceGenerations lex:ListBots lex:ListBotVersions lex:ListBuiltInIntents lex:ListBuiltInSlotTypes

Prefixo do serviço	Ações
	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 lex:PutIntent lex:PutSlotType lex:SearchAssociatedTranscripts lex:StartBotRecommendation lex:StartImport lex:StartMigration lex:StartTestExecution lex:StartTestSetGeneration

Prefixo do serviço	Ações
	lex:StopBotRecommendation lex:UpdateBotAlias lex:UpdateBotRecommendation lex:UpdateExport lex:UpdateResourcePolicy
license-manager-linux-subscriptions	license-manager-linux-subscriptions:GetServiceSettings license-manager-linux-subscriptions:ListLinuxSubscriptionInstances license-manager-linux-subscriptions:ListLinuxSubscriptions license-manager-linux-subscriptions:UpdateServiceSettings

Prefixo do serviço	Ações
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
	lightsail:CreateGUISessionAccessDetails lightsail:CreateInstances lightsail:CreateInstancesFromSnapshot lightsail:CreateInstanceSnapshot 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
	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
	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:GetDisks lightsail:GetDiskSnapshot lightsail:GetDiskSnapshots

Prefixo do serviço	Ações
	lightsail:GetDistributionBundles lightsail:GetDistributionLatestCacheReset lightsail:GetDistributionMetricData lightsail:GetDistributions lightsail:GetDomain lightsail:GetExportSnapshotRecords lightsail:GetInstance lightsail:GetInstanceMetricData lightsail:GetInstancePortStates lightsail:GetInstances lightsail:GetInstanceSnapshot lightsail:GetInstanceSnapshots lightsail:GetInstanceState lightsail:GetKeyPair lightsail:GetKeyPairs lightsail:GetLoadBalancer lightsail:GetLoadBalancerMetricData lightsail:GetLoadBalancers lightsail:GetLoadBalancerTlsCertificates lightsail:GetLoadBalancerTlsPolicies lightsail:GetOperation

Prefixo do serviço	Ações
	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:GetRelationalDatabases lightsail:GetRelationalDatabaseSnapshot lightsail:GetRelationalDatabaseSnapshots lightsail:GetSetupHistory lightsail:GetStaticIp lightsail:GetStaticIps lightsail:ImportKeyPair lightsail:IsVpcPeered lightsail:OpenInstancePublicPorts

Prefixo do serviço	Ações
	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
	lightsail:UpdateBucketBundle lightsail:UpdateContainerService lightsail:UpdateDistribution lightsail:UpdateDistributionBundle lightsail:UpdateDomainEntry lightsail:UpdateInstanceMetadataOptions lightsail:UpdateLoadBalancerAttribute lightsail:UpdateRelationalDatabase lightsail:UpdateRelationalDatabaseParameters

Prefixo do serviço	Ações
logs	logs:AssociateKmsKey
	logs:CancelExportTask
	logs:CreateExportTask
	logs:CreateLogAnomalyDetector
	logs:CreateLogGroup
	logs:CreateLogStream
	logs:DeleteDataProtectionPolicy
	logs:DeleteDelivery
	logs:DeleteDeliveryDestination
	logs:DeleteDeliveryDestinationPolicy
	logs:DeleteDeliverySource
	logs:DeleteDestination
	logs:DeleteLogGroup
	logs:DeleteLogStream
	logs:DeleteMetricFilter
	logs:DeleteQueryDefinition
	logs:DeleteResourcePolicy
	logs:DeleteRetentionPolicy
	logs:DeleteSubscriptionFilter
	logs:DescribeAccountPolicies
	logs:DescribeDeliveries

Prefixo do serviço	Ações
	logs:DescribeDeliveryDestinations
	logs:DescribeDeliverySources
	logs:DescribeDestinations
	logs:DescribeExportTasks
	logs:DescribeLogGroups
	logs:DescribeLogStreams
	logs:DescribeMetricFilters
	logs:DescribeQueries
	logs:DescribeQueryDefinitions
	logs:DescribeResourcePolicies
	logs:DescribeSubscriptionFilters
	logs:DisassociateKmsKey
	logs:GetDataProtectionPolicy
	logs:GetDelivery
	logs:GetDeliveryDestination
	logs:GetDeliveryDestinationPolicy
	logs:GetDeliverySource
	logs:GetLogGroupFields
	logs:GetLogRecord
	logs:GetQueryResults
	logs:ListAnomalies

Prefixo do serviço	Ações
	logs:ListLogAnomalyDetectors
	logs:PutDataProtectionPolicy
	logs:PutDeliveryDestination
	logs:PutDeliveryDestinationPolicy
	logs:PutDeliverySource
	logs:PutDestination
	logs:PutDestinationPolicy
	logs:PutMetricFilter
	logs:putQueryDefinition
	logs:PutResourcePolicy
	logs:PutRetentionPolicy
	logs:PutSubscriptionFilter
	logs:StartLiveTail
	logs:StartQuery
	logs:StopQuery
	logs:TestMetricFilter

Prefixo do serviço	Ações
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:Describelabel lookoutequipment:DescribeLabelGroup lookoutequipment:DescribeModel lookoutequipment:DescribeModelVersion lookoutequipment:DescribeResourcePolicy lookoutequipment:DescribeRetrainingScheduler

Prefixo do serviço	Ações
	lookoutequipment:ImportDataset lookoutequipment:ImportModelVersion lookoutequipment:ListDataIngestionJobs lookoutequipment:ListDatasets lookoutequipment:ListInferenceEvents lookoutequipment:ListInferenceExecutions lookoutequipment:ListInferenceSchedulers lookoutequipment:ListLabelGroups lookoutequipment:ListLabels lookoutequipment:ListModels lookoutequipment:ListModelVersions lookoutequipment:ListRetrainingSchedulers lookoutequipment:ListSensorStatistics lookoutequipment:PutResourcePolicy lookoutequipment:StartDataIngestionJob lookoutequipment:StartInferenceScheduler lookoutequipment:StartRetrainingScheduler lookoutequipment:StopInferenceScheduler lookoutequipment:StopRetrainingScheduler lookoutequipment:UpdateActiveModelVersion lookoutequipment:UpdateInferenceScheduler

Prefixo do serviço	Ações
	lookoutequipment:UpdateLabelGroup lookoutequipment:UpdateModel lookoutequipment:UpdateRetrainingScheduler

Prefixo do serviço	Ações
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
	lookoutmetrics:ListAnomalyGroupTimeSeries lookoutmetrics:ListMetricSets lookoutmetrics:PutFeedback lookoutmetrics:UpdateAlert lookoutmetrics:UpdateAnomalyDetector lookoutmetrics:UpdateMetricSet

Prefixo do serviço	Ações
lookoutvision	lookoutvision:CreateDataset lookoutvision:CreateModel lookoutvision:CreateProject lookoutvision:DeleteDataset lookoutvision:DeleteModel lookoutvision:DeleteProject lookoutvision:DescribeDataset lookoutvision:DescribeModel lookoutvision:DescribeModelPackagingJob lookoutvision:DescribeProject lookoutvision:DetectAnomalies lookoutvision:ListDatasetEntries lookoutvision:ListModelPackagingJobs lookoutvision:ListModels lookoutvision:ListProjects lookoutvision:StartModel lookoutvision:StartModelPackagingJob lookoutvision:StopModel lookoutvision:UpdateDatasetEntries

Prefixo do serviço	Ações
m2	m2:CancelBatchJobExecution
	m2:CreateApplication
	m2:CreateDataSetImportTask
	m2:CreateDeployment
	m2:CreateEnvironment
	m2:DeleteApplication
	m2:DeleteApplicationFromEnvironment
	m2:DeleteEnvironment
	m2:GetApplication
	m2:GetApplicationVersion
	m2:GetBatchJobExecution
	m2:GetDataSetDetails
	m2:GetDataSetImportTask
	m2:GetDeployment
	m2:GetEnvironment
	m2:GetSignedBluinsightsUrl
	m2:ListApplications
	m2:ListApplicationVersions
	m2:ListBatchJobDefinitions
	m2:ListBatchJobExecutions
	m2:ListDataSetImportHistory

Prefixo do serviço	Ações
	m2:ListDataSets m2:ListDeployments m2:ListEngineVersions m2:ListEnvironments m2:StartApplication m2:StartBatchJob m2:StopApplication m2:UpdateApplication m2:UpdateEnvironment

Prefixo do serviço	Ações
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:ListProposals

Prefixo do serviço	Ações
	managedblockchain:ListProposalVotes managedblockchain:RejectInvitation managedblockchain:UpdateMember managedblockchain:UpdateNode managedblockchain:VoteOnProposal

Prefixo do serviço	Ações
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:DeregisterGatewayInstance
	mediaconnect:DescribeBridge
	mediaconnect:DescribeFlow
	mediaconnect:DescribeFlowSourceMetadata
	mediaconnect:DescribeGateway
	mediaconnect:DescribeGatewayInstance
	mediaconnect:DescribeOffering
	mediaconnect:DescribeReservation
	mediaconnect:GrantFlowEntitlements

Prefixo do serviço	Ações
	mediaconnect:ListBridges
	mediaconnect:ListEntitlements
	mediaconnect:ListFlows
	mediaconnect:ListGatewayInstances
	mediaconnect:ListGateways
	mediaconnect:ListOfferings
	mediaconnect:ListReservations
	mediaconnect:PurchaseOffering
	mediaconnect:RemoveBridgeOutput
	mediaconnect:RemoveBridgeSource
	mediaconnect:RemoveFlowMediaStream
	mediaconnect:RemoveFlowOutput
	mediaconnect:RemoveFlowSource
	mediaconnect:RemoveFlowVpcInterface
	mediaconnect:RevokeFlowEntitlement
	mediaconnect:StartFlow
	mediaconnect:StopFlow
	mediaconnect:UpdateBridge
	mediaconnect:UpdateBridgeOutput
	mediaconnect:UpdateBridgeSource
	mediaconnect:UpdateBridgeState

Prefixo do serviço	Ações
	mediaconnect:UpdateFlow mediaconnect:UpdateFlowEntitlement mediaconnect:UpdateFlowMediaStream mediaconnect:UpdateFlowOutput mediaconnect:UpdateFlowSource mediaconnect:UpdateGatewayInstance

Prefixo do serviço	Ações
mediaconvert	mediaconvert:AssociateCertificate mediaconvert:CancelJob mediaconvert:CreateJob mediaconvert:CreateJobTemplate mediaconvert:CreatePreset mediaconvert:CreateQueue mediaconvert:DeleteJobTemplate mediaconvert:DeletePolicy mediaconvert:DeletePreset mediaconvert:DeleteQueue mediaconvert:DescribeEndpoints mediaconvert:DisassociateCertificate mediaconvert:GetJob mediaconvert:GetJobTemplate mediaconvert:GetPolicy mediaconvert:GetPreset mediaconvert:GetQueue mediaconvert:ListJobs mediaconvert:ListJobTemplates mediaconvert:ListPresets mediaconvert:ListQueues

Prefixo do serviço	Ações
	mediaconvert:PutPolicy mediaconvert:UpdateJobTemplate mediaconvert:UpdatePreset mediaconvert:UpdateQueue

Prefixo do serviço	Ações
medialive	medialive:AcceptInputDeviceTransfer
	medialive:BatchDelete
	medialive:BatchStart
	medialive:BatchStop
	medialive:BatchUpdateSchedule
	medialive:CancelInputDeviceTransfer
	medialive:ClaimDevice
	medialive:CreateChannel
	medialive:CreateCloudWatchAlarmTemplate
	medialive:CreateCloudWatchAlarmTemplateGroup
	medialive:CreateEventBridgeRuleTemplate
	medialive:CreateEventBridgeRuleTemplateGroup
	medialive:CreateInput
	medialive:CreateInputSecurityGroup
	medialive:CreateMultiplex
	medialive:CreateMultiplexProgram
	medialive:CreatePartnerInput
	medialive:CreateSignalMap
	medialive:DeleteChannel
	medialive:DeleteCloudWatchAlarmTemplate
	medialive:DeleteCloudWatchAlarmTemplateGroup

Prefixo do serviço	Ações
	medialive:DeleteEventBridgeRuleTemplate
	medialive:DeleteEventBridgeRuleTemplateGroup
	medialive:DeleteInput
	medialive:DeleteInputSecurityGroup
	medialive:DeleteMultiplex
	medialive:DeleteMultiplexProgram
	medialive:DeleteReservation
	medialive:DeleteSchedule
	medialive:DeleteSignalMap
	medialive:DescribeAccountConfiguration
	medialive:DescribeChannel
	medialive:DescribeInput
	medialive:DescribeInputDevice
	medialive:DescribeInputDeviceThumbnail
	medialive:DescribeInputSecurityGroup
	medialive:DescribeMultiplex
	medialive:DescribeMultiplexProgram
	medialive:DescribeOffering
	medialive:DescribeReservation
	medialive:DescribeSchedule
	medialive:DescribeThumbnails

Prefixo do serviço	Ações
	medialive:GetCloudWatchAlarmTemplate
	medialive:GetCloudWatchAlarmTemplateGroup
	medialive:GetEventBridgeRuleTemplate
	medialive:GetEventBridgeRuleTemplateGroup
	medialive:GetSignalMap
	medialive:ListChannels
	medialive:ListCloudWatchAlarmTemplateGroups
	medialive:ListCloudWatchAlarmTemplates
	medialive:ListEventBridgeRuleTemplateGroups
	medialive:ListEventBridgeRuleTemplates
	medialive:ListInputDevices
	medialive:ListInputDeviceTransfers
	medialive:ListInputs
	medialive:ListInputSecurityGroups
	medialive:ListMultiplexes
	medialive:ListMultiplexPrograms
	medialive:ListOfferings
	medialive:ListReservations
	medialive:ListSignalMaps
	medialive:PurchaseOffering
	medialive:RebootInputDevice

Prefixo do serviço	Ações
	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:UpdateCloudWatchAlarmTemplate
	medialive:UpdateCloudWatchAlarmTemplateGroup
	medialive:UpdateEventBridgeRuleTemplate
	medialive:UpdateEventBridgeRuleTemplateGroup
	medialive:UpdateInput

Prefixo do serviço	Ações
	medialive:UpdateInputDevice medialive:UpdateInputSecurityGroup medialive:UpdateMultiplex medialive:UpdateMultiplexProgram medialive:UpdateReservation

Prefixo do serviço	Ações
mediastore	mediastore:CreateContainer mediastore:DeleteContainer mediastore:DeleteContainerPolicy mediastore:DeleteCorsPolicy mediastore:DeleteLifecyclePolicy mediastore:DeleteMetricPolicy mediastore:DescribeContainer mediastore:GetContainerPolicy mediastore:GetCorsPolicy mediastore:GetLifecyclePolicy mediastore:GetMetricPolicy mediastore:ListContainers mediastore:PutContainerPolicy mediastore:PutCorsPolicy mediastore:PutLifecyclePolicy mediastore:PutMetricPolicy mediastore:StartAccessLogging mediastore:StopAccessLogging

Prefixo do serviço	Ações
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
	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
memorydb	memorydb:BatchUpdateCluster memorydb:CopySnapshot memorydb:CreateAcl memorydb:CreateCluster memorydb:CreateParameterGroup memorydb:CreateSnapshot memorydb:CreateSubnetGroup memorydb:CreateUser memorydb:DeleteAcl memorydb:DeleteCluster memorydb:DeleteParameterGroup memorydb:DeleteSnapshot memorydb:DeleteSubnetGroup memorydb:DeleteUser memorydb:DescribeAcls memorydb:DescribeClusters memorydb:DescribeEngineVersions memorydb:DescribeEvents memorydb:DescribeParameterGroups memorydb:DescribeParameters memorydb:DescribeReservedNodes

Prefixo do serviço	Ações
	memorydb:DescribeReservedNodesOfferings memorydb:DescribeServiceUpdates memorydb:DescribeSnapshots memorydb:DescribeSubnetGroups memorydb:DescribeUsers memorydb:FailoverShard memorydb:ListAllowedNodeTypeUpdates memorydb:PurchaseReservedNodesOffering memorydb:ResetParameterGroup memorydb:UpdateAcl memorydb:UpdateCluster memorydb:UpdateParameterGroup memorydb:UpdateSubnetGroup memorydb:UpdateUser

Prefixo do serviço	Ações
mgh	mgh:AssociateCreatedArtifact
	mgh:AssociateDiscoveredResource
	mgh:CreateHomeRegionControl
	mgh:CreateProgressUpdateStream
	mgh:DeleteHomeRegionControl
	mgh:DeleteProgressUpdateStream
	mgh:DescribeApplicationState
	mgh:DescribeHomeRegionControls
	mgh:DescribeMigrationTask
	mgh:DisassociateCreatedArtifact
	mgh:DisassociateDiscoveredResource
	mgh:GetHomeRegion
	mgh:ImportMigrationTask
	mgh:ListApplicationStates
	mgh:ListCreatedArtifacts
	mgh:ListDiscoveredResources
	mgh:ListMigrationTasks
	mgh:ListProgressUpdateStreams
	mgh:NotifyApplicationState
	mgh:NotifyMigrationTaskState
	mgh:PutResourceAttributes

Prefixo do serviço	Ações
mgn	mgn:ArchiveApplication mgn:ArchiveWave mgn:AssociateApplications mgn:AssociateSourceServers mgn:ChangeServerLifeCycleState mgn:CreateApplication mgn:CreateConnector mgn:CreateLaunchConfigurationTemplate mgn:CreateReplicationConfigurationTemplate mgn:CreateWave mgn:DeleteApplication mgn:DeleteConnector mgn:DeleteJob mgn:DeleteLaunchConfigurationTemplate mgn:DeleteReplicationConfigurationTemplate mgn:DeleteSourceServer mgn:DeleteVcenterClient mgn:DeleteWave mgn:DescribeJobLogItems mgn:DescribeJobs mgn:DescribeLaunchConfigurationTemplates

Prefixo do serviço	Ações
	mgn:DescribeReplicationConfigurationTemplates mgn:DescribeVcenterClients mgn:DisassociateApplications mgn:DisassociateSourceServers mgn:DisconnectFromService mgn:FinalizeCutover mgn:GetReplicationConfiguration mgn:InitializeService mgn:ListConnectors mgn:ListExportErrors mgn:ListExports mgn:ListImportErrors mgn:ListImports mgn:ListManagedAccounts mgn:ListSourceServerActions mgn:ListTemplateActions mgn:MarkAsArchived mgn:PauseReplication mgn:PutSourceServerAction mgn:PutTemplateAction mgn:RemoveSourceServerAction

Prefixo do serviço	Ações
	mgn:RemoveTemplateAction mgn:ResumeReplication mgn:RetryDataReplication mgn:StartCutover mgn:StartExport mgn:StartImport mgn:StartReplication mgn:StartTest mgn:StopReplication mgn:TerminateTargetInstances mgn:UnarchiveApplication mgn:UnarchiveWave mgn:UpdateApplication mgn:UpdateConnector mgn:UpdateLaunchConfigurationTemplate mgn:UpdateReplicationConfiguration mgn:UpdateReplicationConfigurationTemplate mgn:UpdateSourceServer mgn:UpdateSourceServerReplicationType mgn:UpdateWave

Prefixo do serviço	Ações
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:PutPortfolioPreferences
	migrationhub-strategy:RegisterCollector

Prefixo do serviço	Ações
	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
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
	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
	mobiletargeting:GetApplicationSettings mobiletargeting:GetApps mobiletargeting:GetBaiduChannel mobiletargeting:GetCampaign mobiletargeting:GetCampaignActivities mobiletargeting:GetCampaignDateRangeKpi mobiletargeting:GetCampaigns mobiletargeting:GetCampaignVersion mobiletargeting:GetCampaignVersions 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
	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:GetSegments mobiletargeting:GetSegmentVersion mobiletargeting:GetSegmentVersions mobiletargeting:GetSmsChannel mobiletargeting:GetSmsTemplate mobiletargeting:GetUserEndpoints mobiletargeting:GetVoiceChannel mobiletargeting:GetVoiceTemplate

Prefixo do serviço	Ações
	mobiletargeting:ListJourneys mobiletargeting:ListTemplates mobiletargeting:ListTemplateVersions 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
	mobiletargeting:UpdateJourneyState mobiletargeting:UpdatePushTemplate mobiletargeting:UpdateRecommenderConfiguration mobiletargeting:UpdateSegment mobiletargeting:UpdateSmsChannel mobiletargeting:UpdateSmsTemplate mobiletargeting:UpdateTemplateActiveVersion mobiletargeting:UpdateVoiceChannel mobiletargeting:UpdateVoiceTemplate mobiletargeting:VerifyOTPMessage

Prefixo do serviço	Ações
mq	mq:CreateBroker mq:CreateConfiguration mq:CreateUser mq:DeleteBroker mq:DeleteUser mq:DescribeBroker mq:DescribeBrokerEngineTypes mq:DescribeBrokerInstanceOptions mq:DescribeConfiguration mq:DescribeConfigurationRevision mq:DescribeUser mq:ListBrokers mq:ListConfigurationRevisions mq:ListConfigurations mq:ListUsers mq:Promote mq:RebootBroker mq:UpdateBroker mq:UpdateConfiguration mq:UpdateUser

Prefixo do serviço	Ações
networkmanager	networkmanager:AcceptAttachment networkmanager:AssociateConnectPeer networkmanager:AssociateCustomerGateway networkmanager:AssociateLink networkmanager:AssociateTransitGatewayConnectPeer networkmanager:CreateConnectAttachment networkmanager:CreateConnection networkmanager:CreateConnectPeer networkmanager:CreateCoreNetwork networkmanager:CreateDevice networkmanager:CreateGlobalNetwork networkmanager:CreateLink networkmanager:CreateSite networkmanager:CreateSiteToSiteVpnAttachment networkmanager:CreateTransitGatewayPeering networkmanager:CreateTransitGatewayRouteTableAttachment networkmanager:CreateVpcAttachment networkmanager:DeleteAttachment networkmanager:DeleteConnection networkmanager:DeleteConnectPeer networkmanager:DeleteCoreNetwork

Prefixo do serviço	Ações
	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:GetConnections networkmanager:GetConnectPeer networkmanager:GetConnectPeerAssociations networkmanager:GetCoreNetwork networkmanager:GetCoreNetworkChangeEvents networkmanager:GetCoreNetworkChangeSet

Prefixo do serviço	Ações
	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:GetSites networkmanager:GetSiteToSiteVpnAttachment networkmanager:GetTransitGatewayConnectPeerAssociations networkmanager:GetTransitGatewayPeering networkmanager:GetTransitGatewayRegistrations networkmanager:GetTransitGatewayRouteTableAttachment networkmanager:GetVpcAttachment networkmanager:ListAttachments networkmanager:ListConnectPeers

Prefixo do serviço	Ações
	networkmanager:ListCoreNetworkPolicyVersions networkmanager:ListCoreNetworks networkmanager:ListOrganizationServiceAccessStatus networkmanager:ListPeerings networkmanager:PutCoreNetworkPolicy networkmanager:PutResourcePolicy networkmanager:RegisterTransitGateway networkmanager:RejectAttachment networkmanager:RestoreCoreNetworkPolicyVersion networkmanager:StartOrganizationServiceAccessUpdate networkmanager:StartRouteAnalysis networkmanager:UpdateConnection networkmanager:UpdateCoreNetwork networkmanager:UpdateDevice networkmanager:UpdateGlobalNetwork networkmanager:UpdateLink networkmanager:UpdateNetworkResourceMetadata networkmanager:UpdateSite networkmanager:UpdateVpcAttachment

Prefixo do serviço	Ações
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
	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
omics	omics:AbortMultipartReadSetUpload omics:BatchDeleteReadSet omics:CancelAnnotationImportJob omics:CancelRun omics:CancelVariantImportJob omics:CompleteMultipartReadSetUpload omics:CreateAnnotationStore omics:CreateMultipartReadSetUpload omics:CreateReferenceStore omics:CreateRunGroup omics:CreateSequenceStore omics:CreateVariantStore omics:CreateWorkflow omics:DeleteAnnotationStore omics:DeleteReference omics:DeleteReferenceStore omics:DeleteRun omics:DeleteRunGroup omics:DeleteSequenceStore omics:DeleteVariantStore omics:DeleteWorkflow

Prefixo do serviço	Ações
	omics:GetAnnotationImportJob omics:GetAnnotationStore omics:GetReadSet omics:GetReadSetActivationJob omics:GetReadSetExportJob omics:GetReadSetImportJob omics:GetReadSetMetadata omics:GetReference omics:GetReferenceImportJob omics:GetReferenceMetadata omics:GetReferenceStore omics:GetRun omics:GetRunGroup omics:GetRunTask omics:GetSequenceStore omics:GetVariantImportJob omics:GetVariantStore omics:GetWorkflow omics:ListAnnotationImportJobs omics:ListAnnotationStores omics:ListMultipartReadSetUploads

Prefixo do serviço	Ações
	omics:ListReadSetActivationJobs omics:ListReadSetExportJobs omics:ListReadSetImportJobs omics:ListReadSets omics:ListReadSetUploadParts omics:ListReferenceImportJobs omics:ListReferences omics:ListReferenceStores omics:ListRunGroups omics:ListRuns omics:ListRunTasks omics:ListSequenceStores omics:ListVariantImportJobs omics:ListVariantStores omics:ListWorkflows omics:StartAnnotationImportJob omics:StartReadSetActivationJob omics:StartReadSetExportJob omics:StartReadSetImportJob omics:StartReferenceImportJob omics:StartRun

Prefixo do serviço	Ações
	omics:StartVariantImportJob omics:UpdateAnnotationStore omics:UpdateRunGroup omics:UpdateVariantStore omics:UpdateWorkflow omics:UploadReadSetPart

Prefixo do serviço	Ações
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
	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:DescribeStacks
	opsworks:DescribeStackSummary
	opsworks:DescribeTimeBasedAutoScaling
	opsworks:DescribeUserProfiles

Prefixo do serviço	Ações
	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
	opsworks:UpdateElasticIp
	opsworks:UpdateInstance
	opsworks:UpdateLayer
	opsworks:UpdateMyUserProfile
	opsworks:UpdateRdsDbInstance
	opsworks:UpdateStack
	opsworks:UpdateUserProfile
	opsworks:UpdateVolume

Prefixo do serviço	Ações
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
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
	organizations:DescribePolicy
	organizations:DescribeResourcePolicy
	organizations:DetachPolicy
	organizations:DisableAWSServiceAccess
	organizations:DisablePolicyType
	organizations:EnableAllFeatures
	organizations:EnableAWSServiceAccess
	organizations:EnablePolicyType
	organizations:InviteAccountToOrganization
	organizations:LeaveOrganization
	organizations:ListAccounts
	organizations:ListAccountsForParent
	organizations:ListAWSServiceAccessForOrganization
	organizations:ListChildren
	organizations:ListCreateAccountStatus
	organizations:ListDelegatedAdministrators
	organizations:ListDelegatedServicesForAccount
	organizations:ListHandshakesForAccount
	organizations:ListHandshakesForOrganization
	organizations:ListOrganizationalUnitsForParent
	organizations:ListParents

Prefixo do serviço	Ações
	organizations:ListPolicies organizations:ListPoliciesForTarget organizations:ListRoots organizations:ListTargetsForPolicy organizations:MoveAccount organizations:PutResourcePolicy organizations:RegisterDelegatedAdministrator organizations:RemoveAccountFromOrganization organizations:UpdateOrganizationalUnit organizations:UpdatePolicy

Prefixo do serviço	Ações
outposts	outposts:CancelOrder outposts:CreateOrder outposts:CreateOutpost outposts:CreatePrivateConnectivityConfig outposts:CreateSite outposts:DeleteOutpost outposts:DeleteSite outposts:GetCatalogItem outposts:GetConnection outposts:GetOrder outposts:GetOutpost outposts:GetOutpostInstanceTypes outposts:GetPrivateConnectivityConfig outposts:GetSite outposts:GetSiteAddress outposts:ListAssets outposts:ListCatalogItems outposts:ListOrders outposts:ListOutposts outposts:ListSites outposts:StartConnection

Prefixo do serviço	Ações
	outposts:UpdateOutpost outposts:UpdateSite outposts:UpdateSiteAddress outposts:UpdateSiteRackPhysicalProperties

Prefixo do serviço	Ações
panorama	panorama:CreateApplicationInstance
	panorama:CreateJobForDevices
	panorama:CreateNodeFromTemplateJob
	panorama:CreatePackage
	panorama:CreatePackageImportJob
	panorama:DeleteDevice
	panorama:DeletePackage
	panorama:DeregisterPackageVersion
	panorama:DescribeApplicationInstance
	panorama:DescribeApplicationInstanceDetails
	panorama:DescribeDevice
	panorama:DescribeDeviceJob
	panorama:DescribeNode
	panorama:DescribeNodeFromTemplateJob
	panorama:DescribePackage
	panorama:DescribePackageImportJob
	panorama:DescribePackageVersion
	panorama:ListApplicationInstanceDependencies
	panorama:ListApplicationInstanceNodeInstances
	panorama:ListApplicationInstances
	panorama:ListDevices

Prefixo do serviço	Ações
	panorama:ListDevicesJobs panorama:ListNodeFromTemplateJobs panorama:ListNodes panorama:ListPackageImportJobs panorama:ListPackages panorama:ProvisionDevice panorama:RegisterPackageVersion panorama:RemoveApplicationInstance panorama:SignalApplicationInstanceNodeInstances panorama:UpdateDeviceMetadata
pi	pi:CreatePerformanceAnalysisReport pi:DeletePerformanceAnalysisReport pi:DescribeDimensionKeys pi:GetDimensionKeyDetails pi:GetPerformanceAnalysisReport pi:GetResourceMetadata pi:GetResourceMetrics pi:ListAvailableResourceDimensions pi:ListAvailableResourceMetrics pi:ListPerformanceAnalysisReports

Prefixo do serviço	Ações
pipes	pipes:CreatePipe pipes:DeletePipe pipes:DescribePipe pipes:ListPipes pipes:StartPipe pipes:StopPipe pipes:UpdatePipe
polly	polly:DeleteLexicon polly:DescribeVoices polly:GetLexicon polly:GetSpeechSynthesisTask polly:ListLexicons polly:ListSpeechSynthesisTasks polly:PutLexicon polly:StartSpeechSynthesisTask polly:SynthesizeSpeech

Prefixo do serviço	Ações
profile	profile:AddProfileKey profile:CreateCalculatedAttributeDefinition profile:CreateDomain profile:CreateEventStream profile:CreateProfile profile:DeleteCalculatedAttributeDefinition profile:DeleteDomain profile:DeleteEventStream profile:DeleteIntegration profile:DeleteProfile profile:DeleteProfileKey profile:DeleteProfileObject profile:DeleteProfileObjectType profile:DeleteWorkflow profile:DetectProfileObjectType profile:GetAutoMergingPreview profile:GetCalculatedAttributeDefinition profile:GetCalculatedAttributeForProfile profile:GetDomain profile:GetEventStream profile:GetIdentityResolutionJob

Prefixo do serviço	Ações
	profile:GetIntegration
	profile:GetMatches
	profile:GetProfileObjectType
	profile:GetProfileObjectTypeTemplate
	profile:GetSimilarProfiles
	profile:GetWorkflow
	profile:GetWorkflowSteps
	profile:ListAccountIntegrations
	profile:ListCalculatedAttributeDefinitions
	profile:ListCalculatedAttributesForProfile
	profile:ListDomains
	profile:ListEventStreams
	profile:ListIdentityResolutionJobs
	profile:ListIntegrations
	profile:ListProfileObjects
	profile:ListProfileObjectTypes
	profile:ListProfileObjectTypeTemplates
	profile:ListRuleBasedMatches
	profile:ListWorkflows
	profile:MergeProfiles
	profile:PutIntegration

Prefixo do serviço	Ações
	profile:PutProfileObject profile:PutProfileObjectType profile:SearchProfiles profile:UpdateCalculatedAttributeDefinition profile:UpdateDomain profile:UpdateProfile

Prefixo do serviço	Ações
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
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:ListPermissions
	ram:ListPermissionVersions

Prefixo do serviço	Ações
	ram:ListPrincipals ram:ListReplacePermissionAssociationsWork ram:ListResources ram:ListResourceSharePermissions ram:ListResourceTypes 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
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:CreateDBClusterSnapshot
	rds:CreateDBParameterGroup
	rds:CreateDBProxy
	rds:CreateDBProxyEndpoint
	rds:CreateDBSecurityGroup
	rds:CreateDBSnapshot
	rds:CreateDBSubnetGroup

Prefixo do serviço	Ações
	rds:CreateEventSubscription
	rds:CreateGlobalCluster
	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

Prefixo do serviço	Ações
	rds:DescribeDBClusterAutomatedBackups
	rds:DescribeDBClusterBacktracks
	rds:DescribeDBClusterEndpoints
	rds:DescribeDBClusterParameterGroups
	rds:DescribeDBClusterParameters
	rds:DescribeDBClusters
	rds:DescribeDBClusterSnapshotAttributes
	rds:DescribeDBClusterSnapshots
	rds:DescribeDBEngineVersions
	rds:DescribeDBInstanceAutomatedBackups
	rds:DescribeDBInstances
	rds:DescribeDBLogFiles
	rds:DescribeDBParameterGroups
	rds:DescribeDBParameters
	rds:DescribeDBProxies
	rds:DescribeDBProxyEndpoints
	rds:DescribeDBProxyTargetGroups
	rds:DescribeDBProxyTargets
	rds:DescribeDBRecommendations
	rds:DescribeDBSecurityGroups
	rds:DescribeDBSnapshotAttributes

Prefixo do serviço	Ações
	rds:DescribeDBSnapshots
	rds:DescribeDbSnapshotTenantDatabases
	rds:DescribeDBSubnetGroups
	rds:DescribeEngineDefaultClusterParameters
	rds:DescribeEngineDefaultParameters
	rds:DescribeEventCategories
	rds:DescribeEvents
	rds:DescribeEventSubscriptions
	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

Prefixo do serviço	Ações
	rds:DownloadDBLogFilePortion
	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

Prefixo do serviço	Ações
	rds:PurchaseReservedDBInstancesOffering
	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

Prefixo do serviço	Ações
	rds:StopActivityStream rds:StopDBCluster rds:StopDBInstance rds:StopDBInstanceAutomatedBackupsReplication rds:SwitchoverBlueGreenDeployment rds:SwitchoverGlobalCluster rds:SwitchoverReadReplica

Prefixo do serviço	Ações
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
	redshift:CreateHsmConfiguration 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:DeleteScheduledAction redshift:DeleteSnapshotCopyGrant

Prefixo do serviço	Ações
	redshift:DeleteSnapshotSchedule redshift:DeleteUsageLimit redshift:DescribeAccountAttributes redshift:DescribeAuthenticationProfiles redshift:DescribeClusterDbRevisions redshift:DescribeClusterParameterGroups redshift:DescribeClusterParameters redshift:DescribeClusters redshift:DescribeClusterSecurityGroups redshift:DescribeClusterSnapshots redshift:DescribeClusterSubnetGroups redshift:DescribeClusterTracks redshift:DescribeClusterVersions redshift:DescribeCustomDomainAssociations redshift:DescribeDataShares redshift:DescribeDataSharesForConsumer redshift:DescribeDataSharesForProducer redshift:DescribeDefaultClusterParameters redshift:DescribeEndpointAccess redshift:DescribeEndpointAuthorization redshift:DescribeEventCategories

Prefixo do serviço	Ações
	redshift:DescribeEvents redshift:DescribeEventSubscriptions redshift:DescribeHsmClientCertificates redshift:DescribeHsmConfigurations redshift:DescribeInboundIntegrations redshift:DescribeLoggingStatus redshift:DescribeNodeConfigurationOptions redshift:DescribeOrderableClusterOptions redshift:DescribePartners redshift:DescribeRedshiftIdcApplications redshift:DescribeReservedNodeExchangeStatus redshift:DescribeReservedNodeOfferings redshift:DescribeReservedNodes redshift:DescribeResize redshift:DescribeScheduledActions redshift:DescribeSnapshotCopyGrants redshift:DescribeSnapshotSchedules redshift:DescribeStorage redshift:DescribeTableRestoreStatus redshift:DescribeUsageLimits redshift:DisableLogging

Prefixo do serviço	Ações
	redshift:DisableSnapshotCopy redshift:DisassociateDataShareConsumer redshift:EnableLogging redshift:EnableSnapshotCopy redshift:FailoverPrimaryCompute redshift:GetClusterCredentials redshift:GetClusterCredentialsWithIAM redshift:GetReservedNodeExchangeConfigurationOptions redshift:GetReservedNodeExchangeOfferings redshift:ListRecommendations redshift:ModifyAquaConfiguration redshift:ModifyAuthenticationProfile redshift:ModifyCluster redshift:ModifyClusterDbRevision redshift:ModifyClusterIamRoles redshift:ModifyClusterMaintenance redshift:ModifyClusterParameterGroup redshift:ModifyClusterSnapshot redshift:ModifyClusterSnapshotSchedule redshift:ModifyClusterSubnetGroup redshift:ModifyCustomDomainAssociation

Prefixo do serviço	Ações
	redshift:ModifyEndpointAccess redshift:ModifyEventSubscription redshift:ModifyScheduledAction redshift:ModifySnapshotCopyRetentionPeriod redshift:ModifySnapshotSchedule redshift:ModifyUsageLimit redshift:PauseCluster redshift:PurchaseReservedNodeOffering redshift:RebootCluster redshift:RejectDataShare redshift:ResetClusterParameterGroup redshift:ResizeCluster redshift:RestoreFromClusterSnapshot redshift:RestoreTableFromClusterSnapshot redshift:ResumeCluster redshift:RevokeClusterSecurityGroupIngress redshift:RevokeEndpointAccess redshift:RevokeSnapshotAccess redshift:RotateEncryptionKey redshift:UpdatePartnerStatus

Prefixo do serviço	Ações
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:ListStatements
	redshift-data:ListTables

Prefixo do serviço	Ações
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:ListEnvironments refactor-spaces:ListEnvironmentVpcs refactor-spaces:ListRoutes refactor-spaces:ListServices refactor-spaces:PutResourcePolicy refactor-spaces:UpdateRoute

Prefixo do serviço	Ações
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:DescribeProjects

Prefixo do serviço	Ações
	rekognition:DescribeProjectVersions
	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
	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
	rekognition:StartMediaAnalysisJob rekognition:StartPersonTracking rekognition:StartProjectVersion rekognition:StartSegmentDetection rekognition:StartStreamProcessor rekognition:StartTextDetection rekognition:StopProjectVersion rekognition:StopStreamProcessor rekognition:UpdateDatasetEntries rekognition:UpdateStreamProcessor

Prefixo do serviço	Ações
resiliencehub	resiliencehub:AddDraftAppVersionResourceMappings 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 resiliencehub:DescribeAppVersionTemplate resiliencehub:DescribeDraftAppVersionResourcesImportStatus

Prefixo do serviço	Ações
	resiliencehub:DescribeResiliencyPolicy resiliencehub:ImportResourcesToDraftAppVersion resiliencehub:ListAlarmRecommendations resiliencehub:ListAppAssessments resiliencehub:ListAppComponentCompliances resiliencehub:ListAppComponentRecommendations resiliencehub:ListAppInputSources resiliencehub:ListApps resiliencehub:ListAppVersionAppComponents resiliencehub:ListAppVersionResourceMappings resiliencehub:ListAppVersionResources resiliencehub:ListAppVersions resiliencehub:ListRecommendationTemplates resiliencehub:ListResiliencyPolicies resiliencehub:ListSopRecommendations resiliencehub:ListSuggestedResiliencyPolicies resiliencehub:ListTestRecommendations resiliencehub:ListUnsupportedAppVersionResources resiliencehub:PublishAppVersion resiliencehub:PutDraftAppVersionTemplate resiliencehub:RemoveDraftAppVersionResourceMappings

Prefixo do serviço	Ações
	resiliencehub:ResolveAppVersionResources resiliencehub:StartAppAssessment resiliencehub:UpdateApp resiliencehub:UpdateAppVersion resiliencehub:UpdateAppVersionAppComponent resiliencehub:UpdateAppVersionResource resiliencehub:UpdateResiliencyPolicy

Prefixo do serviço	Ações
resource-explorer-2	resource-explorer-2:AssociateDefaultView
	resource-explorer-2:BatchGetView
	resource-explorer-2:CreateIndex
	resource-explorer-2:CreateView
	resource-explorer-2:DeleteIndex
	resource-explorer-2:DeleteView
	resource-explorer-2:DisassociateDefaultView
	resource-explorer-2:GetAccountLevelServiceConfiguration
	resource-explorer-2:GetDefaultView
	resource-explorer-2:GetIndex
	resource-explorer-2:ListIndexes
	resource-explorer-2:ListIndexesForMembers
	resource-explorer-2:ListSupportedResourceTypes
	resource-explorer-2:ListViews
	resource-explorer-2:Search
	resource-explorer-2:UpdateIndexType
	resource-explorer-2:UpdateView

Prefixo do serviço	Ações
resource-groups	resource-groups:CreateGroup resource-groups:DeleteGroup resource-groups:GetAccountSettings resource-groups:GetGroup resource-groups:GetGroupConfiguration resource-groups:GetGroupQuery resource-groups:GroupResources resource-groups:ListGroupResources resource-groups:ListGroups resource-groups:PutGroupConfiguration resource-groups:SearchResources resource-groups:UngroupResources resource-groups:UpdateAccountSettings resource-groups:UpdateGroup resource-groups:UpdateGroupQuery

Prefixo do serviço	Ações
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
	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
	robomaker:ListSimulationJobs
	robomaker:ListWorldExportJobs
	robomaker:ListWorldGenerationJobs
	robomaker:ListWorlds
	robomaker:ListWorldTemplates
	robomaker:RegisterRobot
	robomaker:RestartSimulationJob
	robomaker:StartSimulationJobBatch
	robomaker:SyncDeploymentJob
	robomaker:UpdateRobotApplication
	robomaker:UpdateSimulationApplication
	robomaker:UpdateWorldTemplate

Prefixo do serviço	Ações
rolesanywhere	rolesanywhere:CreateProfile rolesanywhere:CreateTrustAnchor 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 rolesanywhere:PutNotificationSettings

Prefixo do serviço	Ações
	rolesanywhere:ResetNotificationSettings rolesanywhere:UpdateCrl rolesanywhere:UpdateProfile rolesanywhere:UpdateTrustAnchor

Prefixo do serviço	Ações
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
	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
	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
	route53:UpdateHealthCheck route53:UpdateHostedZoneComment route53:UpdateTrafficPolicyComment route53:UpdateTrafficPolicyInstance

Prefixo do serviço	Ações
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:UpdateControlPanel route53-recovery-control-config:UpdateRoutingControl

Prefixo do serviço	Ações
	route53-recovery-control-config:UpdateSafetyRule

Prefixo do serviço	Ações
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
	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
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
	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
	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 route53resolver:UpdateResolverRule

Prefixo do serviço	Ações
rum	rum:BatchCreateRumMetricDefinitions rum:BatchDeleteRumMetricDefinitions rum:BatchGetRumMetricDefinitions rum:CreateAppMonitor rum:DeleteAppMonitor rum:DeleteRumMetricsDestination rum:GetAppMonitor rum:GetAppMonitorData rum:ListAppMonitors rum:ListRumMetricsDestinations rum:PutRumMetricsDestination rum:UpdateAppMonitor rum:UpdateRumMetricDefinition

Prefixo do serviço	Ações
s3	s3:AssociateAccessGrantsIdentityCenter s3:CreateAccessGrant s3:CreateAccessGrantsInstance s3:CreateAccessGrantsLocation s3:CreateAccessPoint s3:CreateAccessPointForObjectLambda s3:CreateBucket s3:CreateJob s3:CreateMultiRegionAccessPoint s3:DeleteAccessGrant s3:DeleteAccessGrantsInstance s3:DeleteAccessGrantsInstanceResourcePolicy s3:DeleteAccessGrantsLocation s3:DeleteAccessPoint s3:DeleteAccessPointForObjectLambda s3:DeleteAccessPointPolicy s3:DeleteAccessPointPolicyForObjectLambda s3:PutAccountPublicAccessBlock s3:DeleteBucket s3:PutAnalyticsConfiguration s3:PutBucketCORS

Prefixo do serviço	Ações
	s3:PutEncryptionConfiguration s3:PutIntelligentTieringConfiguration s3:PutInventoryConfiguration s3:PutLifecycleConfiguration s3:PutMetricsConfiguration s3:PutBucketOwnershipControls s3:DeleteBucketPolicy s3:PutBucketPublicAccessBlock s3:PutReplicationConfiguration s3:DeleteBucketWebsite s3:DeleteMultiRegionAccessPoint s3:DeleteStorageLensConfiguration s3:DescribeJob s3:DescribeMultiRegionAccessPointOperation s3:DissociateAccessGrantsIdentityCenter s3:GetAccelerateConfiguration s3:GetAccessGrant s3:GetAccessGrantsInstance s3:GetAccessGrantsInstanceForPrefix s3:GetAccessGrantsInstanceResourcePolicy s3:GetAccessGrantsLocation

Prefixo do serviço	Ações
	s3:GetAccessPoint s3:GetAccessPointConfigurationForObjectLambda s3:GetAccessPointForObjectLambda s3:GetAccessPointPolicy s3:GetAccessPointPolicyForObjectLambda s3:GetAccessPointPolicyStatus s3:GetAccessPointPolicyStatusForObjectLambda s3:GetAccountPublicAccessBlock s3:GetBucketAcl s3:GetAnalyticsConfiguration s3:GetBucketCORS s3:GetEncryptionConfiguration s3:GetIntelligentTieringConfiguration s3:GetInventoryConfiguration s3:GetLifecycleConfiguration s3:GetBucketLocation s3:GetBucketLogging s3:GetMetricsConfiguration s3:GetBucketNotification s3:GetBucketObjectLockConfiguration s3:GetBucketOwnershipControls

Prefixo do serviço	Ações
	s3:GetBucketPolicy s3:GetBucketPolicyStatus s3:GetBucketPublicAccessBlock s3:GetReplicationConfiguration s3:GetBucketRequestPayment s3:GetBucketVersioning s3:GetBucketWebsite s3:GetDataAccess s3:GetMultiRegionAccessPoint s3:GetMultiRegionAccessPointPolicy s3:GetMultiRegionAccessPointPolicyStatus s3:GetMultiRegionAccessPointRoutes s3:GetObjectAttributes s3:GetStorageLensConfiguration s3:GetStorageLensDashboard s3:ListAccessGrants s3:ListAccessGrantsInstances s3:ListAccessGrantsLocations s3:ListAccessPoints s3:ListAccessPointsForObjectLambda s3:ListAllMyBuckets

Prefixo do serviço	Ações
	s3:ListJobs s3:ListBucketMultipartUploads s3:ListMultiRegionAccessPoints s3:ListStorageLensConfigurations s3:PutAccelerateConfiguration s3:PutAccessGrantsInstanceResourcePolicy s3:PutAccessPointConfigurationForObjectLambda s3:PutAccessPointPolicy s3:PutAccessPointPolicyForObjectLambda s3:PutAccountPublicAccessBlock s3:PutBucketAcl s3:PutAnalyticsConfiguration s3:PutBucketCORS s3:PutEncryptionConfiguration s3:PutIntelligentTieringConfiguration s3:PutInventoryConfiguration s3:PutLifecycleConfiguration s3:PutBucketLogging s3:PutMetricsConfiguration s3:PutBucketNotification s3:PutBucketObjectLockConfiguration

Prefixo do serviço	Ações
	s3:PutBucketOwnershipControls s3:PutBucketPolicy s3:PutBucketPublicAccessBlock s3:PutReplicationConfiguration s3:PutBucketRequestPayment s3:PutBucketVersioning s3:PutBucketWebsite s3:PutMultiRegionAccessPointPolicy s3:PutStorageLensConfiguration s3:SubmitMultiRegionAccessPointRoutes s3:UpdateAccessGrantsLocation 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
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
savingsplans	savingsplans:CreateSavingsPlan savingsplans:DeleteQueuedSavingsPlan savingsplans:DescribeSavingsPlanRates savingsplans:DescribeSavingsPlans savingsplans:DescribeSavingsPlansOfferingRates savingsplans:DescribeSavingsPlansOfferings savingsplans:ReturnSavingsPlan

Prefixo do serviço	Ações
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:ListSchemas schemas:ListSchemaVersions schemas:PutCodeBinding

Prefixo do serviço	Ações
	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
secretsmanager	secretsmanager:CancelRotateSecret secretsmanager:CreateSecret secretsmanager:DeleteResourcePolicy secretsmanager:DeleteSecret secretsmanager:DescribeSecret secretsmanager:GetRandomPassword secretsmanager:GetResourcePolicy secretsmanager:GetSecretValue secretsmanager:ListSecrets secretsmanager:ListSecretVersionIds secretsmanager:PutResourcePolicy secretsmanager:PutSecretValue secretsmanager:RemoveRegionsFromReplication secretsmanager:ReplicateSecretToRegions secretsmanager:RestoreSecret secretsmanager:RotateSecret secretsmanager:StopReplicationToReplica secretsmanager:UpdateSecret secretsmanager:ValidateResourcePolicy

Prefixo do serviço	Ações
securityhub	securityhub:AcceptAdministratorInvitation
	securityhub:AcceptInvitation
	securityhub:BatchDeleteAutomationRules
	securityhub:BatchDisableStandards
	securityhub:BatchEnableStandards
	securityhub:BatchGetAutomationRules
	securityhub:BatchGetSecurityControls
	securityhub:BatchGetStandardsControlAssociations
	securityhub:BatchImportFindings
	securityhub:BatchUpdateAutomationRules
	securityhub:BatchUpdateFindings
	securityhub:BatchUpdateStandardsControlAssociations
	securityhub:CreateActionTarget
	securityhub:CreateAutomationRule
	securityhub:CreateFindingAggregator
	securityhub:CreateInsight
	securityhub:CreateMembers
	securityhub:DeclineInvitations
	securityhub:DeleteActionTarget
	securityhub:DeleteFindingAggregator
	securityhub:DeleteInsight

Prefixo do serviço	Ações
	securityhub:DeleteInvitations
	securityhub:DeleteMembers
	securityhub:DescribeActionTargets
	securityhub:DescribeHub
	securityhub:DescribeOrganizationConfiguration
	securityhub:DescribeProducts
	securityhub:DescribeStandards
	securityhub:DisableImportFindingsForProduct
	securityhub:DisableOrganizationAdminAccount
	securityhub:DisableSecurityHub
	securityhub:DisassociateFromAdministratorAccount
	securityhub:DisassociateFromMasterAccount
	securityhub:DisassociateMembers
	securityhub:EnableImportFindingsForProduct
	securityhub:EnableOrganizationAdminAccount
	securityhub:EnableSecurityHub
	securityhub:GetAdministratorAccount
	securityhub:GetEnabledStandards
	securityhub:GetFindingAggregator
	securityhub:GetFindingHistory
	securityhub:GetFindings

Prefixo do serviço	Ações
	securityhub:GetInsightResults
	securityhub:GetInsights
	securityhub:GetInvitationsCount
	securityhub:GetMasterAccount
	securityhub:GetMembers
	securityhub:GetSecurityControlDefinition
	securityhub:InviteMembers
	securityhub:ListAutomationRules
	securityhub:ListEnabledProductsForImport
	securityhub:ListFindingAggregators
	securityhub:ListInvitations
	securityhub:ListMembers
	securityhub:ListOrganizationAdminAccounts
	securityhub:ListSecurityControlDefinitions
	securityhub:ListStandardsControlAssociations
	securityhub:StartConfigurationPolicyDisassociation
	securityhub:UpdateActionTarget
	securityhub:UpdateFindingAggregator
	securityhub:UpdateFindings
	securityhub:UpdateInsight
	securityhub:UpdateOrganizationConfiguration

Prefixo do serviço	Ações
	securityhub:UpdateSecurityControl securityhub:UpdateSecurityHubConfiguration

Prefixo do serviço	Ações
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
	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:ListApplications serverlessrepo:ListApplicationVersions serverlessrepo:PutApplicationPolicy serverlessrepo:UnshareApplication serverlessrepo:UpdateApplication

Prefixo do serviço	Ações
servicecatalog	servicecatalog:AcceptPortfolioShare servicecatalog:AssociateBudgetWithResource servicecatalog:AssociatePrincipalWithPortfolio servicecatalog:AssociateProductWithPortfolio servicecatalog:AssociateServiceActionWithProvisioningArtifact servicecatalog:BatchAssociateServiceActionWithProvisioningArtifact servicecatalog:BatchDisassociateServiceActionFromProvisioningArtifact servicecatalog:CopyProduct servicecatalog:CreateConstraint servicecatalog:CreatePortfolio servicecatalog:CreatePortfolioShare servicecatalog:CreateProduct servicecatalog:CreateProvisionedProductPlan servicecatalog:CreateProvisioningArtifact servicecatalog:CreateServiceAction servicecatalog:DeleteConstraint servicecatalog:DeletePortfolio servicecatalog:DeletePortfolioShare servicecatalog:DeleteProduct servicecatalog:DeleteProvisionedProductPlan

Prefixo do serviço	Ações
	servicecatalog:DeleteProvisioningArtifact servicecatalog:DeleteServiceAction servicecatalog:DescribeConstraint servicecatalog:DescribeCopyProductStatus servicecatalog:DescribePortfolio servicecatalog:DescribePortfolioShares servicecatalog:DescribePortfolioShareStatus servicecatalog:DescribeProduct servicecatalog:DescribeProductAsAdmin servicecatalog:DescribeProductView servicecatalog:DescribeProvisionedProductPlan servicecatalog:DescribeProvisioningArtifact servicecatalog:DescribeProvisioningParameters servicecatalog:DescribeRecord servicecatalog:DescribeServiceAction servicecatalog:DescribeServiceActionExecutionParameters servicecatalog:DisableAWSOrganizationsAccess servicecatalog:DisassociateBudgetFromResource servicecatalog:DisassociatePrincipalFromPortfolio servicecatalog:DisassociateProductFromPortfolio servicecatalog:DisassociateServiceActionFromProvisioningArtifact

Prefixo do serviço	Ações
	servicecatalog:EnableAWSOrganizationsAccess servicecatalog:ExecuteProvisionedProductPlan servicecatalog:ExecuteProvisionedProductServiceAction servicecatalog:GetAWSOrganizationsAccessStatus servicecatalog:GetProvisionedProductOutputs servicecatalog:ImportAsProvisionedProduct servicecatalog:ListAcceptedPortfolioShares servicecatalog:ListBudgetsForResource servicecatalog:ListConstraintsForPortfolio servicecatalog:ListLaunchPaths servicecatalog:ListOrganizationPortfolioAccess servicecatalog:ListPortfolioAccess servicecatalog:ListPortfolios servicecatalog:ListPortfoliosForProduct servicecatalog:ListPrincipalsForPortfolio servicecatalog:ListProvisionedProductPlans servicecatalog:ListProvisioningArtifacts servicecatalog:ListProvisioningArtifactsForServiceAction servicecatalog:ListRecordHistory servicecatalog:ListServiceActions servicecatalog:ListServiceActionsForProvisioningArtifact

Prefixo do serviço	Ações
	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 servicecatalog:UpdateProvisionedProductProperties servicecatalog:UpdateProvisioningArtifact servicecatalog:UpdateServiceAction

Prefixo do serviço	Ações
servicediscovery	servicediscovery:CreateHttpNamespace
	servicediscovery:CreatePrivateDnsNamespace
	servicediscovery:CreatePublicDnsNamespace
	servicediscovery:CreateService
	servicediscovery:DeleteNamespace
	servicediscovery:DeleteService
	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
	servicediscovery:UpdatePublicDnsNamespace

Prefixo do serviço	Ações
	servicediscovery:UpdateService
servicequotas	servicequotas:AssociateServiceQuotaTemplate servicequotas:DeleteServiceQuotaIncreaseRequestFromTemplate servicequotas:DisassociateServiceQuotaTemplate servicequotas:GetAssociationForServiceQuotaTemplate servicequotas:GetAWSDefaultServiceQuota servicequotas:GetRequestedServiceQuotaChange servicequotas:GetServiceQuota servicequotas:GetServiceQuotaIncreaseRequestFromTemplate servicequotas:ListAWSDefaultServiceQuotas servicequotas:ListRequestedServiceQuotaChangeHistory servicequotas:ListRequestedServiceQuotaChangeHistoryByQuota servicequotas:ListServiceQuotaIncreaseRequestsInTemplate servicequotas:ListServiceQuotas servicequotas:ListServices servicequotas:PutServiceQuotaIncreaseRequestIntoTemplate servicequotas:RequestServiceQuotaIncrease

Prefixo do serviço	Ações
ses	ses:BatchGetMetricData
	ses:CloneReceiptRuleSet
	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:CreateReceiptFilter
	ses:CreateReceiptRule
	ses:CreateReceiptRuleSet
	ses:CreateTemplate
	ses:DeleteConfigurationSet
	ses:DeleteConfigurationSetEventDestination
	ses:DeleteConfigurationSetTrackingOptions

Prefixo do serviço	Ações
	ses:DeleteContact
	ses:DeleteContactList
	ses:DeleteCustomVerificationEmailTemplate
	ses:DeleteDedicatedIpPool
	ses:DeleteEmailIdentity
	ses:DeleteEmailIdentityPolicy
	ses:DeleteEmailTemplate
	ses:DeleteIdentity
	ses:DeleteIdentityPolicy
	ses:DeleteReceiptFilter
	ses:DeleteReceiptRule
	ses:DeleteReceiptRuleSet
	ses:DeleteSuppressedDestination
	ses:DeleteTemplate
	ses:DeleteVerifiedEmailAddress
	ses:DescribeActiveReceiptRuleSet
	ses:DescribeConfigurationSet
	ses:DescribeReceiptRule
	ses:DescribeReceiptRuleSet
	ses:GetAccount
	ses:GetAccountSendingEnabled

Prefixo do serviço	Ações
	ses:GetBlacklistReports
	ses:GetConfigurationSet
	ses:GetConfigurationSetEventDestinations
	ses:GetContact
	ses:GetContactList
	ses:GetCustomVerificationEmailTemplate
	ses:GetDedicatedIp
	ses:GetDedicatedIpPool
	ses:GetDedicatedIps
	ses:GetDeliverabilityDashboardOptions
	ses:GetDeliverabilityTestReport
	ses:GetDomainDeliverabilityCampaign
	ses:GetDomainStatisticsReport
	ses:GetEmailIdentity
	ses:GetEmailIdentityPolicies
	ses:GetEmailTemplate
	ses:GetIdentityDkimAttributes
	ses:GetIdentityMailFromDomainAttributes
	ses:GetIdentityNotificationAttributes
	ses:GetIdentityPolicies
	ses:GetIdentityVerificationAttributes

Prefixo do serviço	Ações
	ses:GetImportJob
	ses:GetMessageInsights
	ses:GetSendQuota
	ses:GetSendStatistics
	ses:GetSuppressedDestination
	ses:GetTemplate
	ses:ListConfigurationSets
	ses:ListContactLists
	ses:ListContacts
	ses:ListCustomVerificationEmailTemplates
	ses:ListDedicatedIpPools
	ses:ListDeliverabilityTestReports
	ses:ListDomainDeliverabilityCampaigns
	ses:ListEmailIdentities
	ses:ListEmailTemplates
	ses:ListExportJobs
	ses:ListIdentities
	ses:ListIdentityPolicies
	ses:ListImportJobs
	ses:ListReceiptFilters
	ses:ListReceiptRuleSets

Prefixo do serviço	Ações
	ses:ListRecommendations
	ses:ListSuppressedDestinations
	ses:ListTemplates
	ses:ListVerifiedEmailAddresses
	ses:PutAccountDedicatedIpWarmupAttributes
	ses:PutAccountDetails
	ses:PutAccountSendingAttributes
	ses:PutAccountSuppressionAttributes
	ses:PutAccountVdmAttributes
	ses:PutConfigurationSetDeliveryOptions
	ses:PutConfigurationSetReputationOptions
	ses:PutConfigurationSetSendingOptions
	ses:PutConfigurationSetSuppressionOptions
	ses:PutConfigurationSetTrackingOptions
	ses:PutConfigurationSetVdmOptions
	ses:PutDedicatedIpInPool
	ses:PutDedicatedIpPoolScalingAttributes
	ses:PutDedicatedIpWarmupAttributes
	ses:PutDeliverabilityDashboardOption
	ses:PutEmailIdentityConfigurationSetAttributes
	ses:PutEmailIdentityDkimAttributes

Prefixo do serviço	Ações
	ses:PutEmailIdentityDkimSigningAttributes
	ses:PutEmailIdentityFeedbackAttributes
	ses:PutEmailIdentityMailFromAttributes
	ses:PutIdentityPolicy
	ses:PutSuppressedDestination
	ses:ReorderReceiptRuleSet
	ses:SendBounce
	ses:SendCustomVerificationEmail
	ses:SetActiveReceiptRuleSet
	ses:SetIdentityDkimEnabled
	ses:SetIdentityFeedbackForwardingEnabled
	ses:SetIdentityHeadersInNotificationsEnabled
	ses:SetIdentityMailFromDomain
	ses:SetIdentityNotificationTopic
	ses:SetReceiptRulePosition
	ses:TestRenderEmailTemplate
	ses:TestRenderTemplate
	ses:UpdateAccountSendingEnabled
	ses:UpdateConfigurationSetEventDestination
	ses:UpdateConfigurationSetReputationMetricsEnabled
	ses:UpdateConfigurationSetSendingEnabled

Prefixo do serviço	Ações
	ses:UpdateConfigurationSetTrackingOptions ses:UpdateContact ses:UpdateContactList ses:UpdateCustomVerificationEmailTemplate ses:UpdateEmailIdentityPolicy ses:UpdateEmailTemplate ses:UpdateReceiptRule ses:UpdateTemplate ses:VerifyDomainDkim ses:VerifyDomainIdentity ses:VerifyEmailAddress ses:VerifyEmailIdentity

Prefixo do serviço	Ações
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
	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
signer	signer:AddProfilePermission
	signer:CancelSigningProfile
	signer:DescribeSigningJob
	signer:GetRevocationStatus
	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
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
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
	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
sms-voice	sms-voice:CreateConfigurationSet sms-voice:CreateConfigurationSetEventDestination sms-voice:CreateEventDestination sms-voice:CreateOptOutList sms-voice:CreatePool sms-voice:CreateRegistration sms-voice:CreateRegistrationAssociation sms-voice:CreateRegistrationAttachment sms-voice:CreateRegistrationVersion sms-voice:CreateVerifiedDestinationNumber sms-voice:DeleteConfigurationSet sms-voice:DeleteConfigurationSetEventDestination sms-voice:DeleteDefaultMessageType sms-voice:DeleteDefaultSenderId sms-voice:DeleteEventDestination sms-voice:DeleteKeyword sms-voice:DeleteOptedOutNumber sms-voice:DeleteOptOutList sms-voice:DeletePool sms-voice:DeleteRegistration sms-voice:DeleteRegistrationAttachment

Prefixo do serviço	Ações
	sms-voice:DeleteTextMessageSpendLimitOverride sms-voice:DeleteVerifiedDestinationNumber sms-voice:DeleteVoiceMessageSpendLimitOverride sms-voice:DescribeAccountAttributes sms-voice:DescribeAccountLimits sms-voice:DescribeConfigurationSets sms-voice:DescribeKeywords sms-voice:DescribeOptedOutNumbers sms-voice:DescribeOptOutLists sms-voice:DescribePhoneNumbers sms-voice:DescribePools sms-voice:DescribeRegistrationAttachments sms-voice:DescribeRegistrationFieldDefinitions sms-voice:DescribeRegistrationFieldValues sms-voice:DescribeRegistrations sms-voice:DescribeRegistrationSectionDefinitions sms-voice:DescribeRegistrationTypeDefinitions sms-voice:DescribeRegistrationVersions sms-voice:DescribeSenderIds sms-voice:DescribeSpendLimits sms-voice:DescribeVerifiedDestinationNumbers

Prefixo do serviço	Ações
	sms-voice:DisassociateOriginationIdentity sms-voice:DiscardRegistrationVersion sms-voice:GetConfigurationSetEventDestinations sms-voice:ListConfigurationSets sms-voice:ListPoolOriginationIdentities sms-voice:ListRegistrationAssociations sms-voice:PutKeyword sms-voice:PutOptedOutNumber sms-voice:ReleasePhoneNumber sms-voice:ReleaseSenderId sms-voice:RequestPhoneNumber sms-voice:RequestSenderId sms-voice:SendDestinationNumberVerificationCode sms-voice:SetDefaultMessageType sms-voice:SetDefaultSenderId sms-voice:SetTextMessageSpendLimitOverride sms-voice:SetVoiceMessageSpendLimitOverride sms-voice:SubmitRegistrationVersion sms-voice:UpdateConfigurationSetEventDestination sms-voice:UpdateEventDestination sms-voice:UpdatePhoneNumber

Prefixo do serviço	Ações
	sms-voice:UpdatePool sms-voice:UpdateSenderId

Prefixo do serviço	Ações
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
	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
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:DeleteDocument
	ssm:DeleteInventory
	ssm:DeleteMaintenanceWindow
	ssm:DeleteOpsItem
	ssm:DeleteOpsMetadata
	ssm:DeleteParameter
	ssm:DeleteParameters

Prefixo do serviço	Ações
	ssm:DeletePatchBaseline
	ssm:DeleteResourceDataSync
	ssm:DeleteResourcePolicy
	ssm:DeregisterManagedInstance
	ssm:DeregisterPatchBaselineForPatchGroup
	ssm:DeregisterTargetFromMaintenanceWindow
	ssm:DeregisterTaskFromMaintenanceWindow
	ssm:DescribeActivations
	ssm:DescribeAssociation
	ssm:DescribeAssociationExecutions
	ssm:DescribeAssociationExecutionTargets
	ssm:DescribeAutomationExecutions
	ssm:DescribeAutomationStepExecutions
	ssm:DescribeAvailablePatches
	ssm:DescribeDocument
	ssm:DescribeDocumentParameters
	ssm:DescribeDocumentPermission
	ssm:DescribeEffectiveInstanceAssociations
	ssm:DescribeEffectivePatchesForPatchBaseline
	ssm:DescribeInstanceAssociationsStatus
	ssm:DescribeInstanceInformation

Prefixo do serviço	Ações
	ssm:DescribeInstancePatches
	ssm:DescribeInstancePatchStates
	ssm:DescribeInstancePatchStatesForPatchGroup
	ssm:DescribeInstanceProperties
	ssm:DescribeInventoryDeletions
	ssm:DescribeMaintenanceWindowExecutions
	ssm:DescribeMaintenanceWindowExecutionTaskInvocations
	ssm:DescribeMaintenanceWindowExecutionTasks
	ssm:DescribeMaintenanceWindows
	ssm:DescribeMaintenanceWindowSchedule
	ssm:DescribeMaintenanceWindowsForTarget
	ssm:DescribeMaintenanceWindowTargets
	ssm:DescribeMaintenanceWindowTasks
	ssm:DescribeOpsItems
	ssm:DescribeParameters
	ssm:DescribePatchBaselines
	ssm:DescribePatchGroups
	ssm:DescribePatchGroupState
	ssm:DescribePatchProperties
	ssm:DescribeSessions
	ssm:DisassociateOpsItemRelatedItem

Prefixo do serviço	Ações
	ssm:GetAutomationExecution
	ssm:GetCalendarState
	ssm:GetCommandInvocation
	ssm:GetConnectionStatus
	ssm:GetDefaultPatchBaseline
	ssm:GetDeployablePatchSnapshotForInstance
	ssm:GetDocument
	ssm:GetInventory
	ssm:GetInventorySchema
	ssm:GetMaintenanceWindow
	ssm:GetMaintenanceWindowExecution
	ssm:GetMaintenanceWindowExecutionTask
	ssm:GetMaintenanceWindowExecutionTaskInvocation
	ssm:GetMaintenanceWindowTask
	ssm:GetOpsItem
	ssm:GetOpsMetadata
	ssm:GetOpsSummary
	ssm:GetParameter
	ssm:GetParameterHistory
	ssm:GetParameters
	ssm:GetParametersByPath

Prefixo do serviço	Ações
	ssm:GetPatchBaseline
	ssm:GetPatchBaselineForPatchGroup
	ssm:GetResourcePolicies
	ssm:GetServiceSetting
	ssm:LabelParameterVersion
	ssm:ListAssociations
	ssm:ListAssociationVersions
	ssm:ListCommandInvocations
	ssm:ListCommands
	ssm:ListComplianceItems
	ssm:ListComplianceSummaries
	ssm:ListDocumentMetadataHistory
	ssm:ListDocuments
	ssm:ListDocumentVersions
	ssm:ListInstanceAssociations
	ssm:ListInventoryEntries
	ssm:ListOpsItemEvents
	ssm:ListOpsItemRelatedItems
	ssm:ListOpsMetadata
	ssm:ListResourceComplianceSummaries
	ssm:ListResourceDataSync

Prefixo do serviço	Ações
	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
	ssm:StartAssociationsOnce
	ssm:StartAutomationExecution
	ssm:StartChangeRequestExecution
	ssm:StartSession
	ssm:StopAutomationExecution
	ssm:TerminateSession
	ssm:UnlabelParameterVersion

Prefixo do serviço	Ações
	ssm:UpdateAssociation
	ssm:UpdateAssociationStatus
	ssm:UpdateDocument
	ssm:UpdateDocumentDefaultVersion
	ssm:UpdateDocumentMetadata
	ssm:UpdateInstanceInformation
	ssm:UpdateMaintenanceWindow
	ssm:UpdateMaintenanceWindowTarget
	ssm:UpdateMaintenanceWindowTask
	ssm:UpdateManagedInstanceRole
	ssm:UpdateOpsItem
	ssm:UpdateOpsMetadata
	ssm:UpdatePatchBaseline
	ssm:UpdateResourceDataSync
	ssm:UpdateServiceSetting

Prefixo do serviço	Ações
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
	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
ssm-sap	ssm-sap:BackupDatabase
	ssm-sap:DeleteResourcePermission
	ssm-sap:DeregisterApplication
	ssm-sap:GetApplication
	ssm-sap:GetComponent
	ssm-sap:GetDatabase
	ssm-sap:GetOperation
	ssm-sap:GetResourcePermission
	ssm-sap:ListApplications
	ssm-sap:ListComponents
	ssm-sap:ListDatabases
	ssm-sap:ListOperations
	ssm-sap:PutResourcePermission
	ssm-sap:RegisterApplication
	ssm-sap:RestoreDatabase
	ssm-sap:StartApplicationRefresh
	ssm-sap:UpdateApplicationSettings
	ssm-sap:UpdateHANABackupSettings

Prefixo do serviço	Ações
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:ListStateMachines states:ListStateMachineVersions states:SendTaskFailure

Prefixo do serviço	Ações
	states:SendTaskHeartbeat states:SendTaskSuccess states:StartExecution states:StopExecution states:UpdateMapRun states:UpdateStateMachine states:UpdateStateMachineAlias
sts	sts:AssumeRole sts:AssumeRoleWithSAML sts:AssumeRoleWithWebIdentity sts:DecodeAuthorizationMessage sts:GetAccessKeyInfo sts:GetCallerIdentity sts:GetFederationToken sts:GetSessionToken

Prefixo do serviço	Ações
swf	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
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:StopCanary
	synthetics:UpdateCanary

Prefixo do serviço	Ações
tag	tag:DescribeReportCreation tag:GetComplianceSummary tag:GetResources tag:StartReportCreation

Prefixo do serviço	Ações
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:ListAdapters
	textract:ListAdapterVersions
	textract:StartDocumentAnalysis
	textract:StartDocumentTextDetection
	textract:StartExpenseAnalysis
	textract:StartLendingAnalysis

Prefixo do serviço	Ações
	textract:UpdateAdapter
timestream	timestream:CancelQuery timestream:CreateDatabase timestream:CreateScheduledQuery timestream:CreateTable timestream:DeleteDatabase timestream:DeleteScheduledQuery timestream:DeleteTable timestream:DescribeDatabase timestream:DescribeScheduledQuery timestream:DescribeTable timestream:ExecuteScheduledQuery timestream:ListBatchLoadTasks timestream:ListDatabases timestream:ListScheduledQueries timestream:ListTables timestream:PrepareQuery timestream:UpdateDatabase timestream:UpdateScheduledQuery timestream:UpdateTable

Prefixo do serviço	Ações
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
	tnb:ListSolNetworkPackages tnb:PutSolFunctionPackageContent tnb:PutSolNetworkPackageContent tnb:TerminateSolNetworkInstance tnb:UpdateSolFunctionPackage tnb:UpdateSolNetworkInstance tnb:UpdateSolNetworkPackage tnb:ValidateSolFunctionPackageContent tnb:ValidateSolNetworkPackageContent

Prefixo do serviço	Ações
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
	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
	transcribe:UpdateCallAnalyticsCategory transcribe:UpdateMedicalVocabulary transcribe:UpdateVocabulary transcribe:UpdateVocabularyFilter

Prefixo do serviço	Ações
transferência	transfer:CreateAccess
	transfer:CreateAgreement
	transfer:CreateConnector
	transfer:CreateProfile
	transfer:CreateServer
	transfer:CreateUser
	transfer:CreateWorkflow
	transfer:DeleteAccess
	transfer:DeleteAgreement
	transfer:DeleteCertificate
	transfer:DeleteConnector
	transfer:DeleteHostKey
	transfer:DeleteProfile
	transfer:DeleteServer
	transfer:DeleteSshPublicKey
	transfer:DeleteUser
	transfer:DeleteWorkflow
	transfer:DescribeAccess
	transfer:DescribeAgreement
	transfer:DescribeCertificate
	transfer:DescribeConnector

Prefixo do serviço	Ações
	transfer:DescribeExecution
	transfer:DescribeHostKey
	transfer:DescribeProfile
	transfer:DescribeSecurityPolicy
	transfer:DescribeServer
	transfer:DescribeUser
	transfer:DescribeWorkflow
	transfer:ImportCertificate
	transfer:ImportHostKey
	transfer:ImportSshPublicKey
	transfer:ListAccesses
	transfer:ListCertificates
	transfer:ListConnectors
	transfer:ListExecutions
	transfer:ListHostKeys
	transfer:ListProfiles
	transfer:ListSecurityPolicies
	transfer:ListServers
	transfer:ListUsers
	transfer:ListWorkflows
	transfer:SendWorkflowStepState

Prefixo do serviço	Ações
	transfer:StartFileTransfer
	transfer:StartServer
	transfer:StopServer
	transfer:TestConnection
	transfer:TestIdentityProvider
	transfer:UpdateAccess
	transfer:UpdateAgreement
	transfer:UpdateCertificate
	transfer:UpdateConnector
	transfer:UpdateHostKey
	transfer:UpdateProfile
	transfer:UpdateServer
	transfer:UpdateUser

Prefixo do serviço	Ações
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
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
	voiceid:OptOutSpeaker voiceid:StartFraudsterRegistrationJob voiceid:StartSpeakerEnrollmentJob voiceid:UpdateDomain voiceid:UpdateWatchlist

Prefixo do serviço	Ações
vpc-lattice	vpc-lattice:CreateAccessLogSubscription vpc-lattice:CreateListener vpc-lattice:CreateRule vpc-lattice:CreateService vpc-lattice:CreateServiceNetwork vpc-lattice:CreateServiceNetworkServiceAssociation vpc-lattice:CreateServiceNetworkVpcAssociation vpc-lattice:CreateTargetGroup vpc-lattice:DeleteAccessLogSubscription vpc-lattice:DeleteAuthPolicy vpc-lattice:DeleteListener vpc-lattice:DeleteResourcePolicy vpc-lattice:DeleteRule vpc-lattice:DeleteService vpc-lattice:DeleteServiceNetwork vpc-lattice:DeleteServiceNetworkServiceAssociation vpc-lattice:DeleteServiceNetworkVpcAssociation vpc-lattice:DeleteTargetGroup vpc-lattice:DeregisterTargets vpc-lattice:GetAccessLogSubscription vpc-lattice:GetAuthPolicy

Prefixo do serviço	Ações
	vpc-lattice:GetListener vpc-lattice:GetResourcePolicy vpc-lattice:GetRule vpc-lattice:GetService vpc-lattice:GetServiceNetwork vpc-lattice:GetServiceNetworkServiceAssociation vpc-lattice:GetServiceNetworkVpcAssociation vpc-lattice:GetTargetGroup vpc-lattice:ListAccessLogSubscriptions vpc-lattice:ListListeners vpc-lattice:ListRules vpc-lattice:ListServiceNetworks vpc-lattice:ListServiceNetworkServiceAssociations vpc-lattice:ListServiceNetworkVpcAssociations vpc-lattice:ListServices vpc-lattice:ListTargetGroups vpc-lattice:ListTargets vpc-lattice:PutAuthPolicy vpc-lattice:PutResourcePolicy vpc-lattice:RegisterTargets vpc-lattice:UpdateAccessLogSubscription

Prefixo do serviço	Ações
	vpc-lattice:UpdateListener vpc-lattice:UpdateRule vpc-lattice:UpdateService vpc-lattice:UpdateServiceNetwork vpc-lattice:UpdateServiceNetworkVpcAssociation vpc-lattice:UpdateTargetGroup

Prefixo do serviço	Ações
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
	wafv2:GetIPSet
	wafv2:GetLoggingConfiguration
	wafv2:GetManagedRuleSet
	wafv2:GetMobileSdkRelease
	wafv2:GetPermissionPolicy
	wafv2:GetRateBasedStatementManagedKeys
	wafv2:GetRegexPatternSet
	wafv2:GetRuleGroup
	wafv2:GetSampledRequests
	wafv2:GetWebACLForResource
	wafv2:ListAPIKeys
	wafv2:ListAvailableManagedRuleGroups
	wafv2:ListAvailableManagedRuleGroupVersions
	wafv2:ListIPSets
	wafv2:ListLoggingConfigurations
	wafv2:ListManagedRuleSets
	wafv2:ListMobileSdkReleases
	wafv2:ListRegexPatternSets
	wafv2:ListResourcesForWebACL
	wafv2:ListRuleGroups
	wafv2:ListWebACLs

Prefixo do serviço	Ações
	wafv2:PutLoggingConfiguration wafv2:PutManagedRuleSetVersions wafv2:PutPermissionPolicy wafv2:UpdateIPSet wafv2:UpdateManagedRuleSetVersionExpiryDate wafv2:UpdateRegexPatternSet wafv2:UpdateRuleGroup wafv2:UpdateWebACL

Prefixo do serviço	Ações
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
	wellarchitected:GetAnswer wellarchitected:GetConsolidatedReport 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:ListLenses wellarchitected:ListLensReviewImprovements wellarchitected:ListLensReviews wellarchitected:ListLensShares

Prefixo do serviço	Ações
	wellarchitected:ListMilestones wellarchitected:ListNotifications wellarchitected:ListProfileNotifications wellarchitected:ListProfiles wellarchitected:ListProfileShares wellarchitected:ListReviewTemplateAnswers wellarchitected:ListReviewTemplates wellarchitected:ListShareInvitations wellarchitected:ListTemplateShares wellarchitected:ListWorkloads wellarchitected:ListWorkloadShares wellarchitected:UpdateAnswer wellarchitected:UpdateGlobalSettings wellarchitected:UpdateLensReview wellarchitected:UpdateProfile wellarchitected:UpdateReviewTemplate wellarchitected:UpdateReviewTemplateLensReview wellarchitected:UpdateShareInvitation wellarchitected:UpdateWorkload wellarchitected:UpdateWorkloadShare wellarchitected:UpgradeLensReview

Prefixo do serviço	Ações
	wellarchitected:UpgradeProfileVersion wellarchitected:UpgradeReviewTemplateLensReview

Prefixo do serviço	Ações
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:GetContentSummary
	wisdom:GetImportJob
	wisdom:GetKnowledgeBase
	wisdom:GetRecommendations
	wisdom:GetSession
	wisdom:ListAssistantAssociations

Prefixo do serviço	Ações
	wisdom:ListAssistants
	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

Prefixo do serviço	Ações
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
	worklink:RestoreDomainAccess worklink:RevokeDomainAccess worklink:SignOutUser worklink:UpdateAuditStreamConfiguration worklink:UpdateCompanyNetworkConfiguration worklink:UpdateDevicePolicyConfiguration worklink:UpdateDomainMetadata worklink:UpdateFleetMetadata worklink:UpdateIdentityProviderConfiguration

Prefixo do serviço	Ações
espaços de trabalho	workspaces:AssociateConnectionAlias
	workspaces:AssociateIpGroups
	workspaces:AssociateWorkspaceApplication
	workspaces:CopyWorkspaceImage
	workspaces:CreateConnectClientAddIn
	workspaces:CreateConnectionAlias
	workspaces:CreateIpGroup
	workspaces:CreateStandbyWorkspaces
	workspaces:CreateUpdatedWorkspaceImage
	workspaces:CreateWorkspaceBundle
	workspaces:CreateWorkspaceImage
	workspaces:CreateWorkspaces
	workspaces:DeleteClientBranding
	workspaces:DeleteConnectClientAddIn
	workspaces:DeleteConnectionAlias
	workspaces:DeleteIpGroup
	workspaces:DeleteWorkspaceBundle
	workspaces:DeleteWorkspaceImage
	workspaces:DeployWorkspaceApplications
	workspaces:DeregisterWorkspaceDirectory
	workspaces:DescribeAccount

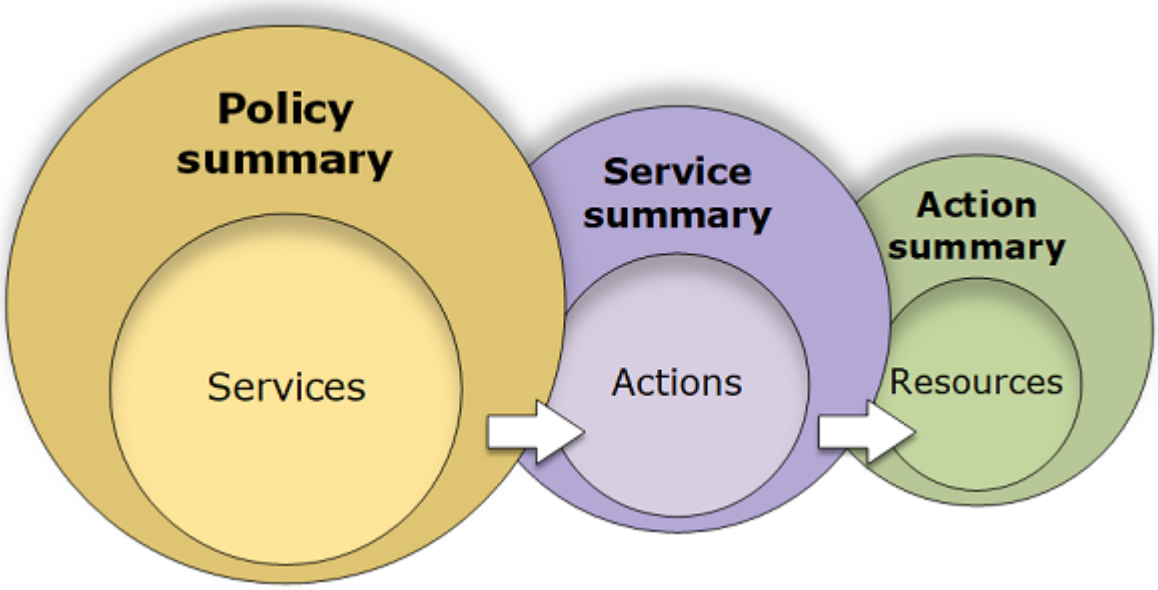
Prefixo do serviço	Ações
	workspaces:DescribeAccountModifications
	workspaces:DescribeApplicationAssociations
	workspaces:DescribeApplications
	workspaces:DescribeBundleAssociations
	workspaces:DescribeClientBranding
	workspaces:DescribeClientProperties
	workspaces:DescribeConnectClientAddIns
	workspaces:DescribeConnectionAliases
	workspaces:DescribeConnectionAliasPermissions
	workspaces:DescribeImageAssociations
	workspaces:DescribeIpGroups
	workspaces:DescribeWorkspaceAssociations
	workspaces:DescribeWorkspaceBundles
	workspaces:DescribeWorkspaceDirectories
	workspaces:DescribeWorkspaceImagePermissions
	workspaces:DescribeWorkspaces
	workspaces:DescribeWorkspacesConnectionStatus
	workspaces:DescribeWorkspaceSnapshots
	workspaces:DisassociateConnectionAlias
	workspaces:DisassociateIpGroups
	workspaces:DisassociateWorkspaceApplication

Prefixo do serviço	Ações
	workspaces:ImportClientBranding
	workspaces:ImportWorkspaceImage
	workspaces:ListAvailableManagementCidrRanges
	workspaces:MigrateWorkspace
	workspaces:ModifyAccount
	workspaces:ModifyCertificateBasedAuthProperties
	workspaces:ModifyClientProperties
	workspaces:ModifySamlProperties
	workspaces:ModifySelfservicePermissions
	workspaces:ModifyWorkspaceAccessProperties
	workspaces:ModifyWorkspaceCreationProperties
	workspaces:ModifyWorkspaceProperties
	workspaces:ModifyWorkspaceState
	workspaces:RebootWorkspaces
	workspaces:RebuildWorkspaces
	workspaces:RegisterWorkspaceDirectory
	workspaces:RestoreWorkspace
	workspaces:StartWorkspaces
	workspaces:StopWorkspaces
	workspaces:TerminateWorkspaces
	workspaces:UpdateConnectClientAddIn

Prefixo do serviço	Ações
	workspaces:UpdateConnectionAliasPermission workspaces:UpdateWorkspaceBundle workspaces:UpdateWorkspaceImagePermission
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

Noções básicas sobre as permissões concedidas por uma política

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.

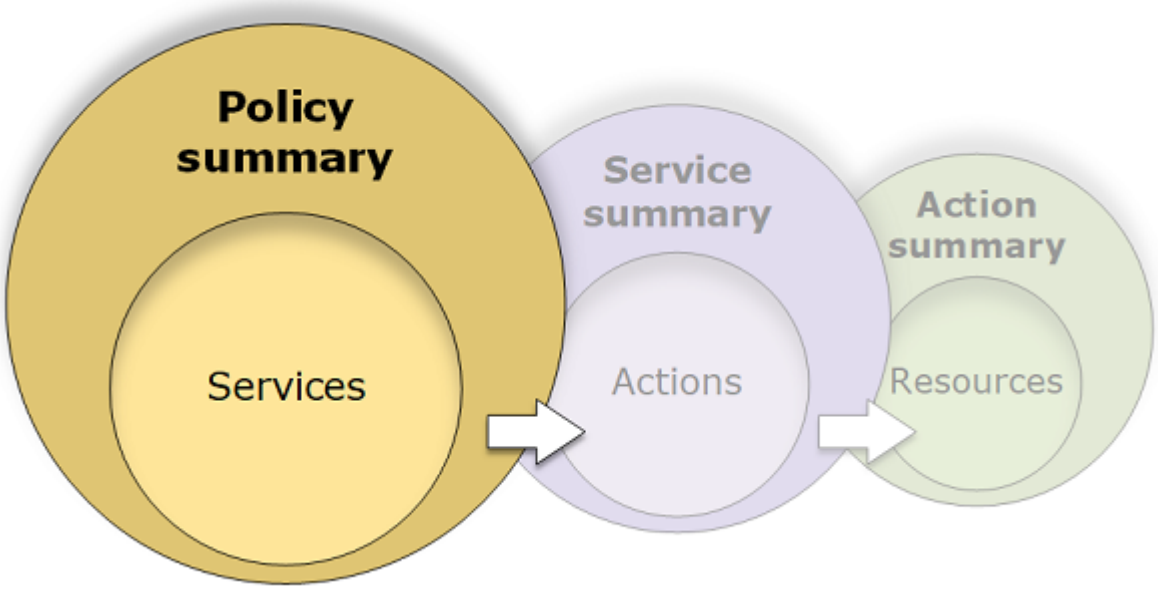
Tópicos

- Resumo da política (lista de serviços)
- 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`).

Visualizar resumos de políticas

Você pode visualizar os resumos para qualquer política anexada a um usuário escolhendo o nome da política na guia Permissões na página de detalhes do usuário. Você pode visualizar os resumos para qualquer política anexada a um perfil escolhendo o nome da política na guia Permissões na página de detalhes do perfil. Visualize o resumo de políticas para políticas gerenciadas na página Políticas. Se a política não incluir um resumo, consulte Resumo de política ausente para saber por quê.

Para visualizar o resumo da política na página Políticas

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha 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.

Para visualizar o resumo de uma política anexada a um usuário

1. Faça login no AWS Management Console 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.

Para visualizar o resumo de uma política anexada a uma função

1. Faça login no AWS Management Console 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 “Edição de políticas do IAM”. Você não pode editar políticas gerenciadas pela AWS.

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. Selecione a opção 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.

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.

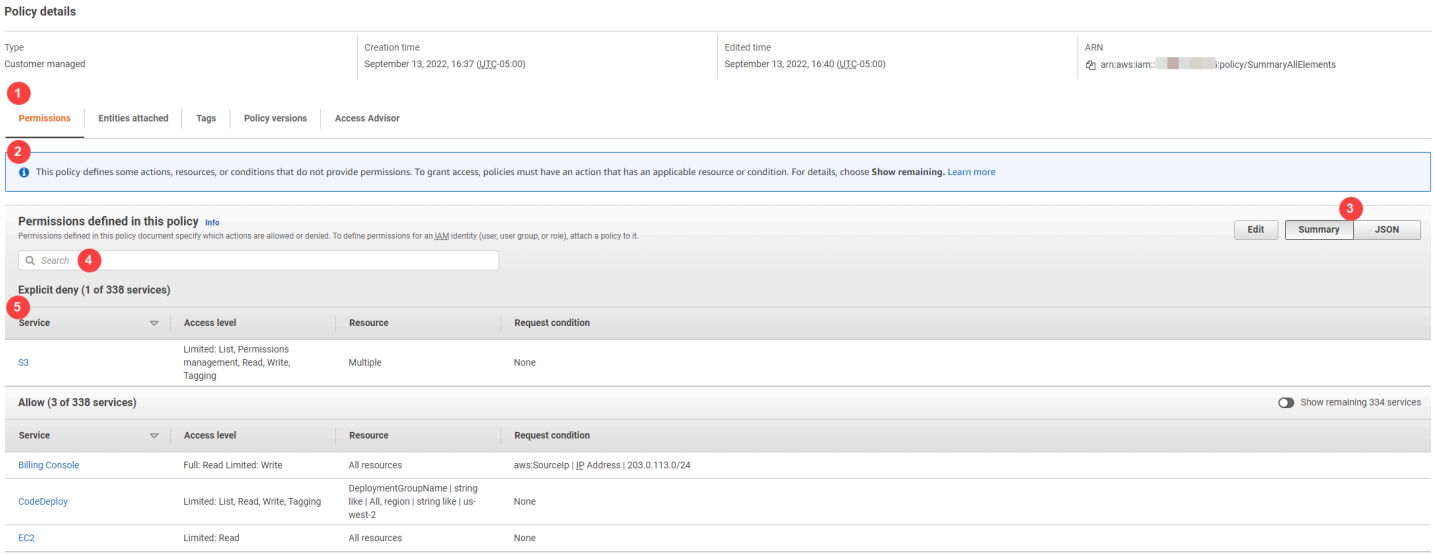
> **Note**
>
> Você pode alternar entre as opções de editor Visual e JSON a qualquer momento. Porém, se você fizer alterações ou escolher Avançar na opção de editor Visual, o IAM poderá reestruturar a política a fim de otimizá-la para o editor visual. Para ter mais informações, consulte Reestruturação 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, erros ou avisos gerais gerados durante a validação de política e depois escolha Avançar. Se você ainda encontrar um problema, escolha Anterior para retornar à tela de edição.
5. Escolha Salvar alterações para salvar suas alterações.

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 Noções básicas sobre níveis de acesso em resumos de políticas.

- 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.

```
{
    "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:::developer_bucket",
                "arn:aws:s3:::developer_bucket/*",
                "arn:aws:autoscling:us-east-2:123456789012:autoscalgrp"
            ],
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": [
                        "public-read"
                    ],
                    "s3:prefix": [
                        "custom",
                        "other"
                    ]
                }
            }
        }
    ]
}
```

Noções básicas sobre níveis de acesso em resumos de políticas

Resumo do nível de acesso da AWS

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 Noções básicas sobre as permissões concedidas por uma política. 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 `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.
CodeDeploy	(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”.

Como mencionado anteriormente, 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 `List`, `Read`, `Tagging`, `Write` ou `Permissions management` da AWS.

Níveis de acesso da AWS

A AWS 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`.

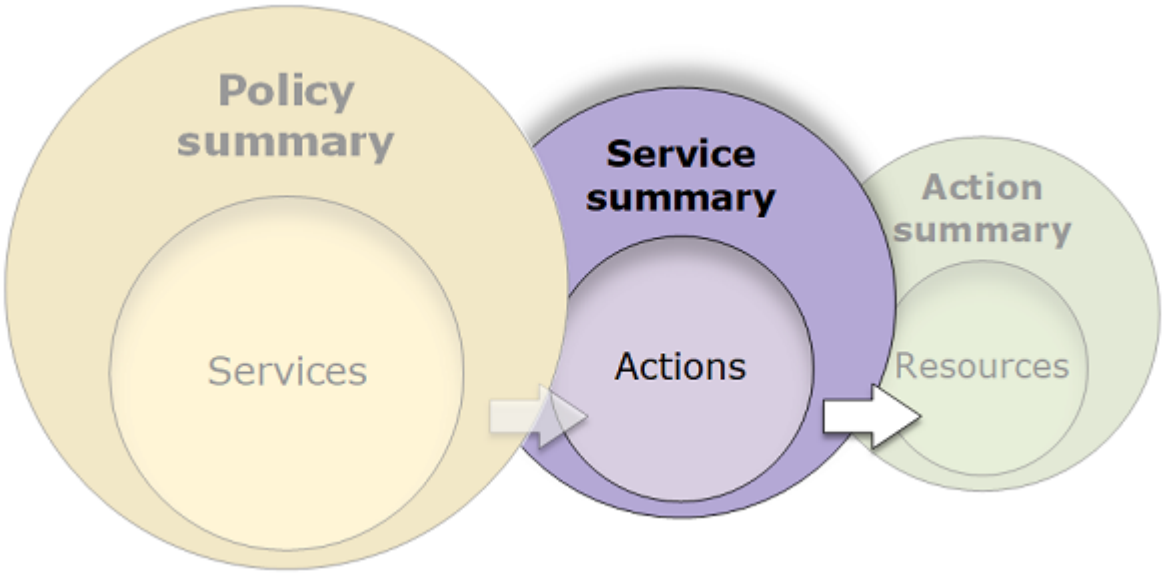
- Gravar: permissão para criar, excluir ou modificar recursos no serviço. Por exemplo, as ações `CreateBucket`, `DeleteBucket` e `PutObject` do Amazon S3 têm o nível de acesso Write (Gravação). As ações de `Write` (gravação) também podem permitir a modificação de uma etiqueta de recurso. No entanto, uma ação que permite apenas alterações nas tags tem o nível de acesso `Tagging`.
- Gerenciamento de permissões: permissão para conceder ou modificar permissões de recursos no serviço. Por exemplo, a maioria das ações do IAM e do AWS Organizations, bem como ações como as ações `PutBucketPolicy` e `DeleteBucketPolicy` do Amazon S3 têm o nível de acesso Permissions management (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 sob uma das seções de nível de acesso (List [Listar], Read [Ler], Write [Gravar] e Permissions management [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.

Visualização do resumo de serviços

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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha 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 o resumo de serviços de uma política anexada a um usuário

1. Faça login no AWS Management Console 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.

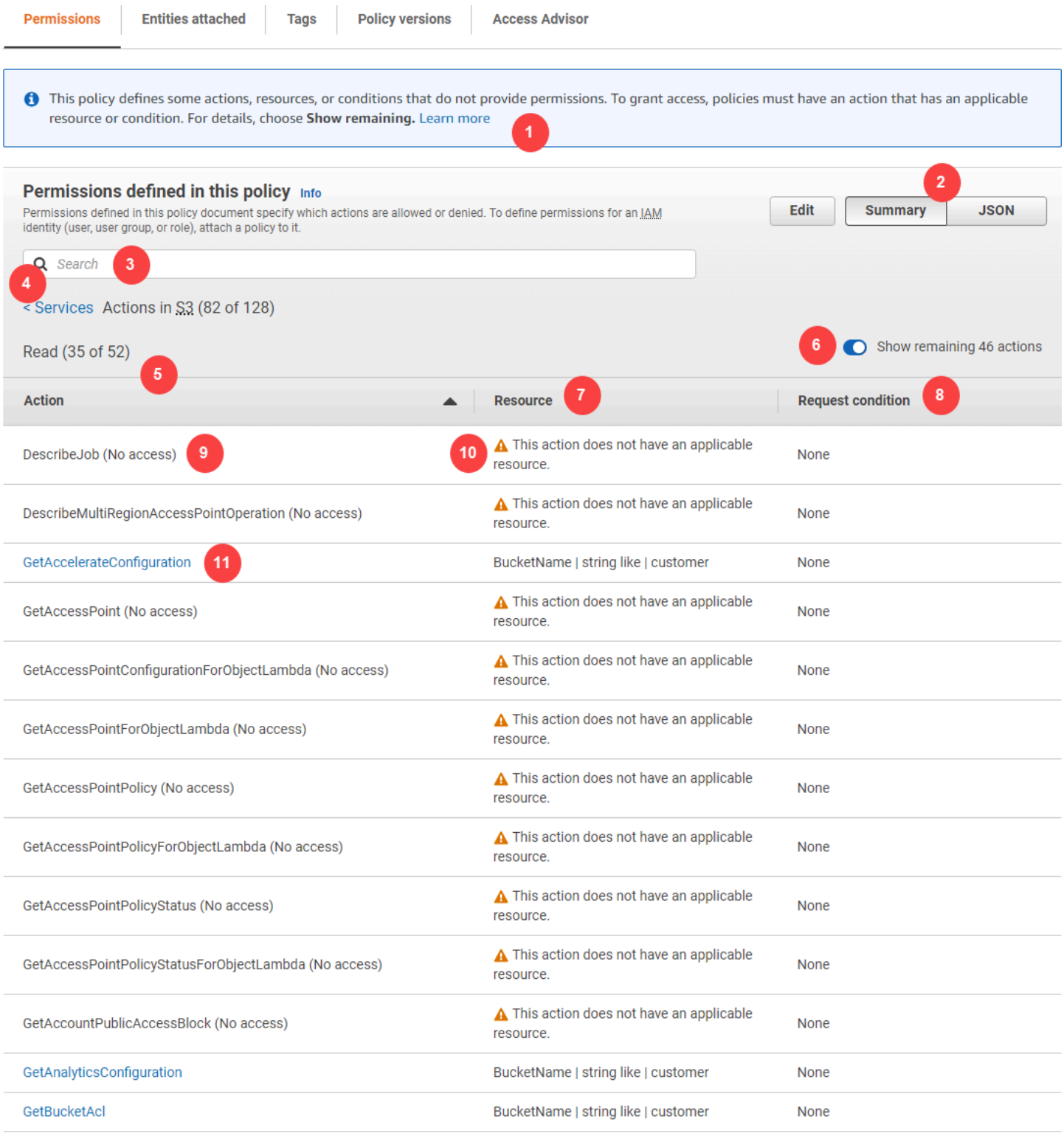
Para visualizar o resumo de serviços de uma política anexada a uma função

1. Faça login no AWS Management Console 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.

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”.

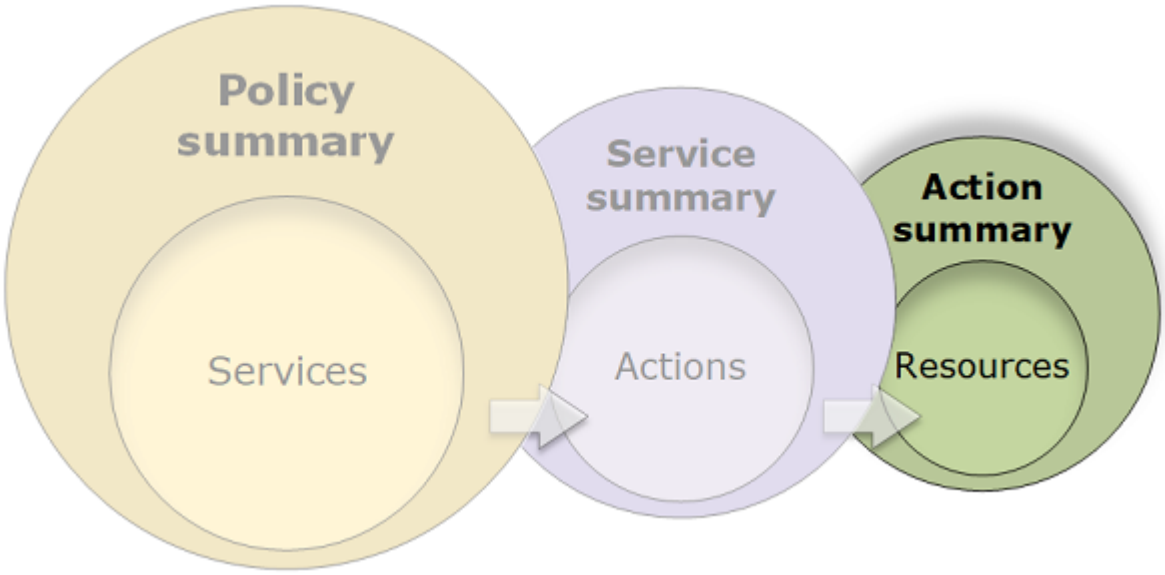
2. Escolha JSON para ver detalhes adicionais sobre a política. Faça isso para visualizar todas as condições que são aplicadas às ações. (Se você estiver visualizando o resumo de serviços para uma política em linha que esteja anexado diretamente a um usuário, será necessário fechar a caixa de diálogo 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.

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.

Visualização de resumos de ação

Você pode visualizar o resumo das ações das políticas gerenciadas, qualquer política anexada a um usuário e qualquer política anexada a um perfil na página Políticas.

Para visualizar o resumo da ação de uma política gerenciada

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha 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.

Para visualizar o resumo da ação de uma política anexada a um usuário

1. Faça login no AWS Management Console 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.

Para visualizar o resumo da ação de uma política anexada a uma função

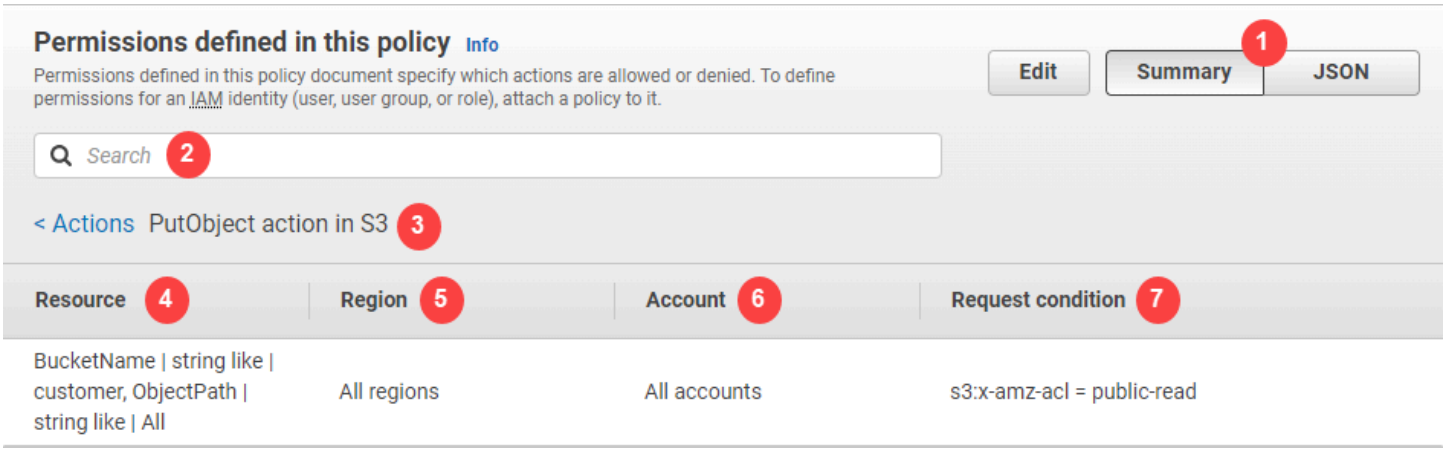
1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis.
3. Na lista de funções, escolha o nome da função cuja política deseja visualizar.
4. Na página Resumo da função, visualize a guia Permissões para ver a lista 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.

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.

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.

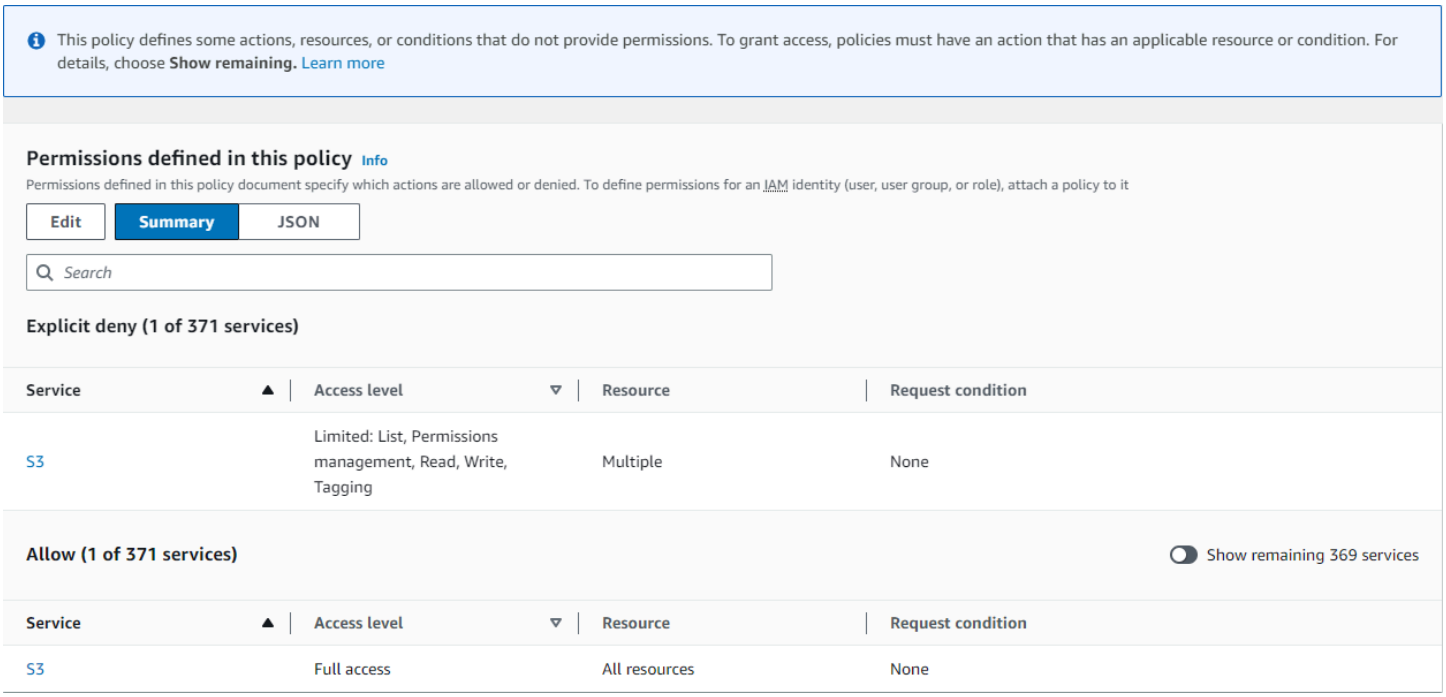
Política 1: DenyCustomerBucket

Essa política demonstra uma permissão e negação para o mesmo serviço.

```
{
    "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.

```
{
```

```
    "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:

Allow (1 of 370 services) — Show remaining 369 services

Service	Access level	Resource	Request condition
DynamoDB	Limited: Read, Write	region| string like |us-west-1, TableName| string like |myDynamoTable	dynamodb:LeadingKeys = ${cognito-identity.amazonaws.com:sub}

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}

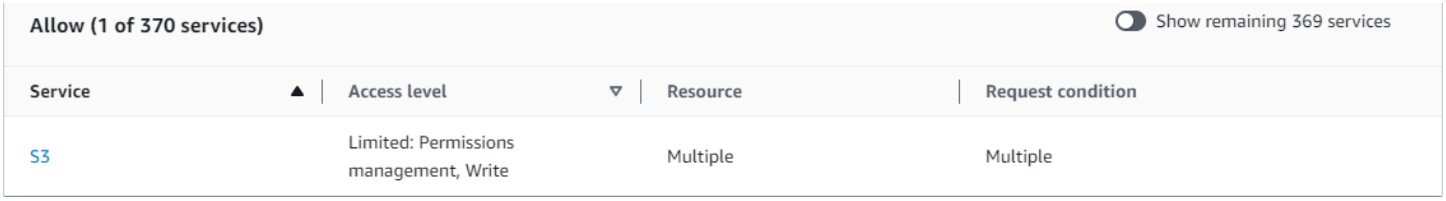
Política 3: MultipleResourceCondition

Essa política inclui vários recursos e condições.

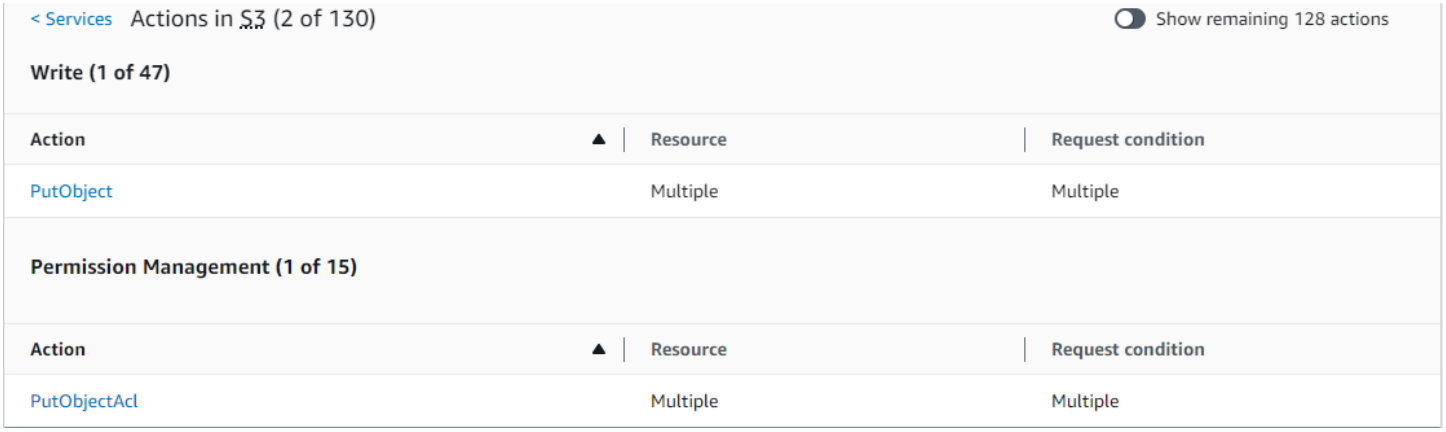
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": ["arn:aws:s3:::Apple_bucket/*"],
            "Condition": {"StringEquals": {"s3:x-amz-acl": ["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"]
            }}
        }
    ]
}
```

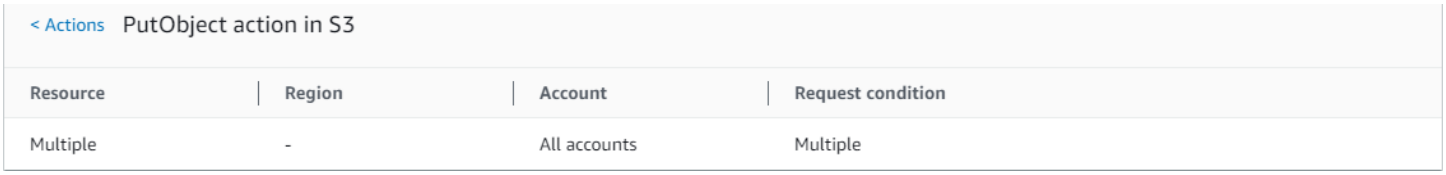
Resumo da política MultipleResourceCondition:



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.

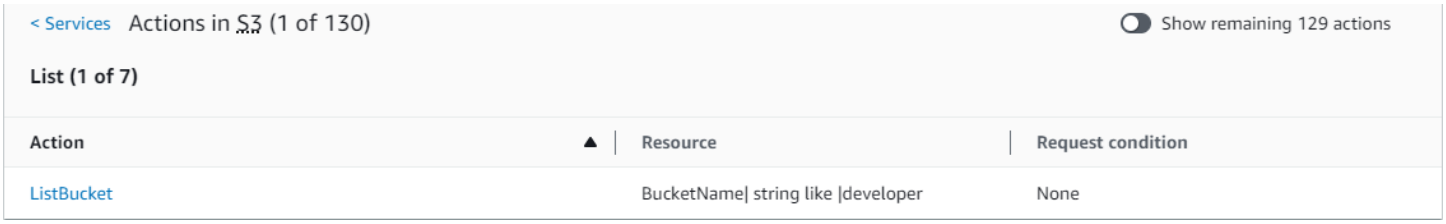
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetConsoleScreenshot"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::developer"
            ]
        }
    ]
}
```

Resumo da política EC2_Troubleshoot:

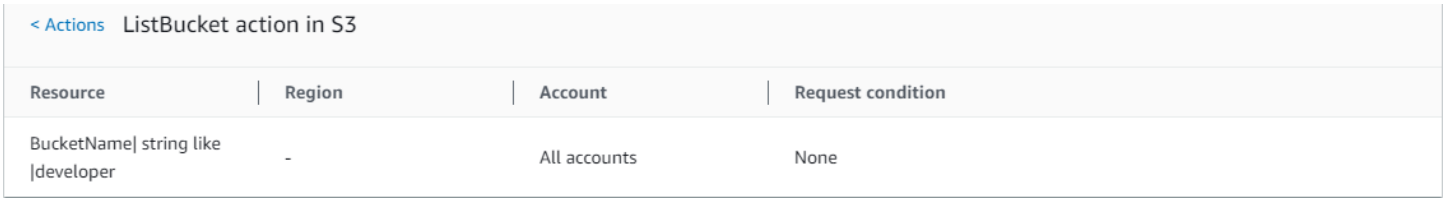
Allow (2 of 370 services) — Show remaining 368 services

Service	Access level	Resource	Request condition
EC2	Limited: Read	All resources	None
S3	Limited: List	BucketName| string like |developer	None

Resumo do serviço EC2_Troubleshoot S3 (Permitir):



Resumo da ação ListBucket (Listar):

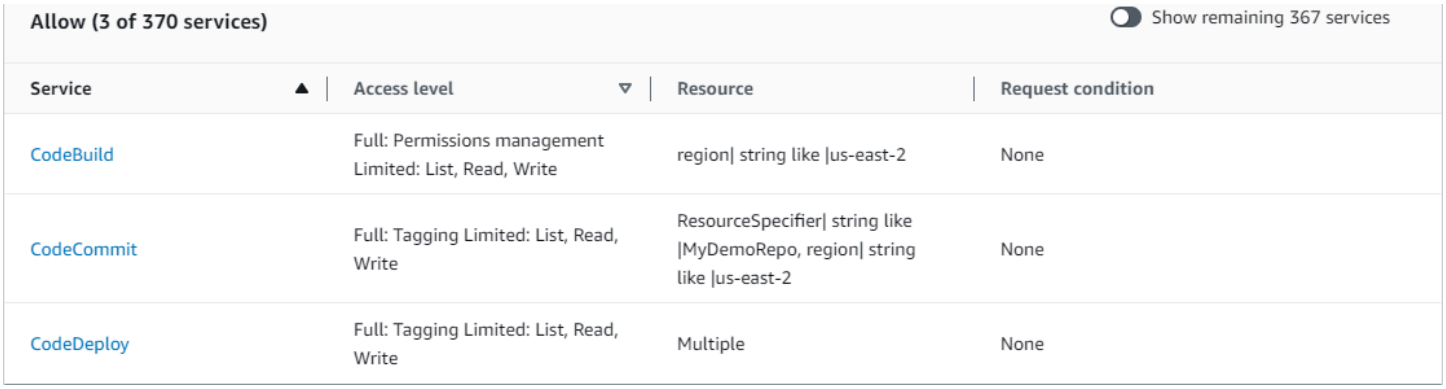


Política 5: CodeBuild_CodeCommit_CodeDeploy

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.

```
{
    "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:

```
{
  "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 de API `CreateAccessKey` e `UpdateAccessKey`) geralmente tem um elemento de `Resource`. Neste caso, o ARN inclui uma

variável (`${aws:username}`) que é substituída pelo nome do usuário atual, como no exemplo a seguir:

```
{
    "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):

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id: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 AWS Management Console

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 AWS Management Console, 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 Fornecer 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 uma função 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 Conceder permissões a um usuário para passar uma função 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 para o 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.

```
{
    "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 AWS Management Console. 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 AWS Management Console:

- `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 credencial, consulte Obter relatórios de credenciais da 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.

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

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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código

- Usar DeactivateMfaDevice com o AWS SDK ou a CLI
- Usar DeleteAccessKey com o AWS SDK ou a CLI
- Usar DeleteAccountAlias com o AWS SDK ou a CLI
- Usar DeleteAccountPasswordPolicy com o AWS SDK ou a CLI
- Usar DeleteGroup com o AWS SDK ou a CLI
- Usar DeleteGroupPolicy com o AWS SDK ou a CLI
- Usar DeleteInstanceProfile com o AWS SDK ou a CLI
- Usar DeleteLoginProfile com o AWS SDK ou a CLI
- Usar DeleteOpenIdConnectProvider com o AWS SDK ou a 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 o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar DeleteUser com o AWS SDK ou a CLI
- Usar DeleteUserPermissionsBoundary com o AWS SDK ou a CLI
- Usar DeleteUserPolicy com o AWS SDK ou a CLI
- Usar DeleteVirtualMfaDevice com o AWS SDK ou a CLI
- Usar DetachGroupPolicy com o AWS SDK ou a CLI
- Usar DetachRolePolicy com o AWS SDK ou a CLI
- Usar DetachUserPolicy com o AWS SDK ou a CLI
- Usar EnableMfaDevice com o AWS SDK ou a CLI
- Usar GenerateCredentialReport com o AWS SDK ou a CLI
- Usar GenerateServiceLastAccessedDetails com o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar GetContextKeysForPrincipalPolicy com o AWS SDK ou a CLI
- Usar GetCredentialReport com o AWS SDK ou a CLI
- Usar GetGroup com o AWS SDK ou a CLI
- Usar GetGroupPolicy com o AWS SDK ou a CLI
- Usar GetInstanceProfile com o AWS SDK ou a CLI
- Usar GetLoginProfile com o AWS SDK ou a CLI
- Usar GetOpenIdConnectProvider com o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar GetSamlProvider com o AWS SDK ou a CLI
- Usar GetServerCertificate com o AWS SDK ou a CLI
- Usar GetServiceLastAccessedDetails com o AWS SDK ou a CLI
- Usar GetServiceLastAccessedDetailsWithEntities com o AWS SDK ou a CLI
- Usar GetServiceLinkedRoleDeletionStatus com o AWS SDK ou a CLI
- Usar GetUser com o AWS SDK ou a CLI
- Usar GetUserPolicy com o AWS SDK ou a CLI
- Usar ListAccessKeys com o AWS SDK ou a CLI
- Usar ListAccountAliases com o AWS SDK ou a CLI
- Usar ListAttachedGroupPolicies com o AWS SDK ou a CLI
- Usar ListAttachedRolePolicies com o AWS SDK ou a CLI
- Usar ListAttachedUserPolicies com o AWS SDK ou a CLI
- Usar ListEntitiesForPolicy com o AWS SDK ou a CLI
- Usar ListGroupPolicies com o AWS SDK ou a CLI
- Usar ListGroups com o AWS SDK ou a CLI
- Usar ListGroupsForUser com o AWS SDK ou a CLI

- Usar ListInstanceProfiles com o AWS SDK ou a CLI
- Usar ListInstanceProfilesForRole com o AWS SDK ou a CLI
- Usar ListMfaDevices com o AWS SDK ou a CLI
- Usar ListOpenIdConnectProviders com o AWS SDK ou a 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 o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar ListUserPolicies com o AWS SDK ou a CLI
- Usar ListUserTags com o AWS SDK ou a CLI
- Usar ListUsers com o AWS SDK ou a CLI
- Usar ListVirtualMfaDevices com o AWS SDK ou a CLI
- Usar PutGroupPolicy com o AWS SDK ou a CLI
- Usar PutRolePermissionsBoundary com o AWS SDK ou a CLI
- Usar PutRolePolicy com o AWS SDK ou a CLI
- Usar PutUserPermissionsBoundary com o AWS SDK ou a CLI
- Usar PutUserPolicy com o AWS SDK ou a CLI
- Usar RemoveClientIdFromOpenIdConnectProvider com o AWS SDK ou a CLI
- Usar RemoveRoleFromInstanceProfile com o AWS SDK ou a CLI
- Usar RemoveUserFromGroup com o AWS SDK ou a CLI
- Usar ResyncMfaDevice com o AWS SDK ou a CLI
- Usar SetDefaultPolicyVersion com o AWS SDK ou a CLI
- Usar TagRole com o AWS SDK ou a CLI
- Usar TagUser com o AWS SDK ou a CLI
- Usar UntagRole com o AWS SDK ou a CLI
- Usar UntagUser com o AWS SDK ou a CLI

- Usar UpdateAccessKey com o AWS SDK ou a CLI
- Usar UpdateAccountPasswordPolicy com o AWS SDK ou a CLI
- Usar UpdateAssumeRolePolicy com o AWS SDK ou a CLI
- Usar UpdateGroup com o AWS SDK ou a CLI
- Usar UpdateLoginProfile com o AWS SDK ou a CLI
- Usar UpdateOpenIdConnectProviderThumbprint com o AWS SDK ou a CLI
- Usar UpdateRole com o AWS SDK ou a CLI
- Usar UpdateRoleDescription com o AWS SDK ou a CLI
- Usar UpdateSamlProvider com o AWS SDK ou a CLI
- Usar UpdateServerCertificate com o AWS SDK ou a CLI
- Usar UpdateSigningCertificate com o AWS SDK ou a CLI
- Usar UpdateUser com o AWS SDK ou a CLI
- Usar UploadServerCertificate com o AWS SDK ou a CLI
- Usar UploadSigningCertificate com o AWS SDK ou a CLI

- Cenários para o IAM usando AWS SDKs
  - Criar e gerenciar um serviço resiliente usando um AWS SDK
  - Criar um grupo do IAM e adicionar um usuário ao grupo usando um AWS SDK
  - Criar um usuário do IAM e assumir uma função com o AWS STS usando um AWS SDK
  - Criar usuários do IAM somente leitura e leitura/gravação usando um AWS SDK
  - 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
  - Reverter uma versão de política do IAM usando um AWS SDK
  - Trabalhar com a API IAM Policy Builder usando um AWS SDK

- Exemplos de código para o AWS STS usando AWS SDKs
  - Ações para o AWS STS usando AWS SDKs
    - Usar AssumeRole com o AWS SDK ou a CLI
    - Usar AssumeRoleWithWebIdentity com o AWS SDK ou a CLI
    - Usar DecodeAuthorizationMessage com o AWS SDK ou a CLI

    - Usar GetFederationToken com o AWS SDK ou a 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
    - Crie uma URL com o AWS STS para usuários federados usando um AWS SDK
    - Obtenha um token de sessão que exija um token de MFA com o AWS STS usando um AWS SDK

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.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Conceitos básicos

Olá, IAM

O exemplo de código a seguir mostra como começar a usar o IAM.

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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}");
        });
    }
}
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK for .NET.

C++

SDK para C++

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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;
}
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK for C++.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 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(context.TODO(), &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)
  }
 }
}
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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());
        });
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
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) {
      page.Policies.forEach((p) => {
        console.log(`${p.PolicyName}`);
        policyCount++;
      });
    }
  }
  console.log(`Found ${policyCount} policies.`);
};
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK for JavaScript.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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<(), 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)
}
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK for Rust.

Exemplos de código

- Ações do IAM usando AWS SDKs
  - Usar AddClientIdToOpenIdConnectProvider com o AWS SDK ou a CLI
  - Usar AddRoleToInstanceProfile com o AWS SDK ou a CLI
  - Usar AddUserToGroup com o AWS SDK ou a CLI
  - Usar AttachGroupPolicy com o AWS SDK ou a CLI
  - Usar AttachRolePolicy com o AWS SDK ou a CLI
  - Usar AttachUserPolicy com o AWS SDK ou a CLI
  - Usar ChangePassword com o AWS SDK ou a 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 o AWS SDK ou a CLI
  - Usar CreateOpenIdConnectProvider com o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar DeactivateMfaDevice com o AWS SDK ou a CLI
- Usar DeleteAccessKey com o AWS SDK ou a CLI
- Usar DeleteAccountAlias com o AWS SDK ou a CLI
- Usar DeleteAccountPasswordPolicy com o AWS SDK ou a CLI
- Usar DeleteGroup com o AWS SDK ou a CLI
- Usar DeleteGroupPolicy com o AWS SDK ou a CLI
- Usar DeleteInstanceProfile com o AWS SDK ou a CLI
- Usar DeleteLoginProfile com o AWS SDK ou a CLI
- Usar DeleteOpenIdConnectProvider com o AWS SDK ou a 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 o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar DeleteUser com o AWS SDK ou a CLI
- Usar DeleteUserPermissionsBoundary com o AWS SDK ou a CLI
- Usar DeleteUserPolicy com o AWS SDK ou a CLI
- Usar DeleteVirtualMfaDevice com o AWS SDK ou a CLI
- Usar DetachGroupPolicy com o AWS SDK ou a CLI
- Usar DetachRolePolicy com o AWS SDK ou a CLI
- Usar DetachUserPolicy com o AWS SDK ou a CLI

- Usar EnableMfaDevice com o AWS SDK ou a CLI
- Usar GenerateCredentialReport com o AWS SDK ou a CLI
- Usar GenerateServiceLastAccessedDetails com o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar GetContextKeysForPrincipalPolicy com o AWS SDK ou a CLI
- Usar GetCredentialReport com o AWS SDK ou a CLI
- Usar GetGroup com o AWS SDK ou a CLI
- Usar GetGroupPolicy com o AWS SDK ou a CLI
- Usar GetInstanceProfile com o AWS SDK ou a CLI
- Usar GetLoginProfile com o AWS SDK ou a CLI
- Usar GetOpenIdConnectProvider com o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar GetSamlProvider com o AWS SDK ou a CLI
- Usar GetServerCertificate com o AWS SDK ou a CLI
- Usar GetServiceLastAccessedDetails com o AWS SDK ou a CLI
- Usar GetServiceLastAccessedDetailsWithEntities com o AWS SDK ou a CLI
- Usar GetServiceLinkedRoleDeletionStatus com o AWS SDK ou a CLI
- Usar GetUser com o AWS SDK ou a CLI
- Usar GetUserPolicy com o AWS SDK ou a CLI
- Usar ListAccessKeys com o AWS SDK ou a CLI
- Usar ListAccountAliases com o AWS SDK ou a CLI
- Usar ListAttachedGroupPolicies com o AWS SDK ou a CLI
- Usar ListAttachedRolePolicies com o AWS SDK ou a CLI

- Usar ListAttachedUserPolicies com o AWS SDK ou a CLI
- Usar ListEntitiesForPolicy com o AWS SDK ou a CLI
- Usar ListGroupPolicies com o AWS SDK ou a CLI
- Usar ListGroups com o AWS SDK ou a CLI
- Usar ListGroupsForUser com o AWS SDK ou a CLI
- Usar ListInstanceProfiles com o AWS SDK ou a CLI
- Usar ListInstanceProfilesForRole com o AWS SDK ou a CLI
- Usar ListMfaDevices com o AWS SDK ou a CLI
- Usar ListOpenIdConnectProviders com o AWS SDK ou a 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 o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar ListUserPolicies com o AWS SDK ou a CLI
- Usar ListUserTags com o AWS SDK ou a CLI
- Usar ListUsers com o AWS SDK ou a CLI
- Usar ListVirtualMfaDevices com o AWS SDK ou a CLI
- Usar PutGroupPolicy com o AWS SDK ou a CLI
- Usar PutRolePermissionsBoundary com o AWS SDK ou a CLI
- Usar PutRolePolicy com o AWS SDK ou a CLI
- Usar PutUserPermissionsBoundary com o AWS SDK ou a CLI
- Usar PutUserPolicy com o AWS SDK ou a CLI
- Usar RemoveClientIdFromOpenIdConnectProvider com o AWS SDK ou a CLI
- Usar RemoveRoleFromInstanceProfile com o AWS SDK ou a CLI
- Usar RemoveUserFromGroup com o AWS SDK ou a CLI
- Usar ResyncMfaDevice com o AWS SDK ou a CLI

- Usar SetDefaultPolicyVersion com o AWS SDK ou a CLI
- Usar TagRole com o AWS SDK ou a CLI
- Usar TagUser com o AWS SDK ou a CLI
- Usar UntagRole com o AWS SDK ou a CLI
- Usar UntagUser com o AWS SDK ou a CLI
- Usar UpdateAccessKey com o AWS SDK ou a CLI
- Usar UpdateAccountPasswordPolicy com o AWS SDK ou a CLI
- Usar UpdateAssumeRolePolicy com o AWS SDK ou a CLI
- Usar UpdateGroup com o AWS SDK ou a CLI
- Usar UpdateLoginProfile com o AWS SDK ou a CLI
- Usar UpdateOpenIdConnectProviderThumbprint com o AWS SDK ou a CLI
- Usar UpdateRole com o AWS SDK ou a CLI
- Usar UpdateRoleDescription com o AWS SDK ou a CLI
- Usar UpdateSamlProvider com o AWS SDK ou a CLI
- Usar UpdateServerCertificate com o AWS SDK ou a CLI
- Usar UpdateSigningCertificate com o AWS SDK ou a CLI
- Usar UpdateUser com o AWS SDK ou a CLI
- Usar UploadServerCertificate com o AWS SDK ou a CLI
- Usar UploadSigningCertificate com o AWS SDK ou a CLI

- Cenários para o IAM usando AWS SDKs
  - Criar e gerenciar um serviço resiliente usando um AWS SDK
  - Criar um grupo do IAM e adicionar um usuário ao grupo usando um AWS SDK
  - Criar um usuário do IAM e assumir uma função com o AWS STS usando um AWS SDK
  - Criar usuários do IAM somente leitura e leitura/gravação usando um AWS SDK
  - 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
  - Reverter uma versão de política do IAM usando um AWS SDK
  - Trabalhar com a API IAM Policy Builder usando um AWS SDK

Ações do IAM usando AWS SDKs

Os exemplos de código a seguir demonstram como realizar ações individuais do IAM com AWS SDKs. Esses trechos chamam a API do IAM e são trechos de código de programas maiores que devem ser executados no contexto. Cada exemplo inclui um link para o GitHub, onde você pode encontrar instruções para configurar e executar o código.

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 (IAM).

Exemplos

- Usar AddClientIdToOpenIdConnectProvider com o AWS SDK ou a CLI
- Usar AddRoleToInstanceProfile com o AWS SDK ou a CLI
- Usar AddUserToGroup com o AWS SDK ou a CLI
- Usar AttachGroupPolicy com o AWS SDK ou a CLI
- Usar AttachRolePolicy com o AWS SDK ou a CLI
- Usar AttachUserPolicy com o AWS SDK ou a CLI
- Usar ChangePassword com o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar CreateOpenIdConnectProvider com o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar DeactivateMfaDevice com o AWS SDK ou a CLI

- Usar DeleteAccessKey com o AWS SDK ou a CLI
- Usar DeleteAccountAlias com o AWS SDK ou a CLI
- Usar DeleteAccountPasswordPolicy com o AWS SDK ou a CLI
- Usar DeleteGroup com o AWS SDK ou a CLI
- Usar DeleteGroupPolicy com o AWS SDK ou a CLI
- Usar DeleteInstanceProfile com o AWS SDK ou a CLI
- Usar DeleteLoginProfile com o AWS SDK ou a CLI
- Usar DeleteOpenIdConnectProvider com o AWS SDK ou a 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 o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar DeleteUser com o AWS SDK ou a CLI
- Usar DeleteUserPermissionsBoundary com o AWS SDK ou a CLI
- Usar DeleteUserPolicy com o AWS SDK ou a CLI
- Usar DeleteVirtualMfaDevice com o AWS SDK ou a CLI
- Usar DetachGroupPolicy com o AWS SDK ou a CLI
- Usar DetachRolePolicy com o AWS SDK ou a CLI
- Usar DetachUserPolicy com o AWS SDK ou a CLI
- Usar EnableMfaDevice com o AWS SDK ou a CLI
- Usar GenerateCredentialReport com o AWS SDK ou a CLI
- Usar GenerateServiceLastAccessedDetails com o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar GetContextKeysForPrincipalPolicy com o AWS SDK ou a CLI
- Usar GetCredentialReport com o AWS SDK ou a CLI
- Usar GetGroup com o AWS SDK ou a CLI
- Usar GetGroupPolicy com o AWS SDK ou a CLI
- Usar GetInstanceProfile com o AWS SDK ou a CLI
- Usar GetLoginProfile com o AWS SDK ou a CLI
- Usar GetOpenIdConnectProvider com o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar GetSamlProvider com o AWS SDK ou a CLI
- Usar GetServerCertificate com o AWS SDK ou a CLI
- Usar GetServiceLastAccessedDetails com o AWS SDK ou a CLI
- Usar GetServiceLastAccessedDetailsWithEntities com o AWS SDK ou a CLI
- Usar GetServiceLinkedRoleDeletionStatus com o AWS SDK ou a CLI
- Usar GetUser com o AWS SDK ou a CLI
- Usar GetUserPolicy com o AWS SDK ou a CLI
- Usar ListAccessKeys com o AWS SDK ou a CLI
- Usar ListAccountAliases com o AWS SDK ou a CLI
- Usar ListAttachedGroupPolicies com o AWS SDK ou a CLI
- Usar ListAttachedRolePolicies com o AWS SDK ou a CLI
- Usar ListAttachedUserPolicies com o AWS SDK ou a CLI
- Usar ListEntitiesForPolicy com o AWS SDK ou a CLI
- Usar ListGroupPolicies com o AWS SDK ou a CLI
- Usar ListGroups com o AWS SDK ou a CLI
- Usar ListGroupsForUser com o AWS SDK ou a CLI

- Usar ListInstanceProfiles com o AWS SDK ou a CLI
- Usar ListInstanceProfilesForRole com o AWS SDK ou a CLI
- Usar ListMfaDevices com o AWS SDK ou a CLI
- Usar ListOpenIdConnectProviders com o AWS SDK ou a 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 o AWS SDK ou a 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 o AWS SDK ou a CLI
- Usar ListUserPolicies com o AWS SDK ou a CLI
- Usar ListUserTags com o AWS SDK ou a CLI
- Usar ListUsers com o AWS SDK ou a CLI
- Usar ListVirtualMfaDevices com o AWS SDK ou a CLI
- Usar PutGroupPolicy com o AWS SDK ou a CLI
- Usar PutRolePermissionsBoundary com o AWS SDK ou a CLI
- Usar PutRolePolicy com o AWS SDK ou a CLI
- Usar PutUserPermissionsBoundary com o AWS SDK ou a CLI
- Usar PutUserPolicy com o AWS SDK ou a CLI
- Usar RemoveClientIdFromOpenIdConnectProvider com o AWS SDK ou a CLI
- Usar RemoveRoleFromInstanceProfile com o AWS SDK ou a CLI
- Usar RemoveUserFromGroup com o AWS SDK ou a CLI
- Usar ResyncMfaDevice com o AWS SDK ou a CLI
- Usar SetDefaultPolicyVersion com o AWS SDK ou a CLI
- Usar TagRole com o AWS SDK ou a CLI
- Usar TagUser com o AWS SDK ou a CLI
- Usar UntagRole com o AWS SDK ou a CLI
- Usar UntagUser com o AWS SDK ou a CLI

- Usar UpdateAccessKey com o AWS SDK ou a CLI
- Usar UpdateAccountPasswordPolicy com o AWS SDK ou a CLI
- Usar UpdateAssumeRolePolicy com o AWS SDK ou a CLI
- Usar UpdateGroup com o AWS SDK ou a CLI
- Usar UpdateLoginProfile com o AWS SDK ou a CLI
- Usar UpdateOpenIdConnectProviderThumbprint com o AWS SDK ou a CLI
- Usar UpdateRole com o AWS SDK ou a CLI
- Usar UpdateRoleDescription com o AWS SDK ou a CLI
- Usar UpdateSamlProvider com o AWS SDK ou a CLI
- Usar UpdateServerCertificate com o AWS SDK ou a CLI
- Usar UpdateSigningCertificate com o AWS SDK ou a CLI
- Usar UpdateUser com o AWS SDK ou a CLI
- Usar UploadServerCertificate com o AWS SDK ou a CLI
- Usar UploadSigningCertificate com o AWS SDK ou a CLI

Usar **`AddClientIdToOpenIdConnectProvider`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte AddClientIdToOpenIdConnectProvider na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte AddClientIdToOpenIdConnectProvider na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`AddRoleToInstanceProfile`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `AddRoleToInstanceProfile`.

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.

- Para obter detalhes da API, consulte AddRoleToInstanceProfile na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte AddRoleToInstanceProfile na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`AddUserToGroup`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `AddUserToGroup`.

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 um grupo e adicionar um usuário

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <summary>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Para obter detalhes da API, consulte AddUserToGroup na Referência da API AWS SDK for .NET.

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.

- Para obter detalhes da API, consulte AddUserToGroup na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

Exemplo 1: este comando adiciona o usuário chamado **`Bob`** ao grupo denominado **`Admins`**.

```
Add-IAMUserToGroup -UserName "Bob" -GroupName "Admins"
```

- Para obter detalhes da API, consulte AddUserToGroup na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`AttachGroupPolicy`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte AttachGroupPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte AttachGroupPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um grupo e adicionar um usuário
- Criar um usuário e assumir uma função
- Gerenciar funções

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte AttachRolePolicy na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
##############################################################################
# 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
}
```

- Para obter detalhes da API, consulte AttachRolePolicy 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::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();
}
```

- Para obter detalhes da API, consulte AttachRolePolicy na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte AttachRolePolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte AttachRolePolicy na Referência da API AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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");
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte AttachRolePolicy na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte AttachRolePolicy na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun attachIAMRolePolicy(roleNameVal: String, policyArnVal: String) {

    val request = ListAttachedRolePoliciesRequest {
        roleName = roleNameVal
    }

    IamClient { 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
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
$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,
            ]);
        });
    }
```

- Para obter detalhes da API, consulte AttachRolePolicy na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte AttachRolePolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 AttachRolePolicy na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
}
```

- Para obter detalhes da API, consulte AttachRolePolicy na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
public func attachRolePolicy(role: String, policyArn: String) async throws {
    let input = AttachRolePolicyInput(
        policyArn: policyArn,
        roleName: role
    )
    do {
        _ = try await client.attachRolePolicy(input: input)
    } catch {
        throw error
    }
}
```

- Para obter detalhes da API, consulte AttachRolePolicy na referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no 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.

- Para obter detalhes da API, consulte AttachUserPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte AttachUserPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
  # 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
```

- Para obter detalhes da API, consulte AttachUserPolicy na Referência da API do AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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(())
}
```

- Para obter detalhes da API, consulte AttachUserPolicy na Referência da API AWS SDK for Rust.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ChangePassword`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte ChangePassword na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ChangePassword na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um grupo e adicionar um usuário
- Criar um usuário e assumir uma função
- Criar usuários somente leitura e leitura/gravação usando
- Gerenciar chaves de acesso

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;

    }
```

- Para obter detalhes da API, consulte CreateAccessKey na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
###############################################################################
# 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
}
```

- Para obter detalhes da API, consulte CreateAccessKey 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.

```
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;
}
```

- Para obter detalhes da API, consulte CreateAccessKey na Referência da API AWS SDK for 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte CreateAccessKey na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte CreateAccessKey na Referência da API AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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 "";
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte CreateAccessKey na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte CreateAccessKey na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun createIAMAccessKey(user: String?): String {

    val request = CreateAccessKeyRequest {
        userName = user
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createAccessKey(request)
        return response.accessKey?.accessKeyId.toString()
    }
}
```

- Para obter detalhes da API, consulte CreateAccessKey na Referência da API AWS SDK for Kotlin.

PowerShell

Tools for PowerShell

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 da API, consulte CreateAccessKey na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 => e
    @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 => e
    @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
```

- Para obter detalhes da API, consulte CreateAccessKey na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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())
}
```

- Para obter detalhes da API, consulte CreateAccessKey na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
   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 {
            throw error
        }
    }
```

- Para obter detalhes da API, consulte CreateAccessKey na Referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no contexto no seguinte exemplo de código:

- Gerenciar sua conta

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::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();
}
```

- Para obter detalhes da API, consulte CreateAccountAlias na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte CreateAccountAlias na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte CreateAccountAlias na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte CreateAccountAlias na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun createIAMAccountAlias(alias: String) {

    val request = CreateAccountAliasRequest {
        accountAlias = alias
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.createAccountAlias(request)
        println("Successfully created account alias named $alias")
    }
}
```

- Para obter detalhes da API, consulte CreateAccountAlias na Referência da API AWS SDK for Kotlin.

PowerShell

Tools for PowerShell

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 da API, consulte CreateAccountAlias na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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
```

- Para obter detalhes da API, consulte CreateAccountAlias na Referência da API AWS SDK for Ruby.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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`.

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 um grupo e adicionar um usuário

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <summary>
    /// Create an IAM group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
{ GroupName = groupName });
        return response.Group;
    }
```

- Para obter detalhes da API, consulte CreateGroup na Referência da API AWS SDK for .NET.

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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
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;
};
```

- Para obter detalhes da API, consulte CreateGroup na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte CreateGroup na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no contexto no seguinte exemplo de código:

- Criar e gerenciar um serviço resiliente

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
   /// <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;
    }
```

- Para obter detalhes da API, consulte CreateInstanceProfile na Referência de API do AWS SDK for .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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
const { InstanceProfile } = await iamClient.send(
  new CreateInstanceProfileCommand({
    InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
  }),
);
await waitUntilInstanceProfileExists(
  { client: iamClient },
  { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
);
```

- Para obter detalhes da API, consulte CreateInstanceProfile na Referência de API do AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte CreateInstanceProfile na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

Este exemplo cria uma política, um perfil e um perfil de instância e vincula uns aos outros.

```
class AutoScaler:
    """
```

```
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :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
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        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"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def create_instance_profile(
```

```
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        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",
                }
            ],
        }
        with open(policy_file) as file:
            instance_policy_doc = file.read()

        policy_arn = None
        try:
            pol_response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=instance_policy_doc
            )
```

```
            policy_arn = pol_response["Policy"]["Arn"]
            log.info("Created policy with ARN %s.", policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Policy %s already exists, nothing to do.", policy_name)
                list_pol_response = self.iam_client.list_policies(Scope="Local")
                for pol in list_pol_response["Policies"]:
                    if pol["PolicyName"] == policy_name:
                        policy_arn = pol["Arn"]
                        break
            if policy_arn is None:
                raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")

        try:
            self.iam_client.create_role(
                RoleName=role_name,
AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            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("Created role %s and attached policy %s.", role_name,
policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Role %s already exists, nothing to do.", role_name)
            else:
                raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

        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
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create profile {profile_name} and attach it to
role\n"
                    f"{role_name}: {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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateLoginProfile`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte CreateLoginProfile na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte CreateLoginProfile na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateOpenIdConnectProvider`** com o AWS SDK ou a 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"
    ]
}
```

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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte CreateOpenIdConnectProvider na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte CreateOpenIdConnectProvider na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um grupo e adicionar um usuário
- Criar um usuário e assumir uma função
- Criar usuários somente leitura e leitura/gravação usando
- Políticas gerenciadas
- Trabalhar com a API IAM Policy Builder

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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 for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
###############################################################################
# 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"
}
```

- Para obter detalhes da API, consulte CreatePolicy 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.

```
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 for 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`.

```
aws iam create-policy \
    --policy-name my-policy \
    --policy-document file://policy
```

O arquivo `policy` é um documento JSON na pasta atual que concede acesso somente leitura à pasta `shared` em um bucket do Amazon S3 denominado `my-bucket`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket/shared/*"
            ]
        }
    ]
}
```

Saída:

```
{
    "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 seguinte comando cria uma política gerenciada pelo cliente denominada `my-policy` com uma descrição imutável:

```
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 my-bucket"
```

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 `my-bucket`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::my-bucket"
             ]
```

```
        }
    ]
}
```

Saída:

```
{
    "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: como criar uma política gerenciada pelo cliente com etiquetas

O comando apresentado a seguir cria uma política gerenciada pelo cliente, denominada `my-policy`, 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-policy \
    --policy-name my-policy \
    --policy-document file://policy.json \
    --tags '{"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 `my-bucket`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                 "s3:ListBucket*",
                 "s3:PutBucket*",
                 "s3:GetBucket*"
             ],
             "Resource": [
                 "arn:aws:s3:::my-bucket"
             ]
         }
     ]
 }
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte CreatePolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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
}



// 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(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(context.TODO(),
 &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 for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes, consulte CreatePolicy na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes, consulte CreatePolicy na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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\": \"*\"" +
        "    }" +
        "   ]" +
        "}"

    val request = CreatePolicyRequest {
        policyName = policyNameVal
        policyDocument = policyDocumentVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createPolicy(request)
        return response.policy?.arn.toString()
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
$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";

    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 for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte CreatePolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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())
}
```

- Para obter detalhes da API, consulte CreatePolicy na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    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 {
            throw error
        }
    }
```

- Para obter detalhes da API, consulte CreatePolicy na Referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte CreatePolicyVersion na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte CreatePolicyVersion na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte CreatePolicyVersion na Referência da API AWS SDK para Python (Boto3).

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um grupo e adicionar um usuário
- Criar um usuário e assumir uma função
- Gerenciar funções

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte CreateRole na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
###############################################################################
# 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
}
```

- Para obter detalhes da API, consulte CreateRole 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::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();
}
```

- Para obter detalhes da API, consulte CreateRole na Referência da API AWS SDK for 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
```

Saída:

```
{
    "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
```

Saída:

```
{
    "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"}'
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte CreateRole na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(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(context.TODO(),
  &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
}
```

- Para obter detalhes da API, consulte CreateRole na Referência da API AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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);
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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);
};
```

- Para obter detalhes da API, consulte CreateRole na Referência da API AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
$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'];
    }
```

- Para obter detalhes da API, consulte CreateRole na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte CreateRole na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte CreateRole na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
# 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
```

- Para obter detalhes da API, consulte CreateRole na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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())
}
```

- Para obter detalhes da API, consulte CreateRole na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
   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 {
            throw error
        }
    }
```

- Para obter detalhes da API, consulte CreateRole na referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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
```

Saída:

```
{
```

```
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
import { CreateSAMLProviderCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "fs";
import * as path from "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;
};
```

- Para obter detalhes da API, consulte CreateSAMLProvider na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte CreateSAMLProvider na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte CreateServiceLinkedRole na Referência da API AWS SDK for .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"
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte CreateServiceLinkedRole na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte CreateServiceLinkedRole na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

Criar uma função vinculada ao serviço.

```
import { CreateServiceLinkedRoleCommand, 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,
  });

  const response = await client.send(command);
  console.log(response);
```

```
  return response;
};
```

- Para obter detalhes da API, consulte CreateServiceLinkedRole na Referência da API AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
$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);
    }
```

- Para obter detalhes da API, consulte CreateServiceLinkedRole na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte CreateServiceLinkedRole na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte CreateServiceLinkedRole na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
 # 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
```

- Para obter detalhes da API, consulte CreateServiceLinkedRole na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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)
}
```

- Para obter detalhes da API, consulte CreateServiceLinkedRole na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
   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 {
           throw error
       }
   }
```

- Para obter detalhes da API, consulte CreateServiceLinkedRole na referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um grupo e adicionar um usuário
- Criar um usuário e assumir uma função
- Criar usuários somente leitura e leitura/gravação usando

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte CreateUser na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
##############################################################################
# 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 WS 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
}
```

- Para obter detalhes da API, consulte CreateUser 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.

```
    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();
```

- Para obter detalhes da API, consulte CreateUser na Referência da API AWS SDK for 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
```

Saída:

```
{
    "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/
```

Saída:

```
{
    "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"}'
```

Saída:

```
{
    "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 4: 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte CreateUser na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte CreateUser na Referência da API AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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 "";
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte CreateUser na Referência da API AWS SDK for 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 for JavaScript.

- Para obter detalhes da API, consulte CreateUser na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun createIAMUser(usernameVal: String?): String? {

    val request = CreateUserRequest {
        userName = usernameVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.createUser(request)
        return response.user?.userName
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
$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'];
    }
```

- Para obter detalhes da API, consulte CreateUser na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte CreateUser na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte CreateUser na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
 # 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
```

- Para obter detalhes da API, consulte CreateUser na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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())
}
```

- Para obter detalhes da API, consulte CreateUser na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    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 {
            throw error
        }
    }
```

- Para obter detalhes da API, consulte CreateUser na referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`CreateVirtualMfaDevice`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte CreateVirtualMfaDevice na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte CreateVirtualMfaDevice na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeactivateMfaDevice`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte DeactivateMfaDevice na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DeactivateMfaDevice na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um grupo e adicionar um usuário
- Criar um usuário e assumir uma função
- Criar usuários somente leitura e leitura/gravação usando
- Gerenciar chaves de acesso

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
   }
```

- Para obter detalhes da API, consulte DeleteAccessKey na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
##############################################################################
# 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 WS 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
}
```

- Para obter detalhes da API, consulte DeleteAccessKey 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::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();
}
```

- Para obter detalhes da API, consulte DeleteAccessKey na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte DeleteAccessKey na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte DeleteAccessKey na Referência da API AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte DeleteAccessKey na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte DeleteAccessKey na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun deleteKey(userNameVal: String, accessKey: String) {

    val request = DeleteAccessKeyRequest {
        accessKeyId = accessKey
        userName = userNameVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccessKey(request)
        println("Successfully deleted access key $accessKey from $userNameVal")
    }
}
```

- Para obter detalhes da API, consulte DeleteAccessKey na Referência da API AWS SDK for Kotlin.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteAccessKey na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 => e
    @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 => e
    @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
```

- Para obter detalhes da API, consulte DeleteAccessKey na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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(())
}
```

- Para obter detalhes da API, consulte DeleteAccessKey na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    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 {
            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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no contexto no seguinte exemplo de código:

- Gerenciar sua conta

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::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();
}
```

- Para obter detalhes da API, consulte DeleteAccountAlias na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte DeleteAccountAlias na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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");
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte DeleteAccountAlias na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte DeleteAccountAlias na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun deleteIAMAccountAlias(alias: String) {

    val request = DeleteAccountAliasRequest {
        accountAlias = alias
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteAccountAlias(request)
        println("Successfully deleted account alias $alias")
    }
```

```
}
```

- Para obter detalhes da API, consulte DeleteAccountAlias na Referência da API AWS SDK for Kotlin.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteAccountAlias na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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
```

- Para obter detalhes da API, consulte DeleteAccountAlias na Referência da API AWS SDK for Ruby.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteAccountPasswordPolicy`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte DeleteAccountPasswordPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteAccountPasswordPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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`.

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 um grupo e adicionar um usuário

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <summary>
    /// Delete an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Para obter detalhes da API, consulte DeleteGroup na Referência da API AWS SDK for .NET.

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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
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;
};
```

- Para obter detalhes da API, consulte DeleteGroup na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteGroup na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteGroupPolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `DeleteGroupPolicy`.

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 um grupo e adicionar um usuário

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obter detalhes da API, consulte DeleteGroupPolicy na Referência da API AWS SDK for .NET.

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.

- Para obter detalhes da API, consulte DeleteGroupPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteGroupPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no contexto no seguinte exemplo de código:

- Criar e gerenciar um serviço resiliente

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
   /// <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.");
        }
    }
```

- Para obter detalhes da API, consulte DeleteInstanceProfile na Referência da API do AWS SDK for .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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
const client = new IAMClient({});
await client.send(
  new DeleteInstanceProfileCommand({
    InstanceProfileName: NAMES.instanceProfileName,
  }),
);
```

- Para obter detalhes da API, consulte DeleteInstanceProfile na Referência da API do AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteInstanceProfile na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

Este exemplo remove o perfil do perfil de instância, separa todas as políticas anexadas ao perfil e exclui todos os recursos.

```
class AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :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
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
        self.instance_profile_name = f"{resource_prefix}-prof"
        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"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    def delete_instance_profile(self, profile_name, role_name):
        """
        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:
```

```
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete instance profile {profile_name} or detach "
                    f"policies and delete role {role_name}: {err}"
                )
```

- Para obter detalhes da API, consulte DeleteInstanceProfile na Referência da API do AWS SDK para Python (Boto3).

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteLoginProfile`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte DeleteLoginProfile na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteLoginProfile na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteOpenIdConnectProvider`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte DeleteOpenIdConnectProvider na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteOpenIdConnectProvider na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um usuário e assumir uma função
- Criar usuários somente leitura e leitura/gravação usando
- Políticas gerenciadas

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte DeletePolicy na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
###############################################################################
# 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 WS 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
}
```

- Para obter detalhes da API, consulte DeletePolicy 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::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();
}
```

- Para obter detalhes da API, consulte DeletePolicy na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
// 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(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &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
}
```

- Para obter detalhes da API, consulte DeletePolicy na Referência da API AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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");
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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);
};
```

- Para obter detalhes da API, consulte DeletePolicy na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun deleteIAMPolicy(policyARNVal: String?) {
```

```
    val request = DeletePolicyRequest {
        policyArn = policyARNVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deletePolicy(request)
        println("Successfully deleted $policyARNVal")
    }
}
```

- Para obter detalhes da API, consulte DeletePolicy na Referência da API AWS SDK for Kotlin.

PowerShell

Tools for PowerShell

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 perfl 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 da API, consulte DeletePolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
pub async fn delete_policy(client: &iamClient, policy: Policy) -> Result<(),
 iamError> {
    client
        .delete_policy()
        .policy_arn(policy.arn.unwrap())
        .send()
        .await?;
    Ok(())
}
```

- Para obter detalhes da API, consulte DeletePolicy na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    public func deletePolicy(policy: IAMClientTypes.Policy) async throws {
        let input = DeletePolicyInput(
            policyArn: policy.arn
        )
        do {
            _ = try await iamClient.deletePolicy(input: input)
        } catch {
            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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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.

- Para obter detalhes da API, consulte DeletePolicyVersion na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DeletePolicyVersion na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um usuário e assumir uma função
- Gerenciar funções

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte DeleteRole na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
##############################################################################
# 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 WS 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
}
```

- Para obter detalhes da API, consulte 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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
// 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(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &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
}
```

- Para obter detalhes da API, consulte DeleteRole na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

Exclua a função.

```
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);
};
```

- Para obter detalhes da API, consulte DeleteRole na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteRole na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte DeleteRole na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
  # Deletes a role and its attached policies.
  #
  # @param role_name [String] The name of the role to delete.
  def delete_role(role_name)
    begin
      # 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
  end
```

- Para obter detalhes da API, consulte DeleteRole na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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(())
}
```

- Para obter detalhes da API, consulte DeleteRole na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
public func deleteRole(role: IAMClientTypes.Role) async throws {
    let input = DeleteRoleInput(
        roleName: role.roleName
    )
    do {
        _ = try await iamClient.deleteRole(input: input)
    } catch {
        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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteRolePermissionsBoundary`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte DeleteRolePermissionsBoundary na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo mostra como remover o limite de permissões anexado a um perfil do IAM.

```
Remove-IAMRolePermissionsBoundary -RoleName MyRoleName
```

- Para obter detalhes da API, consulte DeleteRolePermissionsBoundary na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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`.

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte DeleteRolePolicy na Referência da API AWS SDK for .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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
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);
};
```

- Para obter detalhes da API, consulte DeleteRolePolicy na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteRolePolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
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;
};
```

- Para obter detalhes da API, consulte DeleteSAMLProvider na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteSAMLProvider na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 veja como configurar e executar no repositório de exemplos de código da AWS.

```
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;
}
```

- Para obter detalhes da API, consulte DeleteServerCertificate na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte DeleteServerCertificate na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte DeleteServerCertificate na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo exclui o certificado do servidor denominado **`MyServerCert`**.

```
Remove-IAMServerCertificate -ServerCertificateName MyServerCert
```

- Para obter detalhes da API, consulte DeleteServerCertificate na Referência de Cmdlet do AWS Tools for PowerShell.

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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
```

- Para obter detalhes da API, consulte DeleteServerCertificate na Referência da API do AWS SDK for Ruby.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
// 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(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &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 for 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 Repositório de exemplos de código da AWS.

```
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);
};
```

- Para obter detalhes da API, consulte DeleteServiceLinkedRole na Referência da API AWS SDK for JavaScript SDK for Kotlin.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteServiceLinkedRole na Referência de Cmdlet do AWS Tools for PowerShell.

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
# 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)
    unless e.code == "NoSuchEntity"
      @logger.error("Couldn't delete #{role_name}. Here's why:")
      @logger.error("\t#{e.code}: #{e.message}")
      raise
    end
  end
```

- Para obter detalhes da API, consulte DeleteServiceLinkedRole na Referência da API AWS SDK for Ruby SDK for Kotlin.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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(())
}
```

- Para obter detalhes da API, consulte DeleteServiceLinkedRole na Referência da API AWS SDK for Rust.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteSigningCertificate`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte DeleteSigningCertificate na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteSigningCertificate na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um grupo e adicionar um usuário
- Criar um usuário e assumir uma função
- Criar usuários somente leitura e leitura/gravação usando

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
```

```
    }
```

- Para obter detalhes da API, consulte DeleteUser na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
##############################################################################
# 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 WS 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
}
```

- Para obter detalhes da API, consulte DeleteUser 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.

```
    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();
```

- Para obter detalhes da API, consulte DeleteUser na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte DeleteUser na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &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
}
```

- Para obter detalhes da API, consulte DeleteUser na Referência da API AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte DeleteUser na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte DeleteUser na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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 { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.deleteUser(request)
        println("Successfully deleted user $userNameVal")
    }
}
```

- Para obter detalhes da API, consulte DeleteUser na Referência da API AWS SDK for Kotlin.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteUser na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte DeleteUser na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
 # 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
```

- Para obter detalhes da API, consulte DeleteUser na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
}
```

- Para obter detalhes da API, consulte DeleteUser na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
public func deleteUser(user: IAMClientTypes.User) async throws {
    let input = DeleteUserInput(
        userName: user.userName
    )
    do {
        _ = try await iamClient.deleteUser(input: input)
    } catch {
        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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteUserPermissionsBoundary`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte DeleteUserPermissionsBoundary na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo mostra como remover o limite de permissões anexado a um usuário do IAM.

```
Remove-IAMUserPermissionsBoundary -UserName joe
```

- Para obter detalhes da API, consulte DeleteUserPermissionsBoundary na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um usuário e assumir uma função

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte DeleteUserPolicy na Referência da API AWS SDK for .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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
// 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(userName string, policyName string)
 error {
```

```
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte DeleteUserPolicy na Referência da API AWS SDK for Go.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteUserPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
 # 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
```

- Para obter detalhes da API, consulte DeleteUserPolicy na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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(())
}
```

- Para obter detalhes da API, consulte DeleteUserPolicy na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
   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 {
            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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DeleteVirtualMfaDevice`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte DeleteVirtualMfaDevice na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DeleteVirtualMfaDevice na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DetachGroupPolicy`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte DetachGroupPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DetachGroupPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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:

- Criar um usuário e assumir uma função
- Gerenciar funções

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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 for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
###############################################################################
# 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 veja como configurar e executar no repositório de exemplos de código da AWS.

```
    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 for 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 Repositório de exemplos de código da AWS.

```
// 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(roleName string, policyArn string)
 error {
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &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 for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte.DetachRolePolicy na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte.DetachRolePolicy na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun detachPolicy(roleNameVal: String, policyArnVal: String) {

    val request = DetachRolePolicyRequest {
        roleName = roleNameVal
        policyArn = policyArnVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.detachRolePolicy(request)
        println("Successfully detached policy $policyArnVal from role
 $roleNameVal")
    }
}
```

```
```

- Para obter detalhes da API, consulte DetachRolePolicy na Referência da API AWS SDK for Kotlin.

PowerShell

Tools for PowerShell

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 da API, consulte DetachRolePolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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(())
}
```

- Para obter detalhes da API, consulte DetachRolePolicy na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
   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 {
            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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no 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.

- Para obter detalhes da API, consulte DetachUserPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DetachUserPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
 # 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 for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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(())
}
```

- Para obter detalhes da API, consulte DetachUserPolicy na Referência da API AWS SDK for Rust.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`EnableMfaDevice`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte EnableMfaDevice na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte EnableMfaDevice na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GenerateCredentialReport na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GenerateCredentialReport na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte GenerateCredentialReport na Referência da API AWS SDK for Python (Boto3).

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GenerateServiceLastAccessedDetails`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GenerateServiceLastAccessedDetails na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GenerateServiceLastAccessedDetails na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no contexto no seguinte exemplo de código:

- Gerenciar chaves de acesso

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::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();
}
```

- Para obter detalhes da API, consulte GetAccessKeyLastUsed na Referência da API AWS SDK for C++.

CLI

AWS CLI

Como 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.

- Para obter detalhes da API, consulte GetAccessKeyLastUsed na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte GetAccessKeyLastUsed na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte GetAccessKeyLastUsed na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte GetAccessKeyLastUsed na Referência da API AWS SDK for Python (Boto3).

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no contexto no seguinte exemplo de código:

- Gerenciar sua conta

CLI

AWS CLI

Como 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
```

Saída:

```
{
    "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:::example-bucket",
                                    "arn:aws:s3:::example-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.

- Para obter detalhes da API, consulte GetAccountAuthorizationDetails na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetAccountAuthorizationDetails na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte GetAccountAuthorizationDetails na Referência da API AWS SDK for Python (Boto3).

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte GetAccountPasswordPolicy na Referência da API AWS SDK for .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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetAccountPasswordPolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &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
}
```

- Para obter detalhes da API, consulte GetAccountPasswordPolicy na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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;
};
```

- Para obter detalhes da API, consulte GetAccountPasswordPolicy na Referência da API AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getAccountPasswordPolicy()
    {
        return $this->iamClient->getAccountPasswordPolicy();
    }
```

- Para obter detalhes da API, consulte GetAccountPasswordPolicy na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte GetAccountPasswordPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte GetAccountPasswordPolicy na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
# 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
    begin
      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
end
```

- Para obter detalhes da API, consulte GetAccountPasswordPolicy na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
pub async fn get_account_password_policy(
    client: &iamClient,
) -> Result<GetAccountPasswordPolicyOutput,
 SdkError<GetAccountPasswordPolicyError>> {
    let response = client.get_account_password_policy().send().await?;

    Ok(response)
}
```

- Para obter detalhes da API, consulte GetAccountPasswordPolicy na Referência da API AWS SDK para Rust.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetAccountSummary na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetAccountSummary na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte GetAccountSummary na Referência da API AWS SDK para Python (Boto3).

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetContextKeysForCustomPolicy`** com o AWS SDK ou a 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"}}}}'
```

Saída:

```
{
    "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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetContextKeysForCustomPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetContextKeysForCustomPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetContextKeysForPrincipalPolicy`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetContextKeysForPrincipalPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetContextKeysForPrincipalPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetCredentialReport na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetCredentialReport na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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"]
```

- Para obter detalhes da API, consulte GetCredentialReport na Referência da API AWS SDK for Python (Boto3).

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetGroup`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetGroup na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetGroup na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetGroupPolicy`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetGroupPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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",
      "NotAction": "iam:*",
      "Resource": "*"
    }
  ]
}
```

- Para obter detalhes da API, consulte GetGroupPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetInstanceProfile`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `GetInstanceProfile`.

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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetInstanceProfile na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetInstanceProfile na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetLoginProfile`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetLoginProfile na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetLoginProfile na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetOpenIdConnectProvider`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetOpenIdConnectProvider na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetOpenIdConnectProvider na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no contexto no seguinte exemplo de código:

- Trabalhar com a API IAM Policy Builder

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte GetPolicy na Referência da API AWS SDK for .NET.

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::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();
}
```

- Para obter detalhes da API, consulte GetPolicy na Referência da API AWS SDK for 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetPolicy na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &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
}
```

- Para obter detalhes da API, consulte GetPolicy na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte GetPolicy na Referência da API AWS SDK for 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 for JavaScript.

- Para obter detalhes da API, consulte GetPolicy na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun getIAMPolicy(policyArnVal: String?) {

    val request = GetPolicyRequest {
        policyArn = policyArnVal
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.getPolicy(request)
        println("Successfully retrieved policy ${response.policy?.policyName}")
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
$uuid = uniqid();
```

```
$service = new IAMService();

    public function getPolicy($policyArn)
    {
        return $this->customWaiter(function () use ($policyArn) {
            return $this->iamClient->getPolicy(['PolicyArn' => $policyArn]);
        });
    }
```

- Para obter detalhes da API, consulte GetPolicy na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte GetPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte GetPolicy na Referência da API AWS SDK para Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
 # 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
```

- Para obter detalhes da API, consulte GetPolicy na Referência da API AWS SDK for Ruby.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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 {
        throw error
    }
}
```

- Para obter detalhes da API, consulte GetPolicy na referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetPolicyVersion na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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": "*",
      "Resource": "*"
    }
}
```

- Para obter detalhes da API, consulte GetPolicyVersion na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte GetPolicyVersion na Referência da API AWS SDK para Python (Boto3).

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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`.

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
}
```

- Para obter detalhes da API, consulte GetRole na Referência da API AWS SDK for .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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetRole na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(roleName string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &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
}
```

- Para obter detalhes da API, consulte GetRole na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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);
};
```

- Para obter detalhes da API, consulte GetRole na Referência da API AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
$uuid = uniqid();
$service = new IAMService();

    public function getRole($roleName)
    {
        return $this->customWaiter(function () use ($roleName) {
            return $this->iamClient->getRole(['RoleName' => $roleName]);
        });
    }
```

- Para obter detalhes da API, consulte GetRole na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte GetRole na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte GetRole na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
# 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
```

- Para obter detalhes da API, consulte GetRole na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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)
}
```

- Para obter detalhes da API, consulte GetRole na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    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 {
            throw error
        }
```

```
    }
```

- Para obter detalhes da API, consulte GetRole na referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetRolePolicy`** com o AWS SDK ou a 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
```

Saída:

```
{
  "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.

- Para obter detalhes da API, consulte GetRolePolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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:*:*:*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::*"
      ]
    }
  ]
}
```

- Para obter detalhes da API, consulte GetRolePolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetSamlProvider`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetSamlProvider na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetSamlProvider na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 veja como configurar e executar no repositório de exemplos de código da AWS.

```
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;
}
```

- Para obter detalhes da API, consulte GetServerCertificate na Referência da API AWS SDK for 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
```

Saída:

```
{
```

```
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte GetServerCertificate na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte GetServerCertificate na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte GetServerCertificate na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetServiceLastAccessedDetails`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetServiceLastAccessedDetails na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetServiceLastAccessedDetails na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetServiceLastAccessedDetailsWithEntities`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetServiceLastAccessedDetailsWithEntities na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetServiceLastAccessedDetailsWithEntities na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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
```

Saída:

```
{
"Status": "SUCCEEDED"
}
```

Para obter mais informações, consulte Usar perfis vinculados ao serviço no Guia do usuário do AWS IAM.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
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 for JavaScript.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <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;
```

```
    }
```

- Para obter detalhes da API, consulte GetUser na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
##############################################################################
# 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
}
```

- Para obter detalhes da API, consulte 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
// 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(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &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
}
```

- Para obter detalhes da API, consulte GetUser na Referência da API AWS SDK for Go.

PowerShell

Tools for PowerShell

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 da API, consulte GetUser na Referência de Cmdlet do AWS Tools for PowerShell.

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
 # 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
```

- Para obter detalhes da API, consulte GetUser na Referência da API do AWS SDK for Ruby.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetUserPolicy`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetUserPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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:*"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

- Para obter detalhes da API, consulte GetUserPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no 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 Repositório de exemplos de código da AWS.

```
###############################################################################
# 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
}
```

- Para obter detalhes da API, consulte ListAccessKeys 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::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;
}
```

- Para obter detalhes da API, consulte ListAccessKeys na Referência da API AWS SDK for C ++.

CLI

AWS CLI

Como 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListAccessKeys na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &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 ListAccessKeys na Referência da API AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte ListAccessKeys na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte ListAccessKeys na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun listKeys(userNameVal: String?) {

    val request = ListAccessKeysRequest {
```

```
        userName = userNameVal
    }
    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccessKeys(request)
        response.accessKeyMetadata?.forEach { md ->
            println("Retrieved access key ${md.accessKeyId}")
        }
    }
}
```

- Para obter detalhes da API, consulte ListAccessKeys na Referência da API AWS SDK for Kotlin.

PowerShell

Tools for PowerShell

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 da API, consulte ListAccessKeys na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 => e
    @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 => e
    @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
```

- Para obter detalhes da API, consulte ListAccessKeys na Referência da API do AWS SDK for Ruby.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no contexto no seguinte exemplo de código:

- Gerenciar sua conta

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::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;
}
```

- Para obter detalhes da API, consulte ListAccountAliases na Referência da API AWS SDK for 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
```

Saída:

```
{
```

```
    "AccountAliases": [
    "mycompany"
    ]
}
```

Para obter mais informações, consulte O ID da sua conta da AWS e seu alias no Guia do usuário do AWS IAM.

- Para obter detalhes da API, consulte ListAccountAliases na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte ListAccountAliases na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte ListAccountAliases na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun listAliases() {

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        val response = iamClient.listAccountAliases(ListAccountAliasesRequest {})
        response.accountAliases?.forEach { alias ->
            println("Retrieved account alias $alias")
        }
    }
}
```

```
```

- Para obter detalhes da API, consulte ListAccountAliases na Referência da API AWS SDK for Kotlin.

PowerShell

Tools for PowerShell

Exemplo 1: este comando retorna o alias da conta da Conta da AWS.

```
Get-IAMAccountAlias
```

Saída:

```
ExampleCo
```

- Para obter detalhes da API, consulte ListAccountAliases na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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"]
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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
```

- Para obter detalhes da API, consulte ListAccountAliases na Referência da API AWS SDK for Ruby.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListAttachedGroupPolicies`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListAttachedGroupPolicies na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ListAttachedGroupPolicies na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte ListAttachedRolePolicies na Referência da API AWS SDK for .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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListAttachedRolePolicies na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte ListAttachedRolePolicies na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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;
    }
  }
}
```

- Para obter detalhes da API, consulte ListAttachedRolePolicies na Referência da API AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
$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);
    }
```

- Para obter detalhes da API, consulte ListAttachedRolePolicies na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte ListAttachedRolePolicies na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 ListAttachedRolePolicies na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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)
}
```

- Para obter detalhes da API, consulte ListAttachedRolePolicies na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// 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] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
            let input = ListAttachedRolePoliciesInput(
                marker: marker,
                roleName: role
            )
            let output = try await client.listAttachedRolePolicies(input: input)
```

```
            guard let attachedPolicies = output.attachedPolicies else {
                return policyList
            }

            for attachedPolicy in attachedPolicies {
                policyList.append(attachedPolicy)
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return policyList
    }
```

- Para obter detalhes da API, consulte ListAttachedRolePolicies na referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListAttachedUserPolicies`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListAttachedUserPolicies na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ListAttachedUserPolicies na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListEntitiesForPolicy`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListEntitiesForPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ListEntitiesForPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListGroupPolicies`** com o AWS SDK ou a 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
```

Saída:

```
{
    "PolicyNames": [
        "AdminRoot",
        "ExamplePolicy"
    ]
}
```

Para obter mais informações, consulte Gerenciamento de políticas do IAM no Guia do usuário do AWS IAM.

- Para obter detalhes da API, consulte ListGroupPolicies na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ListGroupPolicies na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte ListGroups na Referência da API AWS SDK for .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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListGroups na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(maxGroups int32) ([]types.Group, error) {
 var groups []types.Group
 result, err := wrapper.IamClient.ListGroups(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte ListGroups na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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;
```

```
    }
  }
}
```

- Para obter detalhes da API, consulte ListGroups na Referência da API AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
$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);
    }
```

- Para obter detalhes da API, consulte ListGroups na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo retorna uma compilação de todos os grupos do IAM definidos na Conta da AWS atual.

```
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 da API, consulte ListGroups na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte ListGroups na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
# 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
```

- Para obter detalhes da API, consulte ListGroups na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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)
}
```

- Para obter detalhes da API, consulte ListGroups na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
public func listGroups() async throws -> [String] {
    var groupList: [String] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListGroupsInput(marker: marker)
        let output = try await client.listGroups(input: input)

        guard let groups = output.groups else {
            return groupList
        }

        for group in groups {
            if let name = group.groupName {
                groupList.append(name)
            }
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
```

```
        return groupList
    }
```

- Para obter detalhes da API, consulte ListGroups na referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListGroupsForUser`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListGroupsForUser na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ListGroupsForUser na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListInstanceProfiles`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListInstanceProfiles na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo retorna uma compilação dos perfis de instância definidos na Conta da AWS atual.

```
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 da API, consulte ListInstanceProfiles na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListInstanceProfilesForRole`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListInstanceProfilesForRole na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ListInstanceProfilesForRole na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListMfaDevices`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListMfaDevices na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ListMfaDevices na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListOpenIdConnectProviders`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListOpenIdConnectProviders na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo retorna uma lista de ARNS de todos os provedores OpenID Connect definidos na Conta da AWS atual.

```
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 da API, consulte ListOpenIdConnectProviders na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK for .NET.

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::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;
}
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK for C++.

CLI

AWS CLI

Como 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListPolicies na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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;
    }
  }
}
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
$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);
    }
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte ListPolicies na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 ListPolicies na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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)
}
```

- Para obter detalhes da API, consulte ListPolicies na Referência da API AWS SDK para Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    public func listPolicies() async throws -> [MyPolicyRecord] {
        var policyList: [MyPolicyRecord] = []
        var marker: String? = nil
        var isTruncated: Bool
```

```
        repeat {
            let input = ListPoliciesInput(marker: marker)
            let output = try await client.listPolicies(input: input)

            guard let policies = output.policies else {
                return policyList
            }

            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))
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListPolicyVersions na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ListPolicyVersions na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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`.

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte ListRolePolicies na Referência da API AWS SDK for .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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListRolePolicies na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte ListRolePolicies na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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;
    }
  }
}
```

- Para obter detalhes da API, consulte ListRolePolicies na Referência da API AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
$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);
        });
    }
```

- Para obter detalhes da API, consulte ListRolePolicies na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

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 da API, consulte ListRolePolicies na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
# 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
```

- Para obter detalhes da API, consulte ListRolePolicies na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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)
}
```

- Para obter detalhes da API, consulte ListRolePolicies na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
public func listRolePolicies(role: String) async throws -> [String] {
    var policyList: [String] = []
    var marker: String? = nil
    var isTruncated: Bool

    repeat {
        let input = ListRolePoliciesInput(
            marker: marker,
            roleName: role
        )
        let output = try await client.listRolePolicies(input: input)

        guard let policies = output.policyNames else {
            return policyList
        }

        for policy in policies {
            policyList.append(policy)
        }
        marker = output.marker
        isTruncated = output.isTruncated
    } while isTruncated == true
    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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListRoleTags`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListRoleTags na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo busca a tag associada ao perfil.

```
Get-IAMRoleTagList -RoleName MyRoleName
```

- Para obter detalhes da API, consulte ListRoleTags na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte ListRoles na Referência da API AWS SDK for .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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListRoles na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(maxRoles int32) ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &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
}
```

- Para obter detalhes da API, consulte ListRoles na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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;
    }
  }
}
```

- Para obter detalhes da API, consulte ListRoles na Referência da API AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
$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);
    }
```

- Para obter detalhes da API, consulte ListRoles na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo recupera uma lista de todos os perfis do IAM na Conta da AWS.

```
Get-IAMRoleList
```

Exemplo 2: este trecho de código de exemplo recupera uma lista de perfis do IAM na conta da AWS, exibe três deles de cada vez e espera que você pressione Enter entre cada grupo. Ele passa o valor de **`Marker`** da chamada anterior para especificar onde o próximo grupo deve começar.

```
$nextMarker = $null
Do
{
    $results = Get-IAMRoleList -MaxItem 3 -Marker $nextMarker
    $nextMarker = $AWSHistory.LastServiceResponse.Marker
    $results
    Read-Host
} while ($nextMarker -ne $null)
```

- Para obter detalhes da API, consulte ListRoles na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte ListRoles, na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
  # 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
```

- Para obter detalhes da API, consulte ListRoles na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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)
}
```

- Para obter detalhes da API, consulte ListRoles na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    public func listRoles() async throws -> [String] {
        var roleList: [String] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
            let input = ListRolesInput(marker: marker)
            let output = try await client.listRoles(input: input)
```

```
            guard let roles = output.roles else {
                return roleList
            }

            for role in roles {
                if let name = role.roleName {
                    roleList.append(name)
                }
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return roleList
    }
```

- Para obter detalhes da API, consulte ListRoles na referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte ListSAMLProvider na Referência da API AWS SDK for .NET.

CLI

AWS CLI

Como 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListSAMLProviders na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte ListSAMLProvider na Referência da API AWS SDK for Go.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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;
};
```

- Para obter detalhes da API, consulte ListSAMLProvider na Referência da API AWS SDK for JavaScript.

PHP

SDK para PHP

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
$uuid = uniqid();
$service = new IAMService();
```

```
    public function listSAMLProviders()
    {
        return $this->iamClient->listSAMLProviders();
    }
```

- Para obter detalhes da API, consulte ListSAMLProvider na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo recupera a lista de provedores SAML 2.0 criados na Conta da AWS atual. 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 da API, consulte ListSAMLProviders na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte ListSAMLProvider na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte ListSAMLProvider na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
pub async fn list_saml_providers(
    client: &Client,
) -> Result<ListSamlProvidersOutput, SdkError<ListSAMLProvidersError>> {
    let response = client.list_saml_providers().send().await?;

    Ok(response)
}
```

- Para obter detalhes da API, consulte ListSAMLProvider na referência da API do AWS SDK for Rust.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 veja como configurar e executar no repositório de exemplos de código da AWS.

```
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;
}
```

- Para obter detalhes da API, consulte ListServerCertificates na Referência da API AWS SDK for C++.

CLI

AWS CLI

Como 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte ListServerCertificates na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte ListServerCertificates na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo recupera a lista de certificados de servidor enviados à Conta da AWS atual.

```
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 da API, consulte ListServerCertificates na Referência de Cmdlet do AWS Tools for PowerShell.

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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
```

- Para obter detalhes da API, consulte ListServerCertificates na Referência da API do AWS SDK for Ruby.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListSigningCertificates`** com o AWS SDK ou a 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
```

Saída:

```
{
```

```
    "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.

- Para obter detalhes da API, consulte ListSigningCertificates na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ListSigningCertificates na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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

Como 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
// 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(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte ListUserPolicies na Referência da API AWS SDK for Go.

PowerShell

Tools for PowerShell

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 da API, consulte ListUserPolicies na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListUserTags`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListUserTags na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo busca a tag associada ao usuário.

```
Get-IAMUserTagList -UserName joe
```

- Para obter detalhes da API, consulte ListUserTags na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no AWS Code 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;
    }
```

- Para obter detalhes da API, consulte ListUsers na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
##############################################################################
# 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
}
```

- Para obter detalhes da API, consulte ListUsers 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::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;
}
```

- Para obter detalhes da API, consulte ListUsers na Referência da API AWS SDK for C++.

CLI

AWS CLI

Como listar os usuários do IAM

O comando `list-users`, apresentado a seguir, lista os usuários do IAM na conta atual.

```
aws iam list-users
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte ListUsers na Referência de comandos da AWS CLI.

Go

SDK para Go V2

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
// 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(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &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
}
```

- Para obter detalhes da API, consulte ListUsers na Referência da API AWS SDK for Go.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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);
  response.Users?.forEach(({ UserName, CreateDate }) => {
    console.log(`${UserName} created on: ${CreateDate}`);
  });
  return response;
};
```

- Para obter mais informações, consulte o Guia do desenvolvedor do AWS SDK for JavaScript.
- Para obter detalhes da API, consulte ListUsers na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte ListUsers na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun listAllUsers() {

    IamClient { 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}")
        }
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
$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);
    }
```

- Para obter detalhes da API, consulte ListUsers na Referência da API AWS SDK for PHP.

PowerShell

Tools for PowerShell

Exemplo 1: este exemplo recupera uma compilação de usuários na Conta da AWS atual.

```
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 da API, consulte ListUsers na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte ListUsers na Referência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
# 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
```

- Para obter detalhes da API, consulte ListUsers na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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)
}
```

- Para obter detalhes da API, consulte ListUsers na Referência da API AWS SDK for Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    public func listUsers() async throws -> [MyUserRecord] {
        var userList: [MyUserRecord] = []
        var marker: String? = nil
        var isTruncated: Bool

        repeat {
            let input = ListUsersInput(marker: marker)
```

```
            let output = try await client.listUsers(input: input)

            guard let users = output.users else {
                return userList
            }

            for user in users {
                if let id = user.userId, let name = user.userName {
                    userList.append(MyUserRecord(id: id, name: name))
                }
            }
            marker = output.marker
            isTruncated = output.isTruncated
        } while isTruncated == true
        return userList
    }
```

- Para obter detalhes da API, consulte ListUsers na referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`ListVirtualMfaDevices`** com o AWS SDK ou a 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
```

Saída:

```
{
```

```
    "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.

- Para obter detalhes da API, consulte ListVirtualMfaDevices na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ListVirtualMfaDevices na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`PutGroupPolicy`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `PutGroupPolicy`.

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 um grupo e adicionar um usuário

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
   /// <summary>
   /// Add or update an inline policy document that is embedded in an IAM group.
   /// </summary>
   /// <param name="groupName">The name of the IAM group.</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> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
   {
       var request = new PutGroupPolicyRequest
       {
           GroupName = groupName,
           PolicyName = policyName,
           PolicyDocument = policyDocument
       };
```

```
        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
```

- Para obter detalhes da API, consulte PutGroupPolicy na Referência da API AWS SDK for .NET.

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.

- Para obter detalhes da API, consulte PutGroupPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte PutGroupPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **PutRolePermissionsBoundary** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte PutRolePermissionsBoundary na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte PutRolePermissionsBoundary na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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`.

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <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;
    }
```

- Para obter detalhes da API, consulte PutRolePolicy na Referência da API AWS SDK for .NET.

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::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();
}
```

- Para obter detalhes da API, consulte PutRolePolicy na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
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:::some-test-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;
};
```

- Para obter detalhes da API, consulte PutRolePolicy na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte PutRolePolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`PutUserPermissionsBoundary`** com o AWS SDK ou a 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 obter mais informações, consulte Adicionar e remover permissões de identidade do IAM no Guia do usuário do AWS IAM.

- Para obter detalhes da API, consulte PutUserPermissionsBoundary na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte PutUserPermissionsBoundary na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no contexto no seguinte exemplo de código:

- Criar um usuário e assumir uma função

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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
// 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(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(context.TODO(),
 &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
}
```

- Para obter detalhes da API, consulte PutUserPolicy na Referência da API AWS SDK for Go.

PowerShell

Tools for PowerShell

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 da API, consulte PutUserPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
# 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
```

- Para obter detalhes da API, consulte PutUserPolicy na Referência da API AWS SDK for Ruby.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    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 {
            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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`RemoveClientIdFromOpenIdConnectProvider`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte RemoveClientIdFromOpenIdConnectProvider na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte RemoveClientIdFromOpenIdConnectProvider na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`RemoveRoleFromInstanceProfile`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `RemoveRoleFromInstanceProfile`.

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.

- Para obter detalhes da API, consulte RemoveRoleFromInstanceProfile na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte RemoveRoleFromInstanceProfile na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`RemoveUserFromGroup`** com o AWS SDK ou a CLI

Os exemplos de código a seguir mostram como usar o `RemoveUserFromGroup`.

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 um grupo e adicionar um usuário

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
    /// <summary>
```

```
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Para obter detalhes da API, consulte RemoveUserFromGroup na Referência da API AWS SDK for .NET.

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.

- Para obter detalhes da API, consulte RemoveUserFromGroup na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte RemoveUserFromGroup na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **ResyncMfaDevice** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte ResyncMfaDevice na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte ResyncMfaDevice na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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.

- Para obter detalhes da API, consulte SetDefaultPolicyVersion na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte SetDefaultPolicyVersion na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **TagRole** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte TagRole na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte TagRole na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`TagUser`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte TagUser na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte TagUser na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UntagRole`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte UntagRole na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UntagRole na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UntagUser`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte UntagUser na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UntagUser na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no contexto no seguinte exemplo de código:

- Gerenciar chaves de acesso

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::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();
}
```

- Para obter detalhes da API, consulte UpdateAccessKey na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte UpdateAccessKey na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte UpdateAccessKey na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte UpdateAccessKey na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateAccessKey na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte UpdateAccessKey, na Referência da API AWS SDK for Python (Boto3).

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateAccountPasswordPolicy`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte UpdateAccountPasswordPolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateAccountPasswordPolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateAssumeRolePolicy`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte UpdateAssumeRolePolicy na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateAssumeRolePolicy na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateGroup`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte UpdateGroup na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateGroup na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateLoginProfile`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte UpdateLoginProfile na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateLoginProfile na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateOpenIdConnectProviderThumbprint`** com o AWS SDK ou a 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
```

Este comando não produz saída.

Para obter mais informações, consulte Criar provedores de identidade OpenID Connect (OIDC) no Guia do usuário do AWS IAM.

- Para obter detalhes da API, consulte UpdateOpenIdConnectProviderThumbprint na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateOpenIdConnectProviderThumbprint na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateRole`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte UpdateRole na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateRole na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateRoleDescription`** com o AWS SDK ou a 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'
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte UpdateRoleDescription na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateRoleDescription na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateSamlProvider`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte UpdateSamlProvider na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateSamlProvider na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 veja como configurar e executar no repositório de exemplos de código da AWS.

```
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;
}
```

- Para obter detalhes da API, consulte UpdateServerCertificate na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte UpdateServerCertificate na Referência da API AWS SDK for 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 for JavaScript.

- Para obter detalhes da API, consulte UpdateServerCertificate na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateServerCertificate na Referência de Cmdlet do AWS Tools for PowerShell.

Ruby

SDK para Ruby

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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
```

- Para obter detalhes da API, consulte UpdateServerCertificate na Referência da API do AWS SDK for Ruby.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateSigningCertificate`** com o AWS SDK ou a 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.

- Para obter detalhes da API, consulte UpdateSigningCertificate na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateSigningCertificate na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no 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 veja como configurar e executar no repositório de exemplos de código da AWS.

```
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();
}
```

- Para obter detalhes da API, consulte UpdateUser na Referência da API AWS SDK for 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.

- Para obter detalhes da API, consulte UpdateUser na Referência de comandos da AWS CLI.

Java

SDK para Java 2.x

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
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);
        }
    }
}
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

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 for JavaScript.
- Para obter detalhes da API, consulte UpdateUser na Referência da API AWS SDK for 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 for JavaScript.
- Para obter detalhes da API, consulte UpdateUser na Referência da API AWS SDK for JavaScript.

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

```
suspend fun updateIAMUser(curName: String?, newName: String?) {

    val request = UpdateUserRequest {
        userName = curName
        newUserName = newName
    }

    IamClient { region = "AWS_GLOBAL" }.use { iamClient ->
        iamClient.updateUser(request)
        println("Successfully updated user to $newName")
    }
}
```

- Para obter detalhes da API, consulte UpdateUser na Referência da API AWS SDK para Kotlin.

PowerShell

Tools for PowerShell

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 da API, consulte UpdateUser na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
# 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
```

- Para obter detalhes da API, consulte UpdateUser na Referência da API AWS SDK for Ruby.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
import { UploadServerCertificateCommand, IAMClient } from "@aws-sdk/client-iam";
import { readFileSync } from "fs";
import { dirnameFromMetaUrl } from "@aws-doc-sdk-examples/lib/utils/util-fs.js";
import * as path from "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);
};
```

- Para obter detalhes da API, consulte UploadServerCertificate na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte UploadServerCertificate na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UploadSigningCertificate`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte UploadSigningCertificate na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte UploadSigningCertificate na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 vários perfis no IAM. Cada exemplo inclui um link para o GitHub, em que é possível encontrar instruções sobre como configurar e executar o código.

Exemplos

- Criar e gerenciar um serviço resiliente usando um AWS SDK
- Criar um grupo do IAM e adicionar um usuário ao grupo usando um AWS SDK
- Criar um usuário do IAM e assumir uma função com o AWS STS usando um AWS SDK
- Criar usuários do IAM somente leitura e leitura/gravação usando um AWS SDK
- 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
- Reverter uma versão de política do IAM usando um AWS SDK

- 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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 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)
    {
        _amazonAutoScaling = amazonAutoScaling;
        _amazonEc2 = amazonEc2;
        _amazonSsm = amazonSsm;
        _amazonIam = amazonIam;

        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)
    {
        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;

    }

    /// <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()
    {
        var zoneResponse = await _amazonEc2.DescribeAvailabilityZonesAsync(
            new DescribeAvailabilityZonesRequest());
        return zoneResponse.AvailabilityZones.Select(z => z.ZoneName).ToList();
    }

    /// <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()
    {
        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];
    }

    /// <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)
    {
        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;
    }

    /// <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 (AmazonClientException)
```

```
        {
            Console.WriteLine($"Unable to delete template {templateName}.");
        }
    }

    /// <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)
    {
        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];
    }

    /// <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)
    {
        await _amazonEc2.ReplaceIamInstanceProfileAssociationAsync(
                new ReplaceIamInstanceProfileAssociationRequest()
                {
                    AssociationId = associationId,
                    IamInstanceProfile = new IamInstanceProfileSpecification()
                    {
                        Name = credsProfileName
                    }
                });
        // Allow time before resetting.
        Thread.Sleep(25000);
        var instanceReady = false;
        var retries = 5;
        while (retries-- > 0 && !instanceReady)
        {
            await _amazonEc2.RebootInstancesAsync(
                new RebootInstancesRequest(new List<string>() { instanceId }));
            Thread.Sleep(10000);
```

```
            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($"Sending restart command to instance {instanceId}");
        await _amazonSsm.SendCommandAsync(
            new SendCommandRequest()
            {
                InstanceIds = new List<string>() { instanceId },
                DocumentName = "AWS-RunShellScript",
                Parameters = new Dictionary<string, List<string>>()
                {
                        {"commands", new List<string>() { "cd / && sudo python3
server.py 80" }}
                }
            });
        Console.WriteLine($"Restarted the web server on instance {instanceId}");
    }

    /// <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 }
            });
    }
}
```

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 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 for .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 Repositório de exemplos de código da AWS.

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)
                    .defaultActions(action)
                    .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 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 Repositório de exemplos de código da AWS.

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 "url";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios);
}
```

Criar etapas para implantar todos os recursos.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
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 { 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 () => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    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,
        },
      }),
      // snippet-end:[javascript.v3.wkflw.resilient.CreateLaunchTemplate]
    );
  }),
  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(
        "${AVAILABILITY_ZONE_NAMES}",
        state.availabilityZoneNames.join(", "),
      ),
),
new ScenarioInput("confirmContinue", MESSAGES.confirmContinue, {
  type: "confirm",
}),
new ScenarioOutput("loadBalancer", MESSAGES.loadBalancer),
new ScenarioOutput("gettingVpc", MESSAGES.gettingVpc),
new ScenarioAction("getVpc", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeVpcs]
  const client = new EC2Client({});
  const { Vpcs } = await client.send(
    new DescribeVpcsCommand({
      Filters: [{ Name: "is-default", Values: ["true"] }],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeVpcs]
  state.defaultVpc = Vpcs[0].VpcId;
}),
new ScenarioOutput("gotVpc", (state) =>
  MESSAGES.gotVpc.replace("${VPC_ID}", state.defaultVpc),
),
new ScenarioOutput("gettingSubnets", MESSAGES.gettingSubnets),
new ScenarioAction("getSubnets", async (state) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeSubnets]
  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"] },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.DescribeSubnets]
    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) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateTargetGroup]
    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,
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateTargetGroup]
```

```
    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) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
    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] },
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateLoadBalancer]
  }),
  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) => {
    // snippet-start:[javascript.v3.wkflw.resilient.CreateListener]
    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 },
        ],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.CreateListener]
    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) => {
    // snippet-start:[javascript.v3.wkflw.resilient.AttachTargetGroup]
    const client = new AutoScalingClient({});
    await client.send(
      new AttachLoadBalancerTargetGroupsCommand({
        AutoScalingGroupName: NAMES.autoScalingGroupName,
        TargetGroupARNs: [state.targetGroupArn],
      }),
    );
    // snippet-end:[javascript.v3.wkflw.resilient.AttachTargetGroup]
  }),
  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),
        );
      } else {
        return MESSAGES.noIpRules;
      }
    },
  ),
  new ScenarioInput(
    "shouldAddInboundRule",
    /**
     * @param {{ myIpRules: any[] }} state
     */
    (state) => {
      if (state.myIpRules.length > 0) {
        return false;
      } else {
        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("${IP_ADDRESS}", state.myIp);
    } else {
      return false;
    }
  }),
  new ScenarioOutput("verifyingEndpoint", (state) =>
    MESSAGES.verifyingEndpoint.replace("${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,
      );
    }
  }),
];
```

Criar etapas para executar a demonstração.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
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) => {
  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetGroups]
  const client = new ElasticLoadBalancingV2Client({});
  const { TargetGroups } = await client.send(
    new DescribeTargetGroupsCommand({
      Names: [NAMES.loadBalancerTargetGroupName],
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetGroups]

  // snippet-start:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  const { TargetHealthDescriptions } = await client.send(
    new DescribeTargetHealthCommand({
      TargetGroupArn: TargetGroups[0].TargetGroupArn,
    }),
  );
  // snippet-end:[javascript.v3.wkflw.resilient.DescribeTargetHealth]
  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];
      // snippet-start:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      const ec2Client = new EC2Client({});
      const { IamInstanceProfileAssociations } = await ec2Client.send(
        new DescribeIamInstanceProfileAssociationsCommand({
          Filters: [
            { Name: "instance-id", Values: [state.targetInstance.InstanceId] },
          ],
        }),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.DescribeIamInstanceProfileAssociations]
      state.instanceProfileAssociationId =
        IamInstanceProfileAssociations[0].AssociationId;
      // snippet-start:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
      await retry({ intervalInMs: 1000, maxRetries: 30 }, () =>
        ec2Client.send(
          new ReplaceIamInstanceProfileAssociationCommand({
            AssociationId: state.instanceProfileAssociationId,
            IamInstanceProfile: { Name: NAMES.ssmOnlyInstanceProfileName },
          }),
        ),
      );
      // snippet-end:
[javascript.v3.wkflw.resilient.ReplaceIamInstanceProfileAssociation]
```

```
      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",
    }),
  );
  // snippet-start:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  const { InstanceProfile } = await iamClient.send(
    new CreateInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
    }),
  );
  await waitUntilInstanceProfileExists(
    { client: iamClient },
    { InstanceProfileName: NAMES.ssmOnlyInstanceProfileName },
  );
  // snippet-end:[javascript.v3.wkflw.resilient.CreateInstanceProfile]
  await iamClient.send(
    new AddRoleToInstanceProfileCommand({
      InstanceProfileName: NAMES.ssmOnlyInstanceProfileName,
      RoleName: NAMES.ssmOnlyRoleName,
    }),
  );

  return InstanceProfile;
}
```

Criar etapas para destruir todos os recursos.

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0
import { unlinkSync } from "node:fs";

import { DynamoDBClient, DeleteTableCommand } from "@aws-sdk/client-dynamodb";
import {
```

```
  EC2Client,
  DeleteKeyPairCommand,
  DeleteLaunchTemplateCommand,
} 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 { 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 = [
  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(
        "${TABLE_NAME}",
        NAMES.tableName,
      );
    } else {
      return MESSAGES.deletedTable.replace("${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,
      );
    } else {
      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("${INSTANCE_ROLE_NAME}", NAMES.instanceRoleName);
    } else {
      return MESSAGES.detachedPolicyFromRole
        .replace("${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(
        "${INSTANCE_POLICY_NAME}",
        NAMES.instancePolicyName,
      );
    } else {
      return MESSAGES.deletedPolicy.replace(
        "${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);
    } else {
      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(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    } else {
      return MESSAGES.deletedInstanceRole.replace(
        "${INSTANCE_ROLE_NAME}",
        NAMES.instanceRoleName,
      );
    }
  }),
  new ScenarioAction("deleteInstanceProfile", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
      const client = new IAMClient({});
      await client.send(
        new DeleteInstanceProfileCommand({
          InstanceProfileName: NAMES.instanceProfileName,
        }),
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteInstanceProfile]
    } catch (e) {
      state.deleteInstanceProfileError = e;
    }
  }),
  new ScenarioOutput("deleteInstanceProfileResult", (state) => {
    if (state.deleteInstanceProfileError) {
```

```
      console.error(state.deleteInstanceProfileError);
      return MESSAGES.deleteInstanceProfileError.replace(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.instanceProfileName,
      );
    } else {
      return MESSAGES.deletedInstanceProfile.replace(
        "${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(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    } else {
      return MESSAGES.deletedAutoScalingGroup.replace(
        "${AUTO_SCALING_GROUP_NAME}",
        NAMES.autoScalingGroupName,
      );
    }
  }),
  new ScenarioAction("deleteLaunchTemplate", async (state) => {
    const client = new EC2Client({});
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
      await client.send(
        new DeleteLaunchTemplateCommand({
          LaunchTemplateName: NAMES.launchTemplateName,
        }),
```

```
      );
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLaunchTemplate]
    } catch (e) {
      state.deleteLaunchTemplateError = e;
    }
  }),
  new ScenarioOutput("deleteLaunchTemplateResult", (state) => {
    if (state.deleteLaunchTemplateError) {
      console.error(state.deleteLaunchTemplateError);
      return MESSAGES.deleteLaunchTemplateError.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    } else {
      return MESSAGES.deletedLaunchTemplate.replace(
        "${LAUNCH_TEMPLATE_NAME}",
        NAMES.launchTemplateName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancer", async (state) => {
    try {
      // snippet-start:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
      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.");
        }
      });
      // snippet-end:[javascript.v3.wkflw.resilient.DeleteLoadBalancer]
    } catch (e) {
      state.deleteLoadBalancerError = e;
    }
  }),
  new ScenarioOutput("deleteLoadBalancerResult", (state) => {
    if (state.deleteLoadBalancerError) {
      console.error(state.deleteLoadBalancerError);
```

```
      return MESSAGES.deleteLoadBalancerError.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    } else {
      return MESSAGES.deletedLoadBalancer.replace(
        "${LB_NAME}",
        NAMES.loadBalancerName,
      );
    }
  }),
  new ScenarioAction("deleteLoadBalancerTargetGroup", async (state) => {
    // snippet-start:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
    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;
    }
    // snippet-end:[javascript.v3.wkflw.resilient.DeleteTargetGroup]
  }),
  new ScenarioOutput("deleteLoadBalancerTargetGroupResult", (state) => {
    if (state.deleteLoadBalancerTargetGroupError) {
      console.error(state.deleteLoadBalancerTargetGroupError);
      return MESSAGES.deleteLoadBalancerTargetGroupError.replace(
        "${TARGET_GROUP_NAME}",
        NAMES.loadBalancerTargetGroupName,
      );
    } else {
      return MESSAGES.deletedLoadBalancerTargetGroup.replace(
        "${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);
    } else {
      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);
    } else {
      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");
    } else {
      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(
        "${INSTANCE_PROFILE_NAME}",
        NAMES.ssmOnlyInstanceProfileName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyInstanceProfile.replace(
        "${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(
        "${POLICY_NAME}",
        NAMES.ssmOnlyPolicyName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyPolicy.replace(
        "${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(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    } else {
      return MESSAGES.deletedSsmOnlyRole.replace(
        "${ROLE_NAME}",
        NAMES.ssmOnlyRoleName,
      );
    }
  }),
];

/**
 * @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;
    } else {
      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 for 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 Repositório de exemplos de código da AWS.

Execute o cenário interativo em um prompt de comando.

```
class Runner:
    def __init__(
        self, resource_path, recommendation, autoscaler, loadbalancer,
 param_helper
    ):
        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

    def deploy(self):
        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"

        print(
```

```
            "\nFor this demo, we'll use the AWS SDK for Python (Boto3) 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"
        )
        print(
            "\t* A DynamoDB table that the web service depends on to provide
book, movie, and song recommendations."
        )
        print(
            "\t* An EC2 launch template that defines EC2 instances that each
contain a Python web server."
        )
        print(
            "\t* An EC2 Auto Scaling group that manages EC2 instances across
several Availability Zones."
        )
        print(
            "\t* An Elastic Load Balancing (ELB) load balancer that targets the
Auto Scaling group to distribute requests."
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to start deploying resources.")

        print(
            f"Creating and populating a DynamoDB table named
'{self.recommendation.table_name}'."
        )
        self.recommendation.create()
        self.recommendation.populate(recommendations_path)
        print("-" * 88)

        print(
            f"Creating an EC2 launch template that runs '{startup_script}' when
an instance starts.\n"
            f"This script starts a Python web server defined in the `server.py`
script. The web server\n"
            f"listens to HTTP requests on port 80 and responds to requests to '/'
and to '/healthcheck'.\n"
            f"For demo purposes, this server is run as the root user. In
production, the best practice is to\n"
```

```
            f"run a web server, such as Apache, with least-privileged
credentials.\n"
        )
        print(
            f"The template also defines an IAM policy that each instance uses to
assume a role that grants\n"
            f"permissions to access the DynamoDB recommendation table and Systems
Manager parameters\n"
            f"that control the flow of the demo.\n"
        )
        self.autoscaler.create_template(startup_script, instance_policy)
        print("-" * 88)

        print(
            f"Creating an EC2 Auto Scaling group that maintains three EC2
instances, each in a different\n"
            f"Availability Zone."
        )
        zones = self.autoscaler.create_group(3)
        print("-" * 88)
        print(
            "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."
        )
        print("-" * 88)
        q.ask("Press Enter when you're ready to continue.")

        print(f"Creating variables that control the flow of the demo.\n")
        self.param_helper.reset()

        print(
            "\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.\n"
        )
        vpc = self.autoscaler.get_default_vpc()
        subnets = self.autoscaler.get_subnets(vpc["VpcId"], zones)
        target_group = self.loadbalancer.create_target_group(
            self.protocol, self.port, vpc["VpcId"]
```

```
        )
        self.loadbalancer.create_load_balancer(
            [subnet["SubnetId"] for subnet in subnets], target_group
        )
        self.autoscaler.attach_load_balancer_target_group(target_group)
        print(f"Verifying access to the load balancer endpoint...")
        lb_success = self.loadbalancer.verify_load_balancer_endpoint()
        if not lb_success:
            print(
                "Couldn't connect to the load balancer, verifying that the port
 is open..."
            )
            current_ip_address = requests.get(
                "http://checkip.amazonaws.com"
            ).text.strip()
            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:
                print(
                    "For 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 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()
        if lb_success:
            print("Your load balancer is ready. You can access it by browsing to:
\n")
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        else:
            print(
                "Couldn'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"
            )
            print(f"\thttp://{self.loadbalancer.endpoint()}\n")
        print("-" * 88)
        q.ask("Press Enter when you're ready to continue with the demo.")

    def demo_choices(self):
        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:
            print("-" * 88)
            print(
                "\nSee the current state of the service by selecting one of the
 following choices:\n"
            )
            choice = q.choose("\nWhich action would you like to take? ", actions)
            print("-" * 88)
            if choice == 0:
                print("Request:\n")
                print(f"GET http://{self.loadbalancer.endpoint()}")
                response = requests.get(f"http://{self.loadbalancer.endpoint()}")
```

```
                print("\nResponse:\n")
                print(f"{response.status_code}")
                if response.headers.get("content-type") == "application/json":
                    pp(response.json())
            elif choice == 1:
                print("\nChecking the health of load balancer targets:\n")
                health = self.loadbalancer.check_target_health()
                for target in health:
                    state = target["TargetHealth"]["State"]
                    print(
                        f"\tTarget {target['Target']['Id']} on port
{target['Target']['Port']} is {state}"
                    )
                    if state != "healthy":
                        print(
                            f"\t\t{target['TargetHealth']['Reason']}:
{target['TargetHealth']['Description']}\n"
                        )
                print(
                    f"\nNote that it can take a minute or two for the health
check to update\n"
                    f"after changes are made.\n"
                )
            elif choice == 2:
                print("\nOkay, let's move on.")
                print("-" * 88)

    def demo(self):
        ssm_only_policy = f"{self.resource_path}/ssm_only_policy.json"

        print("\nResetting parameters to starting values for demo.\n")
        self.param_helper.reset()

        print(
            "\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."
        )
        print("-" * 88)

        print(
```

```
            "At the start, the load balancer endpoint returns recommendations and
reports that all targets are healthy."
        )
        self.demo_choices()

        print(
            f"The web service running on the EC2 instances gets recommendations
by querying a DynamoDB table.\n"
            f"The table name is contained in a Systems Manager parameter named
'{self.param_helper.table}'.\n"
            f"To simulate a failure of the recommendation service, let's set this
parameter to name a non-existent table.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\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."
        )
        self.demo_choices()

        print(
            f"Instead of failing when the recommendation service fails, the web
service can return a static response.\n"
            f"While this is not a perfect solution, it presents the customer with
a somewhat better experience than failure.\n"
        )
        self.param_helper.put(self.param_helper.failure_response, "static")
        print(
            f"\nNow, sending a GET request to the load balancer endpoint returns
a static response.\n"
            f"The service still reports as healthy because health checks are
still shallow.\n"
        )
        self.demo_choices()

        print("Let's reinstate the recommendation service.\n")
        self.param_helper.put(self.param_helper.table,
self.recommendation.table_name)
        print(
            "\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"
```

```
        )
        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)
        print(
            f"\nReplacing the profile for instance {bad_instance_id} with a
profile that contains\n"
            f"bad credentials...\n"
        )
        self.autoscaler.replace_instance_profile(
            bad_instance_id,
            self.autoscaler.bad_creds_profile_name,
            instance_profile["AssociationId"],
        )
        print(
            "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"
        )
        self.demo_choices()

        print(
            "\nLet's implement a deep health check. For this demo, a deep health
check tests whether\n"
            "the web service can access the DynamoDB table that it depends on for
recommendations. Note that\n"
            "the deep health check is only for ELB routing and not for Auto
Scaling instance health.\n"
            "This kind of deep health check is not recommended for Auto Scaling
instance health, because it\n"
            "risks accidental termination of all instances in the Auto Scaling
group when a dependent service fails.\n"
        )
        print(
            "By implementing deep health checks, the load balancer can detect
when one of the instances is failing\n"
            "and take that instance out of rotation.\n"
```

```
        )
        self.param_helper.put(self.param_helper.health_check, "deep")
        print(
            f"\nNow, checking target health indicates that the instance with bad
credentials ({bad_instance_id})\n"
            f"is unhealthy. Note that it might take a minute or two for the load
balancer to detect the unhealthy \n"
            f"instance. Sending a GET request to the load balancer endpoint
always returns a recommendation, because\n"
            "the load balancer takes unhealthy instances out of its rotation.\n"
        )
        self.demo_choices()

        print(
            "\nBecause the instances in this demo are controlled by an auto
scaler, the simplest way to fix an unhealthy\n"
            "instance is to terminate it and let the auto scaler start a new
instance to replace it.\n"
        )
        self.autoscaler.terminate_instance(bad_instance_id)
        print(
            "\nEven while the instance is terminating and the new instance is
starting, sending a GET\n"
            "request to the web service continues to get a successful
recommendation response because\n"
            "the load balancer routes requests to the healthy instances. After
the replacement instance\n"
            "starts and reports as healthy, it is included in the load balancing
rotation.\n"
            "\nNote that terminating and replacing an instance typically takes
several minutes, during which time you\n"
            "can see the changing health check status until the new instance is
running and healthy.\n"
        )
        self.demo_choices()

        print(
            "\nIf the recommendation service fails now, deep health checks mean
all instances report as unhealthy.\n"
        )
        self.param_helper.put(self.param_helper.table, "this-is-not-a-table")
        print(
            "\nWhen all instances are unhealthy, the load balancer continues to
route requests even to\n"
```

```
            "unhealthy instances, allowing them to fail open and return a static
 response rather than fail\n"
            "closed and report failure to the customer."
        )
        self.demo_choices()
        self.param_helper.reset()

    def destroy(self):
        print(
            "This concludes the demo of how to build and manage a resilient
 service.\n"
            "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 q.ask("Do you want to clean up all demo resources? (y/n) ",
 q.is_yesno):
            self.loadbalancer.delete_load_balancer()
            self.loadbalancer.delete_target_group()
            self.autoscaler.delete_group()
            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,
            )
            self.recommendation.destroy()
        else:
            print(
                "Okay, 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."
            )


def main():
    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="../../../workflows/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()

    print("-" * 88)
    print(
        "Welcome to the demonstration of How to Build and Manage a Resilient
 Service!"
    )
    print("-" * 88)

    prefix = "doc-example-resilience"
    recommendation = RecommendationService.from_client(
        "doc-example-recommendation-service"
    )
    autoscaler = AutoScaler.from_client(prefix)
    loadbalancer = LoadBalancer.from_client(prefix)
    param_helper = ParameterHelper.from_client(recommendation.table_name)
    runner = Runner(
        args.resource_path, recommendation, autoscaler, loadbalancer,
 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()

    print("-" * 88)
    print("Thanks for watching!")
    print("-" * 88)


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 AutoScaler:
    """
    Encapsulates Amazon EC2 Auto Scaling and EC2 management actions.
    """

    def __init__(
        self,
        resource_prefix,
        inst_type,
        ami_param,
        autoscaling_client,
        ec2_client,
        ssm_client,
        iam_client,
    ):
        """
        :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
        self.launch_template_name = f"{resource_prefix}-template"
        self.group_name = f"{resource_prefix}-group"
        self.instance_policy_name = f"{resource_prefix}-pol"
        self.instance_role_name = f"{resource_prefix}-role"
```

```
        self.instance_profile_name = f"{resource_prefix}-prof"
        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"
        self.key_pair_name = f"{resource_prefix}-key-pair"


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from Boto3 clients.

        :param resource_prefix: The prefix for naming AWS resources that are
created by this class.
        """
        as_client = boto3.client("autoscaling")
        ec2_client = boto3.client("ec2")
        ssm_client = boto3.client("ssm")
        iam_client = boto3.client("iam")
        return cls(
            resource_prefix,
            "t3.micro",
            "/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2",
            as_client,
            ec2_client,
            ssm_client,
            iam_client,
        )

    def create_instance_profile(
        self, policy_file, policy_name, role_name, profile_name,
aws_managed_policies=()
    ):
        """
        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",
                }
            ],
        }
        with open(policy_file) as file:
            instance_policy_doc = file.read()

        policy_arn = None
        try:
            pol_response = self.iam_client.create_policy(
                PolicyName=policy_name, PolicyDocument=instance_policy_doc
            )
            policy_arn = pol_response["Policy"]["Arn"]
            log.info("Created policy with ARN %s.", policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Policy %s already exists, nothing to do.", policy_name)
                list_pol_response = self.iam_client.list_policies(Scope="Local")
                for pol in list_pol_response["Policies"]:
                    if pol["PolicyName"] == policy_name:
                        policy_arn = pol["Arn"]
                        break
            if policy_arn is None:
                raise AutoScalerError(f"Couldn't create policy {policy_name}:
{err}")
```

```
        try:
            self.iam_client.create_role(
                RoleName=role_name,
AssumeRolePolicyDocument=json.dumps(assume_role_doc)
            )
            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("Created role %s and attached policy %s.", role_name,
policy_arn)
        except ClientError as err:
            if err.response["Error"]["Code"] == "EntityAlreadyExists":
                log.info("Role %s already exists, nothing to do.", role_name)
            else:
                raise AutoScalerError(f"Couldn't create role {role_name}: {err}")

        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
                )
            else:
```

```
                raise AutoScalerError(
                    f"Couldn't create profile {profile_name} and attach it to
role\n"
                    f"{role_name}: {err}"
                )
        return profile_arn


    def get_instance_profile(self, instance_id):
        """
        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]}]
            )
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instance profile association for instance
{instance_id}: {err}"
            )
        else:
            return response["IamInstanceProfileAssociations"][0]


    def replace_instance_profile(
        self, instance_id, new_instance_profile_name, profile_association_id
    ):
        """
        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 update.
        :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)
            inst_ready = False
            tries = 0
            while not inst_ready:
                if tries % 6 == 0:
                    self.ec2_client.reboot_instances(InstanceIds=[instance_id])
                    log.info(
                        "Rebooting instance %s and waiting for it to to be
ready.",
                        instance_id,
                    )
                tries += 1
                time.sleep(10)
                response = self.ssm_client.describe_instance_information()
                for info in response["InstanceInformationList"]:
                    if info["InstanceId"] == instance_id:
                        inst_ready = True
            self.ssm_client.send_command(
                InstanceIds=[instance_id],
                DocumentName="AWS-RunShellScript",
                Parameters={"commands": ["cd / && sudo python3 server.py 80"]},
            )
            log.info("Restarted the Python web server on instance %s.",
instance_id)
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't replace instance profile for association
{profile_association_id}: {err}"
            )
```

```
    def delete_instance_profile(self, profile_name, role_name):
        """
        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:
            if err.response["Error"]["Code"] == "NoSuchEntity":
                log.info(
                    "Instance profile %s doesn't exist, nothing to do.",
profile_name
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete instance profile {profile_name} or detach "
                    f"policies and delete role {role_name}: {err}"
                )


    def create_key_pair(self, key_pair_name):
        """
```

```
        Creates a new key pair.

        :param key_pair_name: The name of the key pair to create.
        :return: The newly created key pair.
        """
        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:
            raise AutoScalerError(f"Couldn't create key pair {key_pair_name}:
{err}")


    def delete_key_pair(self):
        """
        Deletes a key pair.

        :param key_pair_name: The name of the key pair to delete.
        """
        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:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )
        except FileNotFoundError:
            log.info("Key pair %s doesn't exist, nothing to do.",
self.key_pair_name)
        except PermissionError:
            log.info(
                "Inadequate permissions to delete key pair %s.",
self.key_pair_name
            )
        except Exception as err:
            raise AutoScalerError(
                f"Couldn't delete key pair {self.key_pair_name}: {err}"
            )
```

```
    def create_template(self, server_startup_script_file, instance_policy_file):
        """
        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:
            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,
            )
            with open(server_startup_script_file) as file:
                start_server_script = file.read()
            ami_latest = self.ssm_client.get_parameter(Name=self.ami_param)
            ami_id = ami_latest["Parameter"]["Value"]
            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(
                "Created launch template %s for AMI %s on %s.",
                self.launch_template_name,
                ami_id,
                self.inst_type,
            )
        except ClientError as err:
            if (
                err.response["Error"]["Code"]
                == "InvalidLaunchTemplateName.AlreadyExistsException"
            ):
                log.info(
                    "Launch template %s already exists, nothing to do.",
                    self.launch_template_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create launch template
{self.launch_template_name}: {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,
```

```
                )
            else:
                raise AutoScalerError(
                    f"Couldn't delete launch template
{self.launch_template_name}: {err}."
                )


    def get_availability_zones(self):
        """
        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"]]
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get availability zones: {err}.")
        else:
            return zones


    def create_group(self, group_size):
        """
        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.
        """
        zones = []
        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(
                "Created EC2 Auto Scaling group %s with availability zones %s.",
                self.launch_template_name,
                zones,
            )
        except ClientError as err:
            if err.response["Error"]["Code"] == "AlreadyExists":
                log.info(
                    "EC2 Auto Scaling group %s already exists, nothing to do.",
                    self.group_name,
                )
            else:
                raise AutoScalerError(
                    f"Couldn't create EC2 Auto Scaling group {self.group_name}:
{err}"
                )
        return zones


    def get_instances(self):
        """
        Gets data about the instances in the EC2 Auto Scaling group.

        :return: Data about the instances.
        """
        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"]
            ]
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't get instances for Auto Scaling group
{self.group_name}: {err}"
            )
        else:
            return instance_ids
```

```
    def terminate_instance(self, instance_id):
        """
        Terminates and instances 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.
        """
        try:
            self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                InstanceId=instance_id, ShouldDecrementDesiredCapacity=False
            )
            log.info("Terminated instance %s.", instance_id)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't terminate instance {instance_id}:
{err}")

    def attach_load_balancer_target_group(self, lb_target_group):
        """
        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.

        :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:
            raise AutoScalerError(
                f"Couldn't attach load balancer target group
{lb_target_group['TargetGroupName']}\n"
                f"to auto scaling group {self.group_name}"
            )
```

```
   def _try_terminate_instance(self, inst_id):
       stopping = False
       log.info(f"Stopping {inst_id}.")
       while not stopping:
           try:
               self.autoscaling_client.terminate_instance_in_auto_scaling_group(
                   InstanceId=inst_id, ShouldDecrementDesiredCapacity=True
               )
               stopping = True
           except ClientError as err:
               if err.response["Error"]["Code"] == "ScalingActivityInProgress":
                   log.info("Scaling activity in progress for %s. Waiting...",
inst_id)
                   time.sleep(10)
               else:
                   raise AutoScalerError(f"Couldn't stop instance {inst_id}:
{err}.")

   def _try_delete_group(self):
       """
       Tries to delete the EC2 Auto Scaling group. If the group is in use or in
progress,
       the function waits and retries until the group is successfully deleted.
       """
       stopped = False
       while not stopped:
           try:
               self.autoscaling_client.delete_auto_scaling_group(
                   AutoScalingGroupName=self.group_name
               )
               stopped = True
               log.info("Deleted EC2 Auto Scaling group %s.", self.group_name)
           except ClientError as err:
               if (
                   err.response["Error"]["Code"] == "ResourceInUse"
                   or err.response["Error"]["Code"] ==
"ScalingActivityInProgress"
               ):
                   log.info(
                       "Some instances are still running. Waiting for them to
stop..."
                   )
```

```
                    time.sleep(10)
                else:
                    raise AutoScalerError(
                        f"Couldn't delete group {self.group_name}: {err}."
                    )

    def delete_group(self):
        """
        Terminates all instances in the group, deletes the EC2 Auto Scaling
 group.
        """
        try:
            response = self.autoscaling_client.describe_auto_scaling_groups(
                AutoScalingGroupNames=[self.group_name]
            )
            groups = response.get("AutoScalingGroups", [])
            if len(groups) > 0:
                self.autoscaling_client.update_auto_scaling_group(
                    AutoScalingGroupName=self.group_name, MinSize=0
                )
                instance_ids = [inst["InstanceId"] for inst in groups[0]
["Instances"]]
                for inst_id in instance_ids:
                    self._try_terminate_instance(inst_id)
                self._try_delete_group()
            else:
                log.info("No groups found named %s, nothing to do.",
 self.group_name)
        except ClientError as err:
            raise AutoScalerError(f"Couldn't delete group {self.group_name}:
 {err}.")


    def get_default_vpc(self):
        """
        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:
```

```
            raise AutoScalerError(f"Couldn't get default VPC: {err}")
        else:
            return response["Vpcs"][0]


    def verify_inbound_port(self, vpc, port, ip_address):
        """
        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 specific 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("Found default security group %s.", sec_group["GroupId"])
            for ip_perm in sec_group["IpPermissions"]:
                if ip_perm.get("FromPort", 0) == port:
                    log.info("Found inbound rule: %s", 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(
                            "The inbound rule does not appear to be open to
either this computer's IP\n"
                            "address of %s, to all IP addresses (0.0.0.0/0), or
to a prefix list ID.",
                            ip_address,
                        )
                    else:
                        break
        except ClientError as err:
            raise AutoScalerError(
                f"Couldn't verify inbound rule for port {port} for VPC
{vpc['VpcId']}: {err}"
            )
        else:
            return sec_group, port_is_open


    def open_inbound_port(self, sec_group_id, port, ip_address):
        """
        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:
```

```
            raise AutoScalerError(
                f"Couldn't authorize ingress to {sec_group_id} on port {port}
 from {ip_address}: {err}"
            )


    def get_subnets(self, vpc_id, zones):
        """
        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.
        """
        try:
            response = self.ec2_client.describe_subnets(
                Filters=[
                    {"Name": "vpc-id", "Values": [vpc_id]},
                    {"Name": "availability-zone", "Values": zones},
                    {"Name": "default-for-az", "Values": ["true"]},
                ]
            )
            subnets = response["Subnets"]
            log.info("Found %s subnets for the specified zones.", len(subnets))
        except ClientError as err:
            raise AutoScalerError(f"Couldn't get subnets: {err}")
        else:
            return subnets
```

Crie uma classe que envolva ações do Elastic Load Balancing.

```
class LoadBalancer:
    """Encapsulates Elastic Load Balancing (ELB) actions."""

    def __init__(self, target_group_name, load_balancer_name, elb_client):
        """
        :param target_group_name: The name of the target group associated with
 the load balancer.
```

```
        :param load_balancer_name: The name of the load balancer.
        :param elb_client: A Boto3 Elastic Load Balancing client.
        """
        self.target_group_name = target_group_name
        self.load_balancer_name = load_balancer_name
        self.elb_client = elb_client
        self._endpoint = None


    @classmethod
    def from_client(cls, resource_prefix):
        """
        Creates this class from a Boto3 client.

        :param resource_prefix: The prefix to give to AWS resources created by
this class.
        """
        elb_client = boto3.client("elbv2")
        return cls(f"{resource_prefix}-tg", f"{resource_prefix}-lb", elb_client)

    def endpoint(self):
        """
        Gets the HTTP endpoint of the load balancer.

        :return: The endpoint.
        """
        if self._endpoint is None:
            try:
                response = self.elb_client.describe_load_balancers(
                    Names=[self.load_balancer_name]
                )
                self._endpoint = response["LoadBalancers"][0]["DNSName"]
            except ClientError as err:
                raise LoadBalancerError(
                    f"Couldn't get the endpoint for load balancer
{self.load_balancer_name}: {err}"
                )
        return self._endpoint


    def create_target_group(self, protocol, port, vpc_id):
        """
        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.

        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 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=self.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("Created load balancing target group %s.",
self.target_group_name)
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't create load balancing target group
{self.target_group_name}: {err}"
            )
        else:
            return target_group


    def delete_target_group(self):
        """
        Deletes the target group.
        """
        done = False
```

```
        while not done:
            try:
                response = self.elb_client.describe_target_groups(
                    Names=[self.target_group_name]
                )
                tg_arn = response["TargetGroups"][0]["TargetGroupArn"]
                self.elb_client.delete_target_group(TargetGroupArn=tg_arn)
                log.info(
                    "Deleted load balancing target group %s.",
self.target_group_name
                )
                done = True
            except ClientError as err:
                if err.response["Error"]["Code"] == "TargetGroupNotFound":
                    log.info(
                        "Load balancer target group %s not found, nothing to
do.",
                        self.target_group_name,
                    )
                    done = True
                elif err.response["Error"]["Code"] == "ResourceInUse":
                    log.info(
                        "Target group not yet released from load balancer,
waiting..."
                    )
                    time.sleep(10)
                else:
                    raise LoadBalancerError(
                        f"Couldn't delete load balancing target group
{self.target_group_name}: {err}"
                    )


    def create_load_balancer(self, subnet_ids, target_group):
        """
        Creates an Elastic Load Balancing load balancer that uses the specified
subnets
        and forwards requests to the specified target group.

        :param subnet_ids: A list of subnets to associate with the load balancer.
        :param target_group: An existing target group that is added as a listener
to the
                             load balancer.
        :return: Data about the newly created load balancer.
```

```
        """
        try:
            response = self.elb_client.create_load_balancer(
                Name=self.load_balancer_name, Subnets=subnet_ids
            )
            load_balancer = response["LoadBalancers"][0]
            log.info("Created load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancer_available")
            log.info("Waiting for load balancer to be available...")
            waiter.wait(Names=[self.load_balancer_name])
            log.info("Load balancer is available!")
            self.elb_client.create_listener(
                LoadBalancerArn=load_balancer["LoadBalancerArn"],
                Protocol=target_group["Protocol"],
                Port=target_group["Port"],
                DefaultActions=[
                    {
                        "Type": "forward",
                        "TargetGroupArn": target_group["TargetGroupArn"],
                    }
                ],
            )
            log.info(
                "Created listener to forward traffic from load balancer %s to
target group %s.",
                self.load_balancer_name,
                target_group["TargetGroupName"],
            )
        except ClientError as err:
            raise LoadBalancerError(
                f"Failed to create load balancer {self.load_balancer_name}"
                f"and add a listener for target group
{target_group['TargetGroupName']}: {err}"
            )
        else:
            self._endpoint = load_balancer["DNSName"]
            return load_balancer


    def delete_load_balancer(self):
        """
        Deletes a load balancer.
        """
        try:
```

```
            response = self.elb_client.describe_load_balancers(
                Names=[self.load_balancer_name]
            )
            lb_arn = response["LoadBalancers"][0]["LoadBalancerArn"]
            self.elb_client.delete_load_balancer(LoadBalancerArn=lb_arn)
            log.info("Deleted load balancer %s.", self.load_balancer_name)
            waiter = self.elb_client.get_waiter("load_balancers_deleted")
            log.info("Waiting for load balancer to be deleted...")
            waiter.wait(Names=[self.load_balancer_name])
        except ClientError as err:
            if err.response["Error"]["Code"] == "LoadBalancerNotFound":
                log.info(
                    "Load balancer %s does not exist, nothing to do.",
                    self.load_balancer_name,
                )
            else:
                raise LoadBalancerError(
                    f"Couldn't delete load balancer {self.load_balancer_name}:
{err}"
                )


    def verify_load_balancer_endpoint(self):
        """
        Verify this computer can successfully send a GET request to the load
balancer endpoint.
        """
        success = False
        retries = 3
        while not success and retries > 0:
            try:
                lb_response = requests.get(f"http://{self.endpoint()}")
                log.info(
                    "Got response %s from load balancer endpoint.",
                    lb_response.status_code,
                )
                if lb_response.status_code == 200:
                    success = True
                else:
                    retries = 0
            except requests.exceptions.ConnectionError:
                log.info(
                    "Got connection error from load balancer endpoint,
retrying..."
```

```
                )
                retries -= 1
                time.sleep(10)
        return success

    def check_target_health(self):
        """
        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=[self.target_group_name]
            )
            health_response = self.elb_client.describe_target_health(
                TargetGroupArn=tg_response["TargetGroups"][0]["TargetGroupArn"]
            )
        except ClientError as err:
            raise LoadBalancerError(
                f"Couldn't check health of {self.target_group_name} targets:
 {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, dynamodb_client):
        """
        :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

    @classmethod
    def from_client(cls, table_name):
        """
        Creates this class from a Boto3 client.

        :param table_name: The name of the DynamoDB recommendations table.
        """
        ddb_client = boto3.client("dynamodb")
        return cls(table_name, ddb_client)

    def create(self):
        """
        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.

        :return: Data about the newly created table.
        """
        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 do.", self.table_name)
            else:
                raise RecommendationServiceError(
                    self.table_name, f"ClientError when creating table: {err}."
                )
        else:
            return response

    def populate(self, data_file):
        """
        Populates the recommendations table from a JSON file.

        :param data_file: The path to the data file.
        """
        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):
        """
        Deletes the recommendations table.
        """
        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 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 = "doc-example-resilient-architecture-table"
    failure_response = "doc-example-resilient-architecture-failure-response"
    health_check = "doc-example-resilient-architecture-health-check"

    def __init__(self, table_name, ssm_client):
        """
        :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

    @classmethod
    def from_client(cls, table_name):
        ssm_client = boto3.client("ssm")
        return cls(table_name, ssm_client)

    def reset(self):
        """
        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, value):
        """
        Sets the value of a named Systems Manager parameter.

        :param name: The name of the parameter.
        :param value: The new value of the parameter.
        """
        try:
            self.ssm_client.put_parameter(
                Name=name, Value=value, Overwrite=True, Type="String"
            )
            log.info("Setting demo parameter %s to '%s'.", name, value)
        except ClientError as err:
            raise ParameterHelperError(
                f"Couldn't set parameter {name} to {value}: {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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Criar um grupo do IAM e adicionar um usuário ao grupo usando um AWS SDK

O exemplo de código a seguir mostra como:

- Criar um grupo e conceda permissões de acesso completo ao Amazon S3 a ele.
- Criar um novo usuário sem permissões para acessar o Amazon S3.
- Adicione o usuário ao grupo e mostre que agora ele tem permissões para o Amazon S3. Em seguida, limpe os recursos.

.NET

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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>
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
```

```
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <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 group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }


    /// <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 group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

```
    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        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>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</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> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };
```

```
        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <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 IAMGroups;

public class IAMGroups
{
```

```
    private static ILogger logger = null!;

    // Represents JSON code for AWS full access policy for Amazon Simple
    // Storage Service (Amazon S3).
    private const string S3FullAccessPolicyDocument = "{" +
        " \"Statement\" : [{" +
            " \"Action\" : [\"s3:*\"]," +
            " \"Effect\" : \"Allow\"," +
            " \"Resource\" : \"*\"" +
        "}]" +
    "}";


    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<IAMGroups>();

        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();

        var groupUserName = configuration["GroupUserName"];
        var groupName = configuration["GroupName"];
        var groupPolicyName = configuration["GroupPolicyName"];
        var groupBucketName = configuration["GroupBucketName"];
```

```
        var wrapper = host.Services.GetRequiredService<IAMWrapper>();
        var uiWrapper = host.Services.GetRequiredService<UIWrapper>();

        uiWrapper.DisplayGroupsOverview();
        uiWrapper.PressEnter();

        // Create an IAM group.
        uiWrapper.DisplayTitle("Create IAM group");
        Console.WriteLine("Let's begin by creating a new IAM group.");
        var group = await wrapper.CreateGroupAsync(groupName);

        // Add an inline IAM policy to the group.
        uiWrapper.DisplayTitle("Add policy to group");
        Console.WriteLine("Add an inline policy to the group that allows members
to have full access to");
        Console.WriteLine("Amazon Simple Storage Service (Amazon S3) buckets.");

        await wrapper.PutGroupPolicyAsync(group.GroupName, groupPolicyName,
S3FullAccessPolicyDocument);

        uiWrapper.PressEnter();

        // Now create a new user.
        uiWrapper.DisplayTitle("Create an IAM user");
        Console.WriteLine("Now let's create a new IAM user.");
        var groupUser = await wrapper.CreateUserAsync(groupUserName);

        // Add the new user to the group.
        uiWrapper.DisplayTitle("Add the user to the group");
        Console.WriteLine("Adding the user to the group, which will give the user
the same permissions as the group.");
        await wrapper.AddUserToGroupAsync(groupUser.UserName, group.GroupName);

        Console.WriteLine($"User, {groupUser.UserName}, has been added to the
group, {group.GroupName}.");
        uiWrapper.PressEnter();

        Console.WriteLine("Now that we have created a user, and added the user to
the group, let's create an IAM access key.");

        // Create access and secret keys for the user.
        var accessKey = await wrapper.CreateAccessKeyAsync(groupUserName);
        Console.WriteLine("Key created.");
```

```
        uiWrapper.WaitABit(15, "Waiting for the access key to be ready for
use.");

        uiWrapper.DisplayTitle("List buckets");
        Console.WriteLine("To prove that the user has access to Amazon S3, list
the S3 buckets for the account.");

        var s3Client = new AmazonS3Client(accessKey.AccessKeyId,
accessKey.SecretAccessKey);
        var stsClient = new
AmazonSecurityTokenServiceClient(accessKey.AccessKeyId,
accessKey.SecretAccessKey);

        var s3Wrapper = new S3Wrapper(s3Client, stsClient);

        var buckets = await s3Wrapper.ListMyBucketsAsync();

        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName}\tcreated on:
{bucket.CreationDate}");
            });
        }

        // Show that the user also has write access to Amazon S3 by creating
        // a new bucket.
        uiWrapper.DisplayTitle("Create a bucket");
        Console.WriteLine("Since group members have full access to Amazon S3,
let's create a bucket.");
        var success = await s3Wrapper.PutBucketAsync(groupBucketName);

        if (success)
        {
            Console.WriteLine($"Successfully created the bucket:
{groupBucketName}.");
        }

        uiWrapper.PressEnter();

        Console.WriteLine("Let's list the user's S3 buckets again to show the new
bucket.");
```

```
        buckets = await s3Wrapper.ListMyBucketsAsync();

        if (buckets is not null)
        {
            buckets.ForEach(bucket =>
            {
                Console.WriteLine($"{bucket.BucketName}\tcreated on:
 {bucket.CreationDate}");
            });
        }

        uiWrapper.PressEnter();

        uiWrapper.DisplayTitle("Clean up resources");
        Console.WriteLine("First delete the bucket we created.");
        await s3Wrapper.DeleteBucketAsync(groupBucketName);

        Console.WriteLine($"Now remove the user, {groupUserName}, from the group,
 {groupName}.");
        await wrapper.RemoveUserFromGroupAsync(groupUserName, groupName);

        Console.WriteLine("Delete the user's access key.");
        await wrapper.DeleteAccessKeyAsync(accessKey.AccessKeyId, groupUserName);

        // Now we can safely delete the user.
        Console.WriteLine("Now we can delete the user.");
        await wrapper.DeleteUserAsync(groupUserName);

        uiWrapper.PressEnter();

        Console.WriteLine("Now we will delete the IAM policy attached to the
 group.");
        await wrapper.DeleteGroupPolicyAsync(groupName, groupPolicyName);

        Console.WriteLine("Now we delete the IAM group.");
        await wrapper.DeleteGroupAsync(groupName);

        uiWrapper.PressEnter();

        Console.WriteLine("The IAM groups demo has completed.");

        uiWrapper.PressEnter();
    }
}
```

```
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 for .NET.
  - AddUserToGroup
  - AttachRolePolicy
  - CreateAccessKey
  - CreateGroup
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeleteGroup
  - DeleteGroupPolicy
  - DeleteUser
  - PutGroupPolicy
  - RemoveUserFromGroup

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Criar um usuário do IAM e assumir uma função com o AWS STS usando 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 AWS IAM Identity Center.

- 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

AWS SDK for .NET

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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>
```

```
    /// Add an existing IAM user to an existing IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to add.</param>
    /// <param name="groupName">The name of the group to add the user to.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> AddUserToGroupAsync(string userName, string
 groupName)
    {
        var response = await _IAMService.AddUserToGroupAsync(new
 AddUserToGroupRequest
        {
            GroupName = groupName,
            UserName = userName,
        });

        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <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 group.
    /// </summary>
    /// <param name="groupName">The name to give the IAM group.</param>
    /// <returns>The IAM group that was created.</returns>
    public async Task<Group> CreateGroupAsync(string groupName)
    {
        var response = await _IAMService.CreateGroupAsync(new CreateGroupRequest
 { GroupName = groupName });
        return response.Group;
    }


    /// <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 group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupAsync(string groupName)
    {
```

```
        var response = await _IAMService.DeleteGroupAsync(new DeleteGroupRequest
{ GroupName = groupName });
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Delete an IAM policy associated with an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group associated with the
    /// policy.</param>
    /// <param name="policyName">The name of the policy to delete.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> DeleteGroupPolicyAsync(string groupName, string
policyName)
    {
        var request = new DeleteGroupPolicyRequest()
        {
            GroupName = groupName,
            PolicyName = policyName,
        };

        var response = await _IAMService.DeleteGroupPolicyAsync(request);
        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>
    /// Remove a user from an IAM group.
    /// </summary>
    /// <param name="userName">The username of the user to remove.</param>
    /// <param name="groupName">The name of the IAM group to remove the user
from.</param>
    /// <returns>A Boolean value indicating the success of the action.</returns>
    public async Task<bool> RemoveUserFromGroupAsync(string userName, string
groupName)
    {
        // Remove the user from the group.
        var removeUserRequest = new RemoveUserFromGroupRequest()
        {
            UserName = userName,
            GroupName = groupName,
        };

        var response = await
_IAMService.RemoveUserFromGroupAsync(removeUserRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }


    /// <summary>
    /// Add or update an inline policy document that is embedded in an IAM group.
    /// </summary>
    /// <param name="groupName">The name of the IAM group.</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> PutGroupPolicyAsync(string groupName, string
policyName, string policyDocument)
    {
        var request = new PutGroupPolicyRequest
        {
            GroupName = groupName,
```

```
            PolicyName = policyName,
            PolicyDocument = policyDocument
        };

        var response = await _IAMService.PutGroupPolicyAsync(request);
        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }


    /// <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 for .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 Repositório de exemplos de código da AWS.

```
###############################################################################
# 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 WS 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 WS 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 WS 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 WS 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 WS 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
}
```

- Para obter detalhes da API, consulte os 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 veja como configurar e executar no repositório de exemplos de código da AWS.

```
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 for 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 Repositório de exemplos de código da AWS.

Execute um cenário interativo em um prompt de comando.

```
// 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() {
 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()
 accessKey := scenario.CreateAccessKey(user)
 role := scenario.CreateRoleAndPolicies(user)
 noPermsConfig := scenario.ListBucketsWithoutPermissions(accessKey)
 scenario.ListBucketsWithAssumedRole(noPermsConfig, role)
 scenario.Cleanup(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() *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(userName)
 if err != nil {
  panic(err)
 }
 if user == nil {
  user, err = scenario.userWrapper.CreateUser(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(user *types.User)
 *types.AccessKey {
 accessKey, err := scenario.userWrapper.CreateAccessKeyPair(*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(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(scenario.helper.GetName(), *user.Arn)
 if err != nil {panic(err)}
 log.Printf("Created role %v.\n", *listBucketsRole.RoleName)
 listBucketsPolicy, err := scenario.policyWrapper.CreatePolicy(
  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(*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(*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(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(context.TODO(),
  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(context.TODO(), &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(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(context.TODO(),
 &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(context.TODO(),
  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(context.TODO(), &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(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(*role.RoleName)
   if err != nil {panic(err)}
   for _, policy := range policies {
    err = scenario.roleWrapper.DetachRolePolicy(*role.RoleName,
 *policy.PolicyArn)
    if err != nil {panic(err)}
    err = scenario.policyWrapper.DeletePolicy(*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(*role.RoleName)
    if err != nil {panic(err)}
    log.Printf("Deleted role %v.\n", *role.RoleName)

    userPols, err := scenario.userWrapper.ListUserPolicies(*user.UserName)
    if err != nil {panic(err)}
    for _, userPol := range userPols {
     err = scenario.userWrapper.DeleteUserPolicy(*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(*user.UserName)
    if err != nil {panic(err)}
    for _, key := range keys {
     err = scenario.userWrapper.DeleteAccessKey(*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(*user.UserName)
    if err != nil {panic(err)}
    log.Printf("Deleted user %v.\n", *user.UserName)
    log.Println(strings.Repeat("-", 88))
 }

}
```

Defina um struct que encapsule as ações de conta.

```
// 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() (*types.PasswordPolicy,
 error) {
 var pwPolicy *types.PasswordPolicy
 result, err := wrapper.IamClient.GetAccountPasswordPolicy(context.TODO(),
  &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() ([]types.SAMLProviderListEntry,
 error) {
 var providers []types.SAMLProviderListEntry
 result, err := wrapper.IamClient.ListSAMLProviders(context.TODO(),
 &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.

```
// 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"`
}



// 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(maxPolicies int32) ([]types.Policy,
 error) {
 var policies []types.Policy
 result, err := wrapper.IamClient.ListPolicies(context.TODO(),
 &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
}



// 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(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(context.TODO(),
 &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(policyArn string) (*types.Policy, error) {
 var policy *types.Policy
 result, err := wrapper.IamClient.GetPolicy(context.TODO(), &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(policyArn string) error {
 _, err := wrapper.IamClient.DeletePolicy(context.TODO(), &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.

```
// 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(maxRoles int32) ([]types.Role, error) {
 var roles []types.Role
 result, err := wrapper.IamClient.ListRoles(context.TODO(),
  &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(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(context.TODO(),
 &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(roleName string) (*types.Role, error) {
 var role *types.Role
```

```
 result, err := wrapper.IamClient.GetRole(context.TODO(),
  &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(serviceName string,
 description string) (*types.Role, error) {
 var role *types.Role
 result, err := wrapper.IamClient.CreateServiceLinkedRole(context.TODO(),
 &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(roleName string) error {
 _, err := wrapper.IamClient.DeleteServiceLinkedRole(context.TODO(),
 &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(policyArn string, roleName string)
 error {
 _, err := wrapper.IamClient.AttachRolePolicy(context.TODO(),
 &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(roleName string)
 ([]types.AttachedPolicy, error) {
 var policies []types.AttachedPolicy
 result, err := wrapper.IamClient.ListAttachedRolePolicies(context.TODO(),
 &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(roleName string, policyArn string)
 error {
```

```
 _, err := wrapper.IamClient.DetachRolePolicy(context.TODO(),
 &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(roleName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListRolePolicies(context.TODO(),
 &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(roleName string) error {
 _, err := wrapper.IamClient.DeleteRole(context.TODO(), &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.

```
// 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(maxUsers int32) ([]types.User, error) {
 var users []types.User
 result, err := wrapper.IamClient.ListUsers(context.TODO(), &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(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.GetUser(context.TODO(), &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(userName string) (*types.User, error) {
 var user *types.User
 result, err := wrapper.IamClient.CreateUser(context.TODO(),
 &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(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(context.TODO(),
 &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(userName string) ([]string, error) {
 var policies []string
 result, err := wrapper.IamClient.ListUserPolicies(context.TODO(),
 &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(userName string, policyName string)
 error {
```

```
 _, err := wrapper.IamClient.DeleteUserPolicy(context.TODO(),
 &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(userName string) error {
 _, err := wrapper.IamClient.DeleteUser(context.TODO(), &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(userName string)
 (*types.AccessKey, error) {
 var key *types.AccessKey
 result, err := wrapper.IamClient.CreateAccessKey(context.TODO(),
 &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(userName string, keyId string) error {
 _, err := wrapper.IamClient.DeleteAccessKey(context.TODO(),
 &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(userName string)
 ([]types.AccessKeyMetadata, error) {
 var keys []types.AccessKeyMetadata
 result, err := wrapper.IamClient.ListAccessKeys(context.TODO(),
 &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 for 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 veja como configurar e executar no repositório de exemplos de código da AWS.

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 Repositório de exemplos de código da AWS.

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,
  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";

// Set the parameters.
const iamClient = new IAMClient({});
const userName = "test_name";
const policyName = "test_policy";
```

```
const roleName = "test_role";

export const main = async () => {
  // Create a user. The user has no permissions by default.
  const { User } = await iamClient.send(
    new CreateUserCommand({ UserName: userName }),
  );

  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: 60 }, () =>
  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 for JavaScript.
  - AttachRolePolicy
  - CreateAccessKey
  - CreatePolicy
  - CreateRole
  - CreateUser
  - DeleteAccessKey
  - DeletePolicy
  - DeleteRole

- DeleteUser
- DeleteUserPolicy
- DetachRolePolicy
- PutUserPolicy

Kotlin

SDK for Kotlin

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e veja como configurar e executar no repositório de exemplos de código da AWS.

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: String = "{" +
        "  \"Version\": \"2012-10-17\"," +
```

```
        "  \"Statement\": [" +
        "    {" +
        "        \"Effect\": \"Allow\"," +
        "        \"Action\": [" +
        "            \"s3:*\"" +
        "       ]," +
        "       \"Resource\": \"*\"" +
        "    }" +
        "   ]" +
        "}"

    val request = CreatePolicyRequest {
        policyName = policyNameVal
        policyDocument = policyDocumentValue
    }

    IamClient { 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 { 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 {
        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 {
        credentialsProvider = staticCredentials
        region = "us-east-1"
    }

    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 { 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 { 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 Repositório de exemplos de código da AWS.

```
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 for 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 Repositório de exemplos de código da AWS.

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 Repositório de exemplos de código da AWS.

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 for 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 Repositório de exemplos de código da AWS.

```
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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 AWS IAM Identity Center.

- 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 Repositório de exemplos de código da AWS.

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:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 AWS IAM Identity Center.

- 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 Repositório de exemplos de código da AWS.

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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 exemplo de código a seguir 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 Repositório de exemplos de código da AWS.

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:::my-bucket/*' to allow actions on all
 objects
                         in the bucket named 'my-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:::made-up-bucket-name"
    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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 exemplo de código a seguir 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 Repositório de exemplos de código da AWS.

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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 exemplo de código a seguir 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 Repositório de exemplos de código da AWS.

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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 exemplo de código a seguir 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 Repositório de exemplos de código da AWS.

```
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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 exemplo de código a seguir 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 Repositório de exemplos de código da AWS.

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:::BUCKETNAME/*")

 .addResource("arn:aws:s3:::BUCKETNAME")
                                                .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:::DOC-
EXAMPLE-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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código

- Ações para o AWS STS usando AWS SDKs
  - Usar AssumeRole com o AWS SDK ou a CLI
  - Usar AssumeRoleWithWebIdentity com o AWS SDK ou a CLI
  - Usar DecodeAuthorizationMessage com o AWS SDK ou a CLI
  - Usar GetFederationToken com o AWS SDK ou a 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
  - Crie uma URL com o AWS STS para usuários federados usando um AWS SDK
  - Obtenha um token de sessão que exija um token de MFA com o AWS STS usando um AWS SDK

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. Esses trechos chamam a API do AWS STS e são trechos de código de programas maiores que devem ser executados no contexto. Cada exemplo inclui um link para o GitHub, onde você pode encontrar instruções para configurar e executar o código.

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para obter uma lista completa, consulte a Referência de API do AWS Security Token Service (AWS STS).

Exemplos

- Usar AssumeRole com o AWS SDK ou a CLI
- Usar AssumeRoleWithWebIdentity com o AWS SDK ou a CLI
- Usar DecodeAuthorizationMessage com o AWS SDK ou a CLI
- Usar GetFederationToken com o AWS SDK ou a 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

AWS SDK for .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>

        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);

            // 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}");
        }
    }
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API AWS SDK for .NET.

Bash

AWS CLI com script Bash

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
###############################################################################
# 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
        ;;
      \?)
        ech o"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
}
```

- Para obter detalhes da API, consulte 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();
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API AWS SDK for 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
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, rn: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);
        }
    }
}
```

- Para obter detalhes da API, consulte 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);
  }
};
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API AWS SDK for 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);
    }
  });
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API AWS SDK for JavaScript.

PowerShell

Tools for PowerShell

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 da API, consulte AssumeRole na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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)
```

- Para obter detalhes da API, consulte 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 Repositório de exemplos de código da AWS.

```
 # 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
```

- Para obter detalhes da API, consulte AssumeRole na Referência da API AWS SDK for Ruby.

Rust

SDK para Rust

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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),
    }
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência do AWS SDK para API Rust.

Swift

SDK para Swift

> **Note**
>
> Esta é a documentação de pré-lançamento de um SDK na versão de visualização. Está sujeita a alteração.

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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 {
        throw error
    }
}
```

- Para obter detalhes da API, consulte AssumeRole na Referência do AWS SDK para API Swift.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`AssumeRoleWithWebIdentity`** com o AWS SDK ou a 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"
```

Saída:

```
{
```

```
    "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.

- Para obter detalhes da API, consulte AssumeRoleWithWebIdentity na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte AssumeRoleWithWebIdentity na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DecodeAuthorizationMessage`** com o AWS SDK ou a 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte DecodeAuthorizationMessage na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte DecodeAuthorizationMessage na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetFederationToken`** com o AWS SDK ou a 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": "*"
        }
    ]
}
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetFederationToken na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetFederationToken na Referência de Cmdlet do AWS Tools for PowerShell.

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 no 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
```

Saída:

```
{
    "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.

- Para obter detalhes da API, consulte GetSessionToken na Referência de comandos da AWS CLI.

PowerShell

Tools for PowerShell

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 da API, consulte GetSessionToken na Referência de Cmdlet do AWS Tools for PowerShell.

Python

SDK para Python (Boto3).

> **Note**
>
> Há mais no GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

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)
```

- Para obter detalhes da API, consulte GetSessionToken na Referência da API do AWS SDK for Python (Boto3).

Para ver uma lista completa dos Guias do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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. Cada exemplo inclui um link para o GitHub, em que é possível encontrar instruções sobre como configurar e executar o código.

Exemplos

- Assumir um perfil do IAM que exija um token de MFA com o AWS STS usando um AWS SDK
- Crie uma URL com o AWS STS para usuários federados usando um AWS SDK
- Obtenha um token de sessão que exija um token de MFA com o AWS STS usando um AWS SDK

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 AWS IAM Identity Center.

- 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 Repositório de exemplos de código da AWS.

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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 exemplo de código a seguir 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 Repositório de exemplos de código da AWS.

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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 AWS IAM Identity Center.

- 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 Repositório de exemplos de código da AWS.

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 do desenvolvedor de SDK da AWS e exemplos de código, consulte Usar o IAM com um 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 para com a nuvem na AWS é a nossa maior prioridade. Como cliente da AWS, você se beneficiará de datacenters 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 produtos da AWS na Nuvem AWS. A AWS também fornece serviços que podem ser usados com segurança. Auditores de terceiros 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.

Índice

Credenciais de segurança da AWS

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.

Há tipos diferentes de usuários na AWS. Todos os usuários da AWS têm credenciais de segurança. Há o proprietário da conta (usuário raiz), usuários no Centro de Identidade do AWS IAM, os usuários federados e os usuários do 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.

Perfis do IAM, usuários no Centro de Identidade do AWS IAM e usuários federados 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.
- Você pode 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. Você pode 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.
- 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 usuários federados em vez das credenciais de longo prazo fornecidas por usuários do IAM e chaves de acesso.

Identidade federada

As identidades federadas são usuários com identidades externas que recebem credenciais temporárias da AWS que podem ser usadas para acessar recursos seguros da Conta da AWS. As identidades externas podem vir de um armazenamento de identidades corporativas (como LDAP ou Windows Active Directory) ou de terceiros (como Login with Amazon, Facebook ou Google). As identidades federadas não fazem login no AWS Management Console ou no portal de acesso da AWS.

Para permitir que identidades federadas façam login na AWS, é necessário criar um URL personalizado que contenha `https://signin.aws.amazon.com/federation`. Para obter mais informações, consulte Habilitar o acesso do agente de identidades personalizado ao console da AWS.

Para obter mais informações sobre identidades federadas, consulte Provedores de identidade e federação.

Autenticação multifator (MFA)

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 Uso de autenticação multifator (MFA) na AWS.

Quando você ativa o MFA e faz login em sua Conta da AWS, são solicitadas suas credenciais de login, além de uma resposta gerada por um dispositivo com MFA, como um código, um toque ou uma verificação biométrica. Quando você adiciona o MFA, as configurações e os recursos de sua Conta da AWS ficam mais protegidos.

Por padrão, o MFA não é ativado. Você pode ativar e gerenciar dispositivos com MFA para o Usuário raiz da conta da AWS acessando a página Credenciais de segurança ou o painel do IAM no AWS Management Console. Para obter mais informações sobre como ativar a MFA para usuários do IAM, consulte Habilitar dispositivos com MFA para usuários na AWS.

Para obter mais informações sobre como fazer login com dispositivos com autenticação multifator (MFA), consulte Uso de dispositivos com MFA com sua página de login do IAM.

Acesso programático

Você fornece suas chaves de acesso da AWS para fazer chamadas programáticas para a AWS ou usar a AWS Command Line Interface ou o AWS Tools for PowerShell. Recomendamos usar chaves de acesso de curto prazo quando possível.

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.

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 AWS Management Console. A forma de conceder acesso programático depende do tipo de usuário que está acessando a AWS.

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

Qual usuário precisa de acesso programático?	Para	Por
Identificação 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 AWS IAM Identity Center 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 Como usar credenciais temporárias com recursos da AWS no Guia do usuário do IAM.
IAM	(Não recomendado) Use credenciais de longo prazo 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 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

Qual usuário precisa de acesso programático?	Para	Por
		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.
  - O AWS CloudShell é um shell pré-autenticado que você pode iniciar diretamente do AWS Management Console. 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.
  - Integração com a AWS CLI versão 2 com o AWS IAM Identity Center (Centro de Identidade do IAM). É possível autenticar usuários e fornecer credenciais de curto prazo para executar comandos da AWS CLI. Para saber mais, consulte Integrar a AWS CLI com o Centro de Identidade IAM no Guia do usuário do AWS IAM Identity Center e Configurar a AWS CLI para usar o Centro de Identidade do IAM no Guia do usuário da AWS Command Line Interface.
- 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 Contas da AWS selecionadas. Para obter mais informações, consulte O que é o AWS IAM Identity Center, Conectar-se a um provedor de identidades externo e Conjuntos de permissões no Guia do usuário do AWS IAM Identity Center.
- 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.

Acessar a AWS usando suas credenciais da AWS

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 AWS Management Console e usa chaves de acesso para fazer chamadas programáticas para a

AWS. Além disso, toda identidade que você usar, podendo ser o usuário raiz da conta, um usuário do AWS Identity and Access Management (IAM), um usuário do AWS IAM Identity Center ou uma identidade federada, terá credenciais exclusivas na AWS.

Para obter instruções detalhadas sobre como fazer login na AWS de acordo com seu tipo de usuário, consulte Como fazer login na AWS no Guia do usuário do AWS Sign-In.

Diretrizes de auditoria de segurança da AWS

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. O Security Hub usa controles de segurança para avaliar configurações de recursos e padrões de segurança que ajudam você a cumprir vários frameworks de conformidade. Para obter mais informações sobre como usar o Security Hub para avaliar os recursos do IAM, consulte Controles do AWS Identity and Access Management no Guia do usuário do AWS Security Hub.

Conteúdo

- 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 AWS OpsWorks, modelos do AWS 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 usuários federados, como usuários no IAM Identity Center. 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 seu 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 a privacidade de dados, consulte as Perguntas frequentes sobre privacidade de dados. Para ter mais informações sobre a proteção de dados na Europa, consulte a postagem do blog AWS Shared Responsibility Model and GDPR no Blog de segurança da AWS.

Para fins de proteção de dados, recomendamos que você proteja as credenciais da Conta da AWS e configure as contas de usuário individuais com o AWS IAM Identity Center ou o 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 o registro em log das atividades da API e do usuário com o AWS CloudTrail.
- Use as soluções de criptografia da AWS, juntamente com todos os controles de segurança padrão dos Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sigilosos armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-2 ao acessar a AWS por meio de uma interface de linha de comando ou uma API, use um endpoint do FIPS. Para ter mais informações sobre endpoints do FIPS, consulte Federal Information Processing Standard (FIPS) 140-2.

É altamente recomendável que nunca sejam colocadas informações de identificação confidenciais, como endereços de email dos seus clientes, em marcações ou campos de formato livre, como um campo Name. 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 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, recomendemos fortemente que não sejam incluídas informações de credenciais no URL para validar a solicitação a esse 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 inativa

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: o AWS STS 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 Regiões e endpoints.

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 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 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 determinados limites forem atingidos. Você também pode 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.

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.

Para saber se um AWS service (Serviço da AWS) está no escopo de programas de conformidade específicos, consulte Serviços da AWS em Escopo por Programa de Conformidade e selecione o programa de conformidade em que você está interessado. Para obter informações gerais, consulteAWS Programas de conformidade.

É possível fazer download de relatórios de auditoria de terceiros usando o AWS Artifact. Para obter mais informações, consulte Downloading Reports inAWS Artifact.

Sua responsabilidade de conformidade ao usar o Serviços da AWS é determinada pela confidencialidade dos seus dados, pelos objetivos de conformidade da empresa e pelos regulamentos e leis aplicáveis. A AWS fornece os seguintes recursos para ajudar com a conformidade:

- Guias de referência rápida de conformidade e segurança - estes guias de implantação discutem considerações sobre arquitetura e fornecem as etapas para a implantação de ambientes de linha de base focados em segurança e conformidade na AWS.
- Arquitetura para segurança e conformidade com HIPAA no Amazon Web Services: esse whitepaper descreve como as empresas podem usar a AWS para criar aplicações adequadas aos padrões HIPAA.

  > **Note**
  >
  > Nem todos os Serviços da AWS estão qualificados pela HIPAA. Para obter mais informações, consulte a Referência dos serviços qualificados pela HIPAA.

- Atributos de conformidade da AWS: essa coleção de manuais e guias pode ser aplicada a seu setor e local.
- Guias de conformidade do cliente da AWS: entenda o modelo de responsabilidade compartilhada sob a ótica da conformidade. Os guias resumem as práticas recomendadas para proteção de Serviços da AWS e mapeiam as diretrizes para controles de segurança em várias estruturas

(incluindo o Instituto Nacional de Padrões e Tecnologia (NIST), o Conselho de Padrões de Segurança do Setor de Cartões de Pagamento (PCI) e a Organização Internacional de Padronização (ISO)).

- Avaliar recursos com regras no AWS ConfigGuia do desenvolvedor: o serviço AWS Config avalia como as configurações de recursos estão em conformidade com práticas internas, diretrizes do setor e regulamentos.
- AWS Security Hub: este AWS service (Serviço da AWS) fornece uma visão abrangente do estado de segurança na AWS. O Security Hub usa controles de segurança para avaliar os atributos da AWS e verificar a conformidade com os padrões e as práticas recomendadas do setor de segurança. Para obter uma lista dos serviços e controles aceitos, consulte a Referência de controles do Security Hub.
- Amazon GuardDuty: esse AWS service (Serviço da AWS) detecta possíveis ameaças às workloads, aos contêineres e aos dados das Contas da AWS monitorando o ambiente em busca de atividades suspeitas e mal-intencionadas. O GuardDuty pode ajudar você a atender a vários requisitos de conformidade, como o PCI DSS, atendendo aos requisitos de detecção de intrusões exigidos por determinados frameworks de conformidade.
- AWS Audit Manager – Esse AWS service (Serviço da AWS) ajuda a auditar continuamente seu uso da AWS para simplificar a forma como você gerencia os riscos e a conformidade com regulamentos e padrões do setor.

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.

Por padrão, as solicitações do AWS STS vão para um único endpoint global. 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.

AWS Management Console

- Talvez você possa usar um endpoint de login regional para entrar no AWS Management Console 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 obter informações 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 suporte para:

- 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.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou você pode usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

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

A AWS 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 seguintes recursos da :

- Modelo de responsabilidade compartilhada
- Amazon Web Services: visão geral do processo de segurança (whitepaper)

Os seguintes recursos 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 gerenciadas pela AWS para o AWS Identity and Access Management Access Analyzer

Uma política gerenciada pela AWS é uma política independente criada e administrada pela AWS. As políticas gerenciadas pela AWS são criadas para fornecer permissões a vários casos de uso comuns a fim de que você possa começar a atribuir permissões a usuários, grupos e perfis.

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 todos os clientes da AWS usarem. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo cliente da específicas para seus casos de uso.

Você não pode alterar as permissões definidas em políticas gerenciadas pela AWS. Se a AWS atualiza as permissões definidas em uma política gerenciada pela AWS, a atualização afeta todas as identidades de entidades principais (usuários, grupos e perfis) às quais a política está vinculada. É mais provável que a AWS atualize uma política gerenciada pela AWS quando um novo AWS service (Serviço da AWS) é lançado ou novas operações de API são disponibilizadas para os serviços existentes.

Para mais informações, consulte Políticas gerenciadas pela AWS no Manual 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. Quando você permite essa ação, o IAM anexa a seguinte política a cada usuário:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword"
            ],
            "Resource": [
                "arn:aws:iam::*:user/${aws:username}"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

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.

```
{
  "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.

```
{
  "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 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 e recuperar políticas de repositórios.
- 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 de usuários, perfis de login, chaves de acesso e os últimos dados acessados 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 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 e buckets do Amazon S3 e buckets de diretório do Amazon S3 que usam a classe de armazenamento Amazon S3 Express One.
- 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AccessAnalyzerServiceRolePolicy",
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetResourcePolicy",
        "dynamodb:ListStreams",
        "dynamodb:ListTables",
        "ec2:DescribeAddresses",
        "ec2:DescribeByoipCidrs",
        "ec2:DescribeSnapshotAttribute",
        "ec2:DescribeSnapshots",
        "ec2:DescribeVpcEndpoints",
        "ec2:DescribeVpcs",
        "ec2:GetSnapshotBlockPublicAccessState",
        "ecr:DescribeRepositories",
        "ecr:GetRepositoryPolicy",
        "elasticfilesystem:DescribeFileSystemPolicy",
        "elasticfilesystem:DescribeFileSystems",
```

```
        "iam:GenerateServiceLastAccessedDetails",
        "iam:GetAccessKeyLastUsed"
        "iam:GetGroup",
        "iam:GetLoginProfile",
        "iam:GetRole",
        "iam:GetServiceLastAccessedDetails",
        "iam:GetUser",
        "iam:ListAccessKeys",
        "iam:ListEntitiesForPolicy",
        "iam:ListRoles",
        "iam:ListUsers",
        "kms:DescribeKey",
        "kms:GetKeyPolicy",
        "kms:ListGrants",
        "kms:ListKeyPolicies",
        "kms:ListKeys",
        "lambda:GetFunctionUrlConfig",
        "lambda:GetLayerVersionPolicy",
        "lambda:GetPolicy",
        "lambda:ListAliases",
        "lambda:ListFunctions",
        "lambda:ListLayers",
        "lambda:ListLayerVersions",
        "lambda:ListVersionsByFunction",
        "organizations:DescribeAccount",
        "organizations:DescribeOrganization",
        "organizations:DescribeOrganizationalUnit",
        "organizations:ListAccounts",
        "organizations:ListAccountsForParent",
        "organizations:ListAWSServiceAccessForOrganization",
        "organizations:ListChildren",
        "organizations:ListDelegatedAdministrators",
        "organizations:ListOrganizationalUnitsForParent",
        "organizations:ListParents",
        "organizations:ListRoots",
        "rds:DescribeDBClusterSnapshotAttributes",
        "rds:DescribeDBClusterSnapshots",
        "rds:DescribeDBSnapshotAttributes",
        "rds:DescribeDBSnapshots",
        "s3:DescribeMultiRegionAccessPointOperation",
        "s3:GetAccessPoint",
        "s3:GetAccessPointPolicy",
        "s3:GetAccessPointPolicyStatus",
        "s3:GetAccountPublicAccessBlock",
```

```
        "s3:GetBucketAcl",
        "s3:GetBucketLocation",
        "s3:GetBucketPolicyStatus",
        "s3:GetBucketPolicy",
        "s3:GetBucketPublicAccessBlock",
        "s3:GetMultiRegionAccessPoint",
        "s3:GetMultiRegionAccessPointPolicy",
        "s3:GetMultiRegionAccessPointPolicyStatus",
        "s3:ListAccessPoints",
        "s3:ListAllMyBuckets",
        "s3:ListMultiRegionAccessPoints",
        "s3express:GetBucketPolicy",
        "s3express:ListAllMyDirectoryBuckets",
        "sns:GetTopicAttributes",
        "sns:ListTopics",
        "secretsmanager:DescribeSecret",
        "secretsmanager:GetResourcePolicy",
        "secretsmanager:ListSecrets",
        "sqs:GetQueueAttributes",
        "sqs:ListQueues"
      ],
      "Resource": "*"
    }
  ]
}
```

Atualizações do IAM e do IAM Access Analyzer sobre 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
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
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	26 de novembro de 2023

Alteração	Descrição	Data
	para executar verificações de política em suas políticas.	
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.	1º de março de 2021

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 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 principal 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 uma função do IAM, um usuário federado, um produto da AWS, 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.

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, 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 desejar confirmar que uma alteração feita em uma política resolve um problema de acesso relatado em uma descoberta, você poderá fazer outra verificação do recurso relatado em uma descoberta usando o link Rescan (Verificar novamente) na página de detalhes Findings (Descoberta) ou usando

a operação `StartResourceScan` da API do IAM Access Analyzer. Para saber mais, consulte Resolver descobertas.

> **Important**
>
> 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 para habilitar o IAM Access Analyzer em cada região em que você está usando recursos da AWS compatíveis.

O IAM Access Analyzer analisa os seguintes tipos de recursos:

- Buckets do Amazon Simple Storage Service
- Buckets de diretório do Amazon Simple Storage Service
- Funções do AWS Identity and Access Management
- Chaves do AWS Key Management Service
- Funções e camadas do AWS Lambda
- Filas do Amazon Simple Queue Service
- segredos do AWS Secrets Manager
- 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

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.

As descobertas tanto dos analisadores de acessos externos quanto de acessos não utilizados são organizadas em um painel de resumo visual. O painel destaca suas 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.

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 ter mais informações, consulte Refinar permissões na AWS usando as informações do último acesso.

Validação de políticas em relação às práticas recomendadas da AWS

Você pode 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. Você pode 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 Validação de política do IAM Access Analyzer.

Validar políticas de acordo com seus padrões especificados de segurança

Você pode 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. Você pode 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 Verificações de política personalizadas 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. Você pode usar a política gerada para refinar as permissões de uma entidade anexando-a a um usuário ou uma função 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 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 cobranças e preços 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 AWS Billing and Cost Management. 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 AWS Support.

Descobertas para acessos externos e não utilizados

O IAM Access Analyzer gera descobertas para acessos externos e acessos não utilizados em 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.

O IAM Access Analyzer também gera descobertas sobre as acessos não utilizados concedidos em sua organização e 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 utilizadas: credenciais pertencentes aos usuários do IAM que permitem que eles acessem sua Conta da AWS.
- 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 acessos externos gratuitamente e cobra por descobertas de acessos não utilizados com base no número de funções e usuários do IAM analisados por analisador por mês. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

Tópicos

Funcionamento das descobertas do IAM Access Analyzer

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.

Acessos externos

Para analisadores de acessos externos, o AWS Identity and Access Management Access Analyzer foi desenvolvido com base no Zelkova, que 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) com relação ao problema. O IAM Access Analyzer aplica o Zelkova repetidamente a uma política com consultas cada vez mais específicas para caracterizar classes de comportamentos que a política permite, com base no conteúdo da política. Para saber mais 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 acessou um recurso dentro da zona de confiança. Ele gera uma descoberta quando uma política baseada em recursos permite o acesso a um recurso, mesmo que o recurso não tenha sido acessado pela entidade externa. O IAM Access Analyzer também não considera o estado de quaisquer contas externas ao fazer sua determinação. Ou seja, se ele indicar que a conta 111122223333 pode acessar seu bucket do Amazon S3, ele não sabe nada sobre o estado dos usuários, sobre as funções, sobre as Políticas de controle de serviço (SCP) nem sobre outras configurações relevantes dessa conta. Isto é para a privacidade do cliente, o IAM Access Analyzer não considera quem é o proprietário da outra conta. Isso também é para segurança, se a conta não for de propriedade do cliente do IAM Access Analyzer, ainda será importante saber que uma entidade externa poderá obter acesso aos seus recursos, mesmo que, no momento, não haja entidades principais na conta que possam acessar os recursos.

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, causariam impacto na autorização. Para obter exemplos de chaves de condição que o IAM Access Analyzer considera, consulte Chaves de filtro do IAM Access Analyzer.

No momento, o IAM Access Analyzer não relata descobertas de entidades principais de serviço da AWS ou de contas de serviço internas. Em casos raros em que o IAM Access Analyzer não é capaz de determinar completamente se uma instrução de política concede acesso a uma entidade externa, ele erra ao declarar uma descoberta falsa positiva. O IAM Access Analyzer foi projetado

para fornecer uma visão abrangente do compartilhamento de recursos na conta e faz o possível para minimizar a ocorrência de falsos negativos.

Acessos não utilizados

Você deve criar um analisador para descobertas de acessos não utilizados para suas funções, mesmo que já tenha criado um analisador para gerar descobertas de acessos externos para seus recursos. Após criar o analisador, o IAM Access Analyzer analisa a atividade de acessos para identificar acessos não utilizados. O IAM Access Analyzer analisa as informações do último acesso para todas as funções, chaves de acesso do usuário e senhas de usuário em sua organização e contas AWS para ajudar você a identificar os acessos não utilizados. Para usuários ou perfis do IAM ativos, o IAM Access Analyzer usa as informações do último acesso do serviço e da ação do IAM para identificar permissões não utilizadas. É possível usar analisadores de acessos não utilizados para escalar seu processo de revisão no nível da organização e da conta AWS. É possível usar as informações do último acesso da ação para realizar uma investigação mais profunda das funções individuais.

Painel de resumo

Tanto para os acessos externos quanto para os acessos não utilizados, o IAM Access Analyzer organiza as descobertas em um painel de resumo. Para acessos externos, o painel de resumo destaca a divisão entre as descobertas de acesso entre contas e público e fornece um detalhamento das descobertas por tipo de recurso. Para acessos não utilizados, o painel destaca suas Contas da AWS que têm mais descobertas e fornece um detalhamento das descobertas por tipo. Após criar um analisador para acessos externos ou não utilizados, o IAM Access Analyzer adiciona automaticamente novas descobertas ao painel com foco em funções com permissões não utilizadas.

Conceitos básicos sobre descobertas 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 IAM Access Analyzer, além de como habilitar o IAM Access Analyzer. Para saber mais sobre a função vinculada ao serviço para o IAM Access Analyzer, consulte Usar funções vinculadas ao serviço do AWS Identity and Access Management 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.

Políticas gerenciadas pela AWS 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.

Para ter mais informações, consulte Usar funções vinculadas ao serviço do AWS Identity and Access Management Access Analyzer.

> **Note**
>
> O IAM Access Analyzer é regional. Para utilizar acessos externos, é necessário habilitar o IAM Access Analyzer em cada região de maneira independente.

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 criar um analisador de acessos externos ou não utilizados no IAM Access Analyzer, a página Descobertas ou o painel são carregados sem descobertas 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 acessos externos, pode levar até 30 minutos depois que uma política é modificada para que o IAM Access Analyzer analise o recurso e gere outra descoberta ou atualize uma descoberta existente para o acesso ao recurso. Para analisadores de acessos externos e não utilizados, 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.

Habilitar o IAM Access Analyzer

Como criar um analisador de acessos externos com a Conta da AWS como zona de confiança

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.

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. Selecione Access analyzer (Analisador de acesso).
3. Escolha Configurações do analisador.
4. Selecione Create analyzer (Criar analisador).
5. Na seção Análise, escolha Análise de acessos externos.
6. Na seção Detalhes do analisador confirme se a região exibida é a região em que você deseja habilitar o IAM Access Analyzer.
7. Insira um nome para o analisador.
8. Escolha Conta da AWS atual como a zona de confiança para o 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.

9. Opcional. Adicione as tags que deseja aplicar ao analisador.
10. Selecione Enviar.

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.

Como criar um analisador de acessos externos com a organização como zona de confiança

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. Selecione Access analyzer (Analisador de acesso).
3. Escolha Configurações do analisador.

4. Selecione Create analyzer (Criar analisador).
5. Na seção Análise, escolha Análise de acessos externos.
6. Na seção Detalhes do analisador confirme se a região exibida é a região em que você deseja habilitar o IAM Access Analyzer.
7. Insira um nome para o analisador.
8. Escolha Organização atual como a zona de confiança para o analisador.
9. Opcional. Adicione as tags que deseja aplicar ao analisador.
10. 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.

Para criar 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.

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. Selecione Access analyzer (Analisador de acesso).
3. Escolha Configurações do analisador.
4. Selecione Create analyzer (Criar analisador).
5. Na seção Análise, escolha Análise de acessos não utilizados.
6. Insira um nome para o analisador.
7. Em Período de rastreamento, insira o número de dias para gerar descobertas para permissões não utilizadas. Por exemplo, se você inserir 90 dias, o analisador gerará descobertas para entidades do IAM na conta selecionada para quaisquer permissões que não tenham sido usadas em 90 dias ou mais desde a última verificação do analisador. É possível escolher um valor entre 1 e 180 dias.

8. Para Contas selecionadas, escolha Conta da AWS 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. Adicione as tags que deseja aplicar ao analisador.
10. Selecione Enviar.

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.

Para criar 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 a conta de um 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 IAM em https://console.aws.amazon.com/iam/.
2. Selecione Access analyzer (Analisador de acesso).
3. Escolha Configurações do analisador.
4. Selecione Create analyzer (Criar analisador).

5. Na seção Análise, escolha Análise de acessos não utilizados.
6. Insira um nome para o analisador.
7. Em Período de rastreamento, insira o número de dias para gerar descobertas para permissões não utilizadas. Por exemplo, se você inserir 90 dias, o analisador gerará descobertas para entidades do IAM nas contas da organização selecionada para quaisquer permissões que não tenham sido usadas em 90 dias ou mais desde a última verificação do analisador. É possível escolher um valor entre 1 e 180 dias.
8. Para Contas selecionadas, escolha Organização atual como as contas selecionadas para o analisador.
9. Opcional. Adicione as tags que deseja aplicar ao analisador.
10. Selecione Enviar.

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.

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 acessos externos, o analisador está monitorando ativamente os recursos dentro de sua zona de confiança. O analisador gera ativamente novas descobertas e atualiza as descobertas existentes. Para analisadores de acessos não utilizado s, 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.

Status	Descrição
Criando	A criação do analisador ainda está em andamento. O analisador fica ativo quando a criação é concluída.
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.
Com falha	A criação do analisador falhou devido a um problema de configuração. O analisador não gerará nenhuma descoberta. Exclua o analisador e crie outro analisador.

Visualizar o painel de descobertas do IAM Access Analyzer

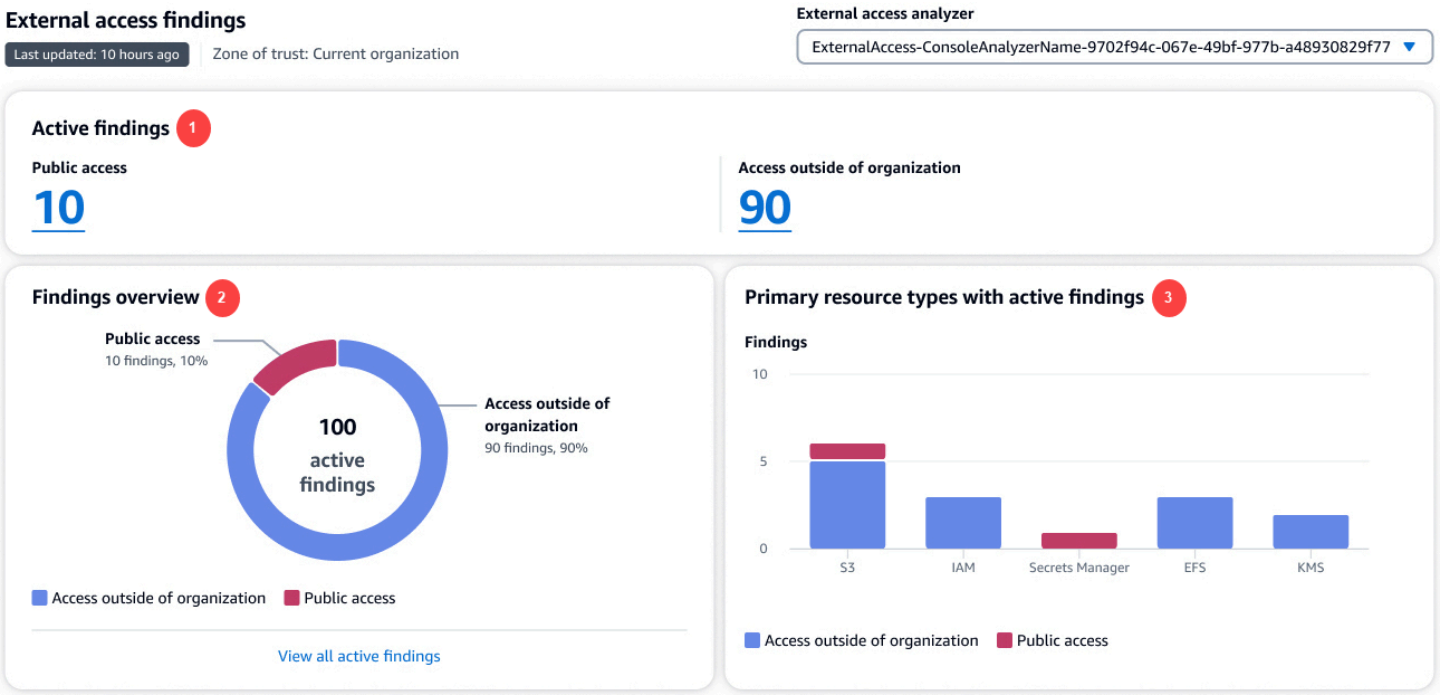
O AWS Identity and Access Management Access Analyzer organiza as acessos externos e as descobertas de acessos não utilizados em um painel de resumo visual. O painel ajuda você a obter visibilidade sobre o uso efetivo de permissões em escala e a identificar contas que precisam de atenção. É possível usar o painel para analisar as descobertas por organização AWS, conta e tipo de descoberta.

Para visualizar o painel de resumo dos analisadores de acessos externos

> **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 IAM em https://console.aws.amazon.com/iam/.

2. Selecione Access analyzer (Analisador de acesso). A janela Resumo será exibida.
3. Escolha um analisador no menu suspenso Analisador de acessos externos. Um resumo das descobertas do analisador é exibido na seção Descobertas de acessos externos.



Na imagem anterior, o painel de descobertas de acessos externos está visível na página Resumo:

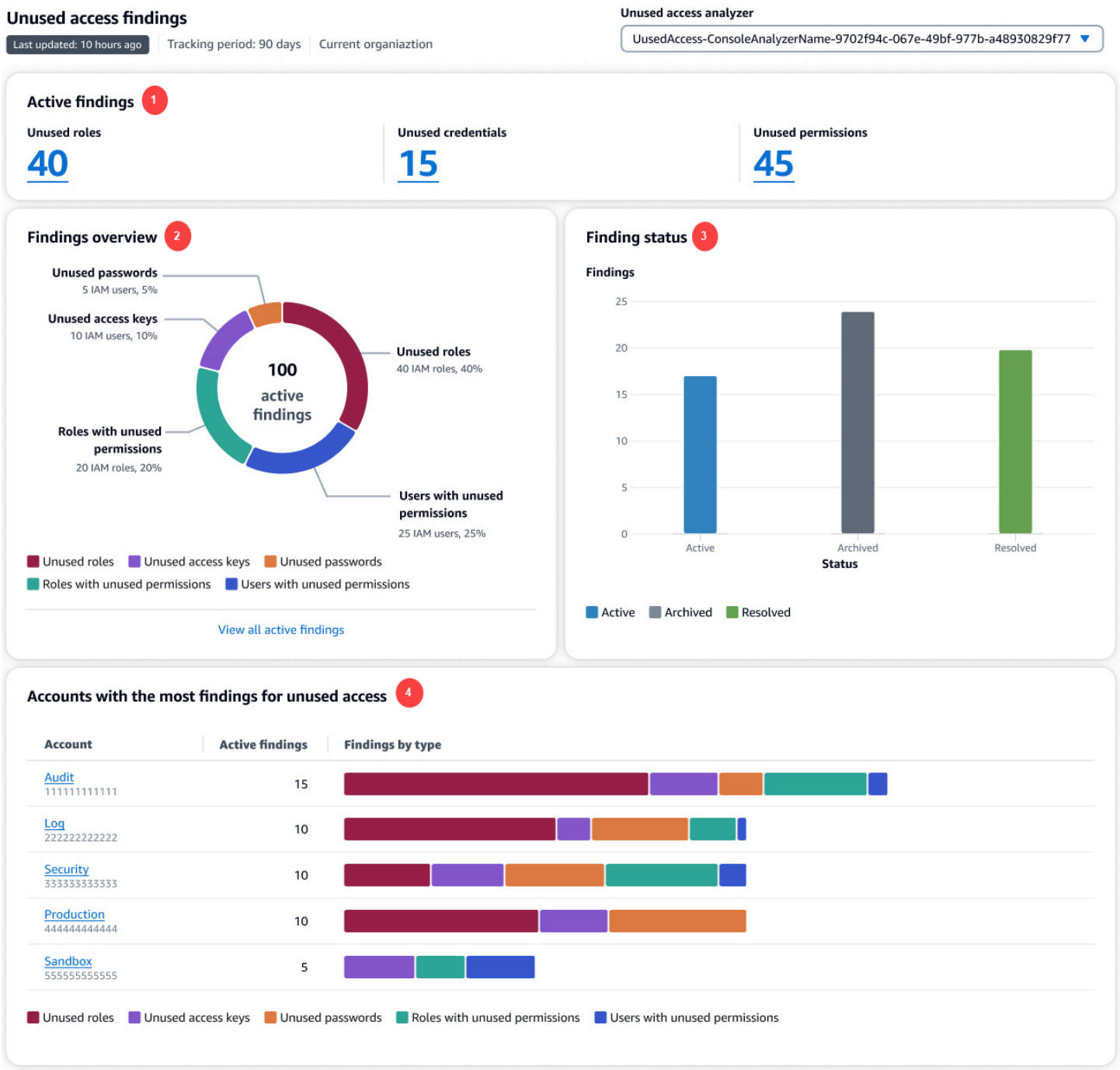
1. A seção Descobertas ativas inclui o número de descobertas ativas para acesso público e o número de descobertas ativas que fornecem acesso fora da conta ou da organização. Escolha um número para listar todas as descobertas ativas de cada tipo.
2. A seção Visão geral das descobertas inclui um detalhamento 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 Tipos de recursos primários com descobertas ativas inclui um detalhamento dos tipos de recursos primários com descobertas ativas. Essas informações ajudam você a priorizar as descobertas dos recursos primários. Por exemplo, Amazon S3, DynamoDB e AWS KMS. Essa não é uma lista completa de todos os tipos de recursos. Seu analisador pode ter descobertas ativas para tipos de recurso não listados nesta seção.

Para visualizar o painel de resumo dos analisadores 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. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

> **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 IAM em https://console.aws.amazon.com/iam/.
2. Selecione Access analyzer (Analisador de acesso). A janela Resumo será exibida.
3. Escolha um analisador no menu suspenso Analisador de acessos não utilizados. Um resumo das descobertas do analisador é exibido na seção Descobertas de acessos não utilizados.



Na imagem anterior, o painel de descobertas de acessos externos está visível 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 (Ativas, Arquivadas e Resolvidas) de sua conta ou organização.
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.

Como trabalhar com descobertas

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. 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.

Descobertas de acessos não utilizados

As descobertas de acessos não utilizados são geradas para entidades 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.

Você deve revisar todas as descobertas em sua conta para determinar se os acessos externos ou não utilizados estão previstos e aprovados. Se os acessos externos ou não utilizados identificados na descoberta forem esperados, será possível 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.

> **Note**
>
> As descobertas de acesso não utilizadas só estão disponíveis usando a ação da API ListFindingsV2.

Analisar descobertas

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.

Como revisar descobertas

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. Selecione Access analyzer (Analisador de acesso).

3. O painel de descobertas é exibido. Selecione as descobertas ativas para seu analisador de acessos externos ou não utilizados.

   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.

Todas as descobertas são exibidas para o analisador. Para visualizar outras descobertas geradas pelo analisador, selecione o tipo de descoberta apropriado no menu suspenso Status:

- Selecione Active (Ativa) para visualizar todas as descobertas ativas geradas pelo analisador.
- Selecione Archived (Arquivada) para visualizar somente as descobertas geradas pelo analisador que foram arquivadas. Para saber mais, consulte Arquivar descobertas.
- Selecione Resolved (Resolvida) para visualizar somente descobertas geradas pelo analisador que foram resolvidas. Ao corrigir o problema que gerou a descoberta, o status da descoberta é alterado para Resolvida.

  > **Important**
  >
  > As descobertas resolvidas são excluídas 90 dias após a última atualização da descoberta. As descobertas ativas e arquivadas não são excluídas, a menos que você exclua o analisador que as gerou.

- Selecione All (Tudo) para visualizar todas as descobertas com qualquer status gerado pelo analisador.

Descobertas de acessos externos

Escolha Acesso externo e, em seguida, escolha o analisador de acesso externo no menu suspenso Exibir analisador. A página Descobertas para analisadores de acesso externo exibe os seguintes detalhes sobre o recurso compartilhado e a instrução de política que gerou a descoberta:

ID da descoberta

O ID exclusivo atribuído à descoberta. Selecione o ID da descoberta para exibir detalhes adicionais sobre o recurso e sobre a instrução de política que gerou a descoberta.

Recurso

O tipo e o nome parcial do recurso com uma política aplicada a ele que concede acesso a uma entidade externa que não está na sua zona de confiança.

Resource owner account (Conta de proprietário do recurso)

Essa coluna será exibida somente se você estiver usando uma organização como zona de confiança. A conta na organização que possui o recurso relatado na descoberta.

External principal (Principal externo)

O principal, que não está dentro da sua zona de confiança, ao qual a política analisada concede acesso. Os valores válidos são:

- Conta da AWS: todas as entidades principais na Conta da AWS listada com permissões do administrador dessa conta podem acessar o recurso.
- Qualquer entidade principal: todas as entidades principais em qualquer Conta da AWS que atenda às condições incluídas na coluna Condições têm permissão para acessar o recurso. Por exemplo, se uma VPC estiver listada, isso significa que qualquer entidade principal em qualquer conta com permissão para acessar a VPC listada pode acessar o recurso.
- Usuário canônico: todas as entidades principais na Conta da AWS com o ID de usuário canônico listado têm permissão para acessar o recurso.
- IAM role (Função do IAM): a função do IAM listada tem permissão para acessar o recurso.
- IAM user (Usuário do IAM) o usuário do IAM listado tem permissão para acessar o recurso.

Condição

A condição da instrução de política que concede o acesso. Por exemplo, se o campo Condition (Condição) incluir Source VPC (VPC de origem), isso significa que o recurso é compartilhado com um principal que tem acesso à VPC listada. As condições podem ser globais ou específicas do serviço. As chaves de condição globais têm o prefixo `aws:`.

Shared through (Compartilhado por)

O campo Shared through (Compartilhado por) indica como o acesso que gerou a descoberta é concedido. Os valores válidos são:

- Bucket policy (Política de bucket): a política do bucket anexada ao bucket do Amazon S3.
- Access control list (Lista de controle de acesso): a lista de controle de acesso (ACL) que está anexada ao bucket do Amazon S3.
- Access point (Ponto de acesso): um ponto de acesso ou ponto de acesso multirregiões associado ao bucket do Amazon S3. O ARN do ponto de acesso é exibido nos detalhes de Findings (Descobertas).

Nível de acesso

O nível de acesso concedido à entidade externa pelas ações da política baseada em recursos. Visualize os detalhes da descoberta para obter mais informações. Os valores do nível de acesso incluem o seguinte:

- List (Listar): permissão para listar recursos dentro do serviço a fim de determinar se um objeto existe. Ações com esse nível de acesso podem listar objetos, mas não podem ver os conteúdos de um recurso.
- Read (Ler): permissão para ler, mas não para editar os conteúdos e os atributos de recursos no serviço.
- Write (Gravar): permissão para criar, excluir ou modificar recursos no serviço.
- Permissions (Permissões): permissão para conceder ou modificar permissões de recursos no serviço.
- Tagging (Marcar): permissão para executar ações que apenas alteram o estado de tags do recurso.

Atualizado

Um carimbo de data/hora para a atualização mais recente no status da descoberta, ou a hora e a data em que a descoberta foi gerada se nenhuma atualização tiver sido feita.

> **Note**
>
> Pode levar até 30 minutos depois que uma política é modificada para que o IAM Access Analyzer analise novamente o recurso e atualize a descoberta.

Status

O status da descoberta, que pode ser Active (Ativa), Archived (Arquivada) ou Resolved (Resolvida).

Descobertas de acessos não utilizados

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.

Escolha Acessos não utilizados e, em seguida, escolha o analisador com acessos nao utilizados no menu suspenso Exibir analisador. A página Descobertas para analisadores de acesso não utilizados exibe os seguintes detalhes sobre a entidade do IAM que gerou a descoberta:

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.

ID do Conta da AWS

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

A filtragem padrão da página de descobertas é exibir todas as descobertas. Para visualizar descobertas ativas, escolha o status Ativa no menu suspenso Status. Para ver as descobertas

arquivadas, escolha o status Arquivadas no menu suspenso Status. 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. Por exemplo, para criar um filtro que exibe somente descobertas de uma Conta da AWS específica, escolha Conta da AWS para a propriedade, e em seguida, Conta da AWS = e insira o número da Conta da AWS da qual deseja visualizar as descobertas.

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 descobertas de acessos externos

Para filtrar descobertas de acessos externos

1. Escolha Acesso externo e, em seguida, escolha o analisador no menu suspenso Exibir analisador.
2. Escolha a caixa de pesquisa para exibir uma lista das propriedades disponíveis.
3. Escolha a propriedade que deve ser usada para filtrar as descobertas exibidas.
4. 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, e em seguida escolha Recurso :, depois digite uma parte do nome ou o nome completo 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 a definição de filtros:

- 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.
- 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.
- 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.
- 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.
- 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.
- 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.
- 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 acessos não utilizados

Para filtrar descobertas de acessos não utilizados

1. Escolha Acessos não utilizados e, em seguida, escolha o analisador no menu suspenso Exibir analisador.
2. Escolha a caixa de pesquisa para exibir uma lista das propriedades disponíveis.
3. Escolha a propriedade que deve ser usada para filtrar as descobertas exibidas.
4. 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, e em seguida escolha Tipo de descobertas =, e escolha Tipo de descobertas = UnusedIAMRole. Somente as descobertas com um tipo UnusedIAMRole 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

Ao obter uma descoberta de acesso a um recurso que é intencional, é possível arquivar as descobertas. Por exemplo, uma descoberta de acessos externos para um perfil do IAM utilizado por vários usuários para fluxos de trabalho aprovados ou uma descoberta de acessos não utilizados 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. Selecione Archive (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

Descobertas de acessos externos

Para resolver descobertas de acessos externos gerados por credenciais que você não pretendia permitir, modifique a instrução da política para remover as permissões que permitem acesso ao recurso identificado. Por exemplo, para descobertas em 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. Use o console para os outros recursos compatíveis a fim de modificar as instruções de política que resultaram em uma descoberta gerada.

Após fazer a alteração para resolver uma descoberta de acessos externos, como modificar uma política aplicada a um perfil do IAM, o IAM Access Analyzer verificará o recurso novamente. Se o recurso não for mais compartilhado fora da sua zona de confiança, o status da descoberta será alterado para Resolved (Resolvida). A descoberta não será mais exibida na lista de descobertas ativas e, em vez disso, será exibida na lista de descobertas resolvidas.

> **Note**
>
> Isso não se aplica às descobertas de Erro. Quando o IAM Access Analyzer não consegue analisar um recurso, ele gera 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.

Se as alterações feitas resultarem no compartilhamento do recurso fora da sua zona de confiança, mas de maneira diferente, como com outra entidade principal ou para outra permissão, o IAM Access Analyzer gerará outra descoberta com status Active (Ativo).

> **Note**
>
> Pode levar até 30 minutos depois que uma política é modificada para que o IAM Access Analyzer analise novamente o recurso e atualize a descoberta. As descobertas resolvidas são excluídas 90 dias após a última atualização do status da descoberta.

Descobertas de acessos não utilizados

Para resolver descobertas de acessos não utilizados, use o console do IAM para remover a chave de acesso, a senha, a permissão ou a função não utilizada. Para obter mais informações, consulte os seguintes recursos do :

- Para obter mais informações sobre como excluir uma chave de acesso, consulte Gerenciar chaves de acesso (console).
- Para obter mais informações sobre como excluir uma senha de usuário do IAM, consulte Criar, alterar ou excluir uma senha de usuário do IAM (console).
- Para obter mais informações sobre como editar as permissões de um usuário do IAM, consulte Alteração das permissões de um usuário (console).
- Para obter mais informações sobre como excluir um perfil do IAM, consulte Exclusão de um perfil do IAM (console).

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, você remove 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 usuários e perfis do IAM analisados por mês. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

Tipos de recursos do IAM Access Analyzer para acessos externos

Para analisadores de acessos externos, o IAM Access Analyzer analisa as políticas baseadas em recurso que são aplicadas aos recursos da AWS na região na qual você habilitou o IAM Access Analyzer. Somente analisa as políticas baseadas em recursos. Revise as informações sobre cada recurso para ver os detalhes sobre como o IAM Access Analyzer gera descobertas para cada tipo de recurso.

> **Note**
>
> Os tipos de recursos suportados listados são para analisadores de acessos externos. Os analisadores de acessos não utilizados oferecem suporte apenas a perfis e usuários do IAM. Para ter mais informações, consulte Como trabalhar com descobertas.

Tipos de recursos compatíveis para acessos externos:

- Buckets do Amazon Simple Storage Service
- Buckets de diretório do Amazon Simple Storage Service
- Funções do AWS Identity and Access Management
- Chaves do AWS Key Management Service
- Funções e camadas do AWS Lambda
- Filas do Amazon Simple Queue Service
- segredos do AWS Secrets Manager
- 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

Buckets do Amazon Simple Storage Service

Quando o IAM Access Analyzer analisa buckets do Amazon S3, ele gera uma descoberta quando uma política de bucket do Amazon S3, uma ACL ou um ponto de acesso, incluindo um ponto de acesso de várias regiões, 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.

> **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.

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 usam a classe de armazenamento Amazon S3 Express One, que é recomendada para cargas de trabalho ou aplicativos de desempenho crítico. Para diretório bucket do Amazon S3, o IAM Access Analyzer analisa as políticas de diretório bucket, incluindo declarações de condição em uma política, que permitem que uma entidade externa acesse um diretório bucket. Para mais informações sobre os buckets do diretório do Amazon S3, consulte Directory buckets no Guia do usuário do Amazon Simple Storage Service.

Funções do AWS Identity and Access Management

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.

Chaves do AWS Key Management Service

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.

Funções e camadas do AWS Lambda

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.

segredos do AWS Secrets Manager

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. Você pode 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 snapshots de banco de dados do Amazon EBS, o IAM Access Analyzer analisa listas de controle de acesso que permitem que uma entidade externa acesse um snapshot. 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.

> **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 snapshots de cluster de banco de dados do Amazon RDS, o IAM Access Analyzer analisa listas de controle de acesso que permitem que uma entidade externa acesse um snapshot. 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.

> **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

O IAM Access Analyzer gerará 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.

Tabelas do Amazon DynamoDB

O IAM Access Analyzer gerará 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.

Configurações 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.

Administrador delegado do IAM Access Analyzer

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.

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 de uma conta-membro da organização Conta da AWS para torná-la o administrador delegado.

   A conta deve ser membro da organização.
6. Escolha Save changes (Salvar alterações).

Para adicionar um administrador delegado usando a AWS CLI ou AWS SDKs

Ao criar um analisador para analisar acessos externos na organização em uma conta de administrador delegado usando a AWS CLI, a API da AWS (usando os SDKs da AWS) ou o AWS CloudFormation, você deverá usar APIs do AWS Organizations para habilitar o acesso ao serviço para o IAM Access Analyzer e registrar a conta-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-membro válida de sua organização da AWS como administrador delegado usando a operação da API `RegisterDelegatedAdministrator` do AWS Organizations ou o comando `register-delegated-administrator` da AWS CLI.

Após alterar o administrador delegado, o novo administrador deverá criar analisadores para começar a monitorar o acesso aos recursos da organização.

Excluindo analisadores

Você pode excluir analisadores de acessos externos e não utilizados existentes na página Configurações do analisador. Quando você exclui um analisador, os recursos especificados no analisador não são mais monitorados e nenhuma nova descoberta é gerada. Todas as descobertas geradas pelo analisador são excluídas.

Para descobertas que são excluídas em função da exclusão do analisador que as gerou, o evento é enviado ao EventBridge nos dois dias subsequentes à exclusão do analisador. Após a exclusão do analisador, pode levar até 90 dias para que as descobertas do Security Hub sejam excluídas.

Para excluir um analisador

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Em Analisador de acessos, selecione Configurações do analisador.
3. Selecione o analisador para excluir e, em seguida, escolha Excluir.
4. Digite **`delete`** na caixa de texto de confirmação e, em seguida, escolha Excluir.

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 regra e adicionar vários critérios, poderá remover um único critério da regra selecionando Remove this criterion (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. Selecione a caixa de seleção para as regras que deseja excluir.
2. Escolha Delete (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.

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:::my-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:::my-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"
}
```

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:::example-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:::example-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:::example-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 de descobertas de acessos externos ou não utilizados, use o padrão do seguinte exemplo:

```
{
  "source": [
    "aws.access-analyzer"
  ],
  "detail-type": [
    "Access Analyzer 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"
  ]
}
```

> **Note**
>
> Não é possível criar uma regra com base somente em um evento de descoberta de acesso externo.

- 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 uma função 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 Access Analyzer com o AWS Security Hub

O AWS Security Hub fornece uma visão abrangente do estado de segurança na AWS e ajuda a verificar o ambiente em relação aos padrões e às práticas recomendadas do setor de segurança. O Security Hub coleta dados de segurança de contas, serviços e produtos compatíveis de terceiros parceiros da AWS e ajuda a analisar suas tendências de segurança e identificar os problemas de segurança de prioridade mais alta.

Ao integrar o AWS Identity and Access Management Access Analyzer com o Security Hub, você pode enviar descobertas do IAM Access Analyzer para o Security Hub. O Security Hub pode então incluir tais descobertas na análise feita sobre a seu procedimento de segurança.

Sumário

Como o IAM Access Analyzer envia descobertas para o Security Hub

No Security Hub, os problemas de segurança são rastreados como descobertas. Algumas descobertas provêm de problemas que são detectados por outros produtos da AWS ou por parceiros terceirizados. O Security Hub também tem um conjunto de regras que ele usa para detectar problemas de segurança e gerar descobertas.

O Security Hub fornece ferramentas para gerenciar descobertas em todas essas fontes. Você pode exibir e filtrar listas de descobertas e exibir detalhes de uma descoberta. Consulte Visualizar descobertas no Guia do usuário do AWS Security Hub. Você também pode rastrear o status de uma investigação em uma descoberta. Consulte Tomar medidas sobre descobertas no Guia do usuário do AWS Security Hub.

Todas as descobertas no Security Hub usam um formato JSON padrão chamado ASFF (Formato de Descoberta de Segurança da AWS). O ASFF inclui detalhes sobre a origem do problema, os recursos afetados e o status atual da descoberta. Consulte ASFF (Formato de Descoberta de Segurança) da AWS no Guia do usuário do AWS Security Hub.

O AWS Identity and Access Management Access Analyzer é um dos serviços da AWS que envia descobertas para o Security Hub. 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. Em seguida, o IAM Access Analyzer envia essas descobertas ao Security Hub. Para acesso externo, o IAM Access Analyzer detecta uma declaração de política que permite acesso público ou acesso entre contas a entidades principais externas em um recurso compatível na sua organização ou conta. O IAM Access Analyzer gera uma descoberta para acesso público, que depois envia ao Security Hub. Para acesso entre contas, o IAM Access Analyzer envia uma única descoberta de uma entidade principal externa por vez ao Security Hub. Se houver várias descobertas entre contas no IAM Access Analyzer, você deve resolver a descoberta do Security Hub 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.

Tipos de descobertas que o IAM Access Analyzer envia

O IAM Access Analyzer envia as descobertas para o Security Hub 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 em até 30 minutos. Em raras ocasiões e sob determinadas condições, o IAM Access Analyzer não é notificado sobre a inclusão ou atualização de uma política. Por exemplo, uma alteração nas configurações de bloqueio de acesso público no nível da conta no Amazon S3 pode levar até 12 horas. Além disso, se houver um problema de entrega com a entrega do log do AWS CloudTrail, a alteração da política não acionará uma nova verificação do recurso que foi relatado na descoberta. Quando isso acontece, o IAM Access Analyzer analisa a política nova ou atualizada durante a próxima verificação periódica.

Tentar novamente quando o Security Hub não estiver disponível

Se o Security Hub não estiver disponível, o IAM Access Analyzer tentará enviar as descobertas periodicamente.

Atualizar as descobertas do existentes no Security Hub

Após enviar uma descoberta ao Security Hub, o AWS Identity and Access Management Access Analyzer envia atualizações para refletir observações adicionais da atividade da descoberta para o Security Hub. As atualizações são refletidas dentro da mesma descoberta.

O IAM Access Analyzer agrupa as descobertas de acessos externos por recurso, a descoberta para um recurso no Security Hub estará ativa se pelo menos uma das descobertas para o recurso no IAM Access Analyzer estiver ativa. Se todas as descobertas no IAM Access Analyzer para um recurso forem arquivadas ou resolvidas, a descoberta do Security Hub será arquivada. A descoberta do

Security Hub é atualizada quando você altera o acesso de políticas entre público e entre contas. Essa atualização pode incluir alterações no tipo, título, descrição e gravidade da descoberta.

O IAM Access Analyzer não agrupa descobertas de acessos não utilizados por recurso, portanto, se uma descoberta de acessos não utilizados for resolvida no IAM Access Analyzer, a descoberta do Security Hub será resolvida. A descoberta do Security Hub é atualizada quando você atualiza o perfil ou a usuário do IAM que gerou a descoberta de acessos não utilizados.

Exibir descobertas do IAM Access Analyzer no Security Hub

Para visualizar suas descobertas do IAM Access Analyzer no Security Hub, escolha Exibir 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`**.

Interpretar nomes de descobertas do IAM Access Analyzer no Security Hub

O AWS Identity and Access Management Access Analyzer envia descobertas para o Security Hub 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.

Tipo de descoberta do ASFF	Título da descoberta do Security Hub	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	O <resource ARN> permite o acesso entre contas	Uma política baseada em recursos anexada ao recurso permite o acesso entre

Tipo de descoberta do ASFF	Título da descoberta do Security Hub	Descrição
daAWS/Acesso externo concedido		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.
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 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:::my-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:::my-bucket/ allows cross-account access",
    "Description": "AWS::S3::Bucket/arn:aws:s3:::my-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:::my-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, você deve habilitar o Security Hub. Para obter informações sobre como habilitar o Security Hub, consulte Configurar o Security Hub no Guia do usuário AWS Security Hub.

Ao habilitar tanto o IAM Access Hub quanto o Security Hub, a integração é habilitada automaticamente. O IAM Access Analyzer começa imediatamente a enviar descobertas para o Security Hub.

Como parar de enviar descobertas

Para parar de enviar descobertas para o Security Hub, você pode usar o console ou a API 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 em Event history (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 da AWS quando ela é criada. 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 os eventos recentes em sua conta da AWS. Para obter mais informações, consulte Como visualizar 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 você cria uma trilha no console, ela é aplicada a todas as regiões da AWS. A trilha registra em log eventos de todas as regiões na partição da AWS e entrega os arquivos de log para o bucket do Amazon S3 especificado por você. 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 compatíveis com o CloudTrail
- Configurar notificações do Amazon SNS para o 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 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 para as 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 do AWS Identity and Access Management (IAM).
- Se a solicitação foi feita com credenciais de segurança temporárias de uma função ou de um usuário federado.
- Se a solicitação foi feita por outro serviço da AWS.

Para obter mais informações, 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 do 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, a data e a hora da ação, os parâmetros de solicitação e assim por diante. Os arquivos de

log do CloudTrail não são um rastreamento de pilha ordenada de chamadas de API pública. Dessa forma, eles 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 AWS CloudFormation para configurar regras de arquivamento.

Criterion	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso
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)	Sim

Criterion	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso
resourceType `AWS::IAM::Role | AWS::KMS::Key | AWS::Lambda::Function | AWS::Lambda::LayerVersion | AWS::S3::Bucket | AWS::S3Express::DirectoryBucket | AWS::SQS::Queue | AWS::SecretsManager::Secret | AWS::EFS::FileSystem | AWS::EC2:`	O tipo de recurso ao qual o principal externo tem acesso.	String	 Sim	 Yes (Sim)	 Sim

Criterion	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso
`:Snapshot | AWS::ECR::Repository | AWS::RDS::DBSnapshot | AWS::RDS::DBClusterSnapshot | AWS::SNS::Topic | AWS::DynamoDB::Stream | AWS::DynamoDB::Table`					
resourceOwnerAccount	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)	Sim

Criterion	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso
isPublic	Indica se a descoberta relata um recurso que tem uma política que permite o acesso público.	Booleano	 Sim	 Yes (Sim)	 Sim
findingType `UnusedIAMRole | UnusedIAMUserAccessKey | UnusedIAMUserPassword | UnusedPermission`	O tipo da descoberta do . É possível filtrar por tipo de descoberta somente para descobertas de acesso não utilizadas.	String	 Sim	 Yes (Sim)	 Sim
status `ACTIVE | ARCHIVED | RESOLVED`	O status atual da descoberta.	String	 No (Não)	 Yes (Sim)	 Sim
error	Indica o erro relatado para a descoberta.	String	 Sim	 Yes (Sim)	 Sim

Criterion	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso
principal.AWS	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)	 Sim
principal.Federated	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)	 Sim
condition.aws:PrincipalArn	O ARN da entidade de segurança (usuário, função 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)	 Sim

Criterion	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso
condition .aws:PrincipalOrgID	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)	 Sim
condition .aws:PrincipalOrgPaths	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)	 Sim
condition .aws:SourceIp	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)	 Sim

Criterion	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso
condition .aws:SourceVpc	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)	 Sim
condition .aws:UserId	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)	 Sim
condition .cognito-identity.amazonaws.com:aud	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)	 Sim

Criterion	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso
condition.graph.facebook.com:app_id	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)	 Sim
condition.accounts.google.com:aud	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)	 Sim

Criterion	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso
condition.kms:CallerAccount	O ID da conta da AWS que possui a entidade que faz a chamada (usuário, função 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)	 Sim
condition.www.amazon.com:app_id	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)	 Sim
id	O ID da descoberta.	String	 No (Não)	 Yes (Sim)	 Sim

Criterion	Descrição	Tipo	Regra de arquivamento	Listar descobertas	Listar descobertas de pré-visualização de acesso
changeType	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
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
existingFindingStatus	O status existente da descoberta, fornecido apenas para descobertas existentes na pré-visualização de acesso.	String	 No (Não)	 No (Não)	 Sim

Usar funções vinculadas ao 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, e essa política não pode ser anexada a nenhuma outra entidade do IAM.

Para obter informações sobre outros serviços compatíveis com funções vinculadas a serviços, consulte Serviços da AWS compatíveis com o IAM e procure os serviços que contenham Yes (Sim) na coluna Service-Linked Role (Função vinculada ao serviço). Escolha um Sim com um link para visualizar a documentação da função vinculada a esse serviço.

Permissões de função vinculada 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 uma função vinculada ao serviço. Para obter mais informações, consulte Permissões de função vinculada ao serviço no Guia do usuário do IAM.

Criar uma função vinculada a um serviço para o IAM Access Analyzer

Não é necessário criar manualmente uma função vinculada ao serviço. Quando você habilita o Access Analyzer no AWS Management Console 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 uma função vinculada ao 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 uma função vinculada ao serviço, você não poderá alterar o nome da função, pois várias entidades podem fazer referência a ela. No entanto, será possível editar a descrição da função usando o IAM. Para obter mais informações, consulte Editar uma função vinculada ao 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 recurso ou serviço que requer uma função vinculada a 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 recursos de sua função vinculada ao serviço antes de exclui-la manualmente.

> **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 por 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. Marque a caixa de seleção no canto superior esquerdo acima da lista de analisadores na tabela Analyzers (Analisadores) para selecionar todos os analisadores.
4. Escolha Delete (Excluir).
5. Para confirmar que deseja excluir os analisadores, insira **`delete`** e selecione Delete (Excluir).

Como excluir manualmente a função vinculada 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 obter mais informações, consulte Excluir uma função vinculada 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 do 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 Habilitar o IAM 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 Habilitar o IAM 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.

AWS KMS key policy (Política de chave do AWS KMS): se a configuração for para uma chave existente e você não especificar a política de chave, 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.

AWS KMS encryption key (Chave de criptografia do AWS KMS): se a configuração proposta for para um novo segredo e você não especificar o ID da chave do AWS KMS, 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. Você pode 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.

Tópicos

- Validação de política do IAM Access Analyzer

- Verificações de política personalizadas do IAM Access Analyzer

Validação de política do IAM Access Analyzer

Você pode 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. Você pode 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 de verificação de política do Access Analyzer.

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 AWS Management Console 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 gerenciadas pelo cliente (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 Verificações de política personalizadas 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 Avançar. 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. Escolha a opção 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 de verificação de política do Access Analyzer

Você pode 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. Você pode 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 Validação de política do IAM Access Analyzer.

Erro: ARN account not allowed (Conta do ARN não permitida)

No AWS Management Console, 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

- atributos de políticas

- Identificadores de conta
- ARNs de recursos
- Recursos do produto da AWS com formatos de ARN

Erro: ARN Region not allowed (Região do ARN não permitida)

No AWS Management Console, 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.

- atributos de políticas
- ARNs de recursos
- Recursos do produto da AWS com formatos de ARN

Erro: Data type mismatch (Incompatibilidade de tipo de dados)

No AWS Management Console, 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)

No AWS Management Console, 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
- Chaves de condição do produto da AWS

Erro: Invalid action (Ação inválida)

No AWS Management Console, 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
- Ações do produto da AWS

Políticas gerenciadas pela AWS 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)

No AWS Management Console, 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

- atributos de políticas
- Identificadores de conta
- ARNs de recursos
- Recursos do produto da AWS com formatos de ARN

Erro: Invalid ARN prefix (Prefixo do ARN inválido)

No AWS Management Console, 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

AWSOs ARNs de recurso devem incluir o prefixo `arn:` exigido.

Termos relacionados

- atributos de políticas
- ARNs de recursos
- Recursos do produto da AWS com formatos de ARN

Erro: Invalid ARN Region (Região do ARN inválida)

No AWS Management Console, 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

- atributos de políticas
- ARNs de recursos
- Nomes e códigos das regiões

Erro: Invalid ARN resource (Recurso do ARN inválido)

No AWS Management Console, 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

- atributos de políticas
- ARNs de recursos
- Recursos do produto da AWS com formatos de ARN

Erro: Invalid ARN service case (Caso de serviço de ARN inválido)

No AWS Management Console, 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

- atributos de políticas
- ARNs de recursos
- Recursos do produto da AWS com formatos de ARN

Erro: Invalid condition data type (Tipo de dados da condição inválido)

No AWS Management Console, 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
- Chaves de condição do produto da AWS

Erro: Invalid condition key format (Formato de chave de condição inválido)

No AWS Management Console, 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
- Chaves de condição do produto da AWS

Erro: Invalid condition multiple Boolean (Condição inválida múltipla booliana)

No AWS Management Console, 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
- Chaves de condição do produto da AWS

Erro: Invalid condition operator (Operador de condição inválido)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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.

Você pode especificar uma entidade 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)

No AWS Management Console, 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)

No AWS Management Console, a descoberta desta verificação inclui a seguinte mensagem:

```
Invalid Region: The Region {{region}} is not valid. Update the condition value to a
 suported 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
 suported 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

- atributos de políticas
- ARNs de recursos
- Nomes e códigos das regiões

Erro: Invalid service (Serviço inválido)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

Você pode 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

- atributos de políticas
- ARNs de recursos
- Recursos do produto da AWS com formatos de ARN

Erro: Missing ARN Region (Região do ARN ausente)

No AWS Management Console, 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

- atributos de políticas
- ARNs de recursos
- Nomes e códigos das regiões

Erro: Missing effect (Efeito ausente)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

Políticas gerenciadas pela AWS 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

Você pode 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)

No AWS Management Console, 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
- Políticas de controle de serviço do AWS Organizations
- Elementos de política JSON do IAM: Action

Erro: SCP syntax error allow condition (Erro de sintaxe de SCP ao permitir condição)

No AWS Management Console, a descoberta desta verificação inclui a seguinte mensagem:

```
SCP syntax error allow condition: SCPs do not support the Condition element with effect
 Allow. Update the element Condition or the effect.
```

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 the Condition element with effect Allow. Update
 the element Condition or the effect."
```

Resolver o erro

As políticas de controle de serviço (SCPs) do AWS Organizations oferecem suporte à especificação de valores no elemento `Condition` somente quando você usa `"Effect": "Deny"`.

Para permitir apenas uma única ação, você pode negar acesso a tudo, exceto à condição que você especificar usando a versão `...NotEquals`de um operador de condição. Isso nega a comparação feita pelo operador.

Termos relacionados

- Elemento Condition de SCP
- Avaliação do SCP
- Políticas de controle de serviço do AWS Organizations
- Política de exemplo: nega acesso à AWS com base na região solicitada
- Elementos de política JSON do IAM: operadores de condição

- Elementos de política JSON do IAM: Condition

Erro: SCP syntax error allow NotAction (Erro de sintaxe de SCP ao permitir NotAction)

No AWS Management Console, a descoberta desta verificação inclui a seguinte mensagem:

```
SCP syntax error allow NotAction: SCPs do not support NotAction with effect Allow.
 Update the element NotAction or the effect.
```

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 NotAction with effect Allow. Update the element
 NotAction or the effect."
```

Resolver o erro

As políticas de controle de serviço (SCPs) do AWS Organizations não são compatíveis com o uso do elemento `NotAction` com `"Effect": "Allow"`.

Você deve reescrever a lógica para permitir uma lista de ações ou para negar todas as ações que não estão listadas.

Termos relacionados

- Ação de SCP e elementos NotAction
- Avaliação do SCP
- Políticas de controle de serviço do AWS Organizations
- Elementos de política JSON do IAM: Action

Erro: SCP syntax error allow resource (Erro de sintaxe de SCP ao permitir recurso)

No AWS Management Console, a descoberta desta verificação inclui a seguinte mensagem:

```
SCP syntax error allow resource: SCPs do not support Resource with effect Allow. Update
 the element Resource or the effect.
```

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 Resource with effect Allow. Update the element
 Resource or the effect."
```

Resolver o erro

As políticas de controle de serviço (SCPs) do AWS Organizations oferecem suporte à especificação de valores no elemento `Resource` somente quando você usa `"Effect": "Deny"`.

Você deve reescrever a lógica para permitir todos os recursos ou para negar todos os recursos listados.

Termos relacionados

- Elemento Resource de SCP
- Avaliação do SCP
- Políticas de controle de serviço do AWS Organizations
- Elementos de política JSON do IAM: Resource

Erro: SCP syntax error NotResource (Erro de sintaxe de SCP para NotResource)

No AWS Management Console, a descoberta desta verificação inclui a seguinte mensagem:

```
SCP syntax error NotResource: SCPs do not support the NotResource element. Update the
 policy to use Resource instead.
```

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 the NotResource element. Update the policy to
 use Resource instead."
```

Resolver o erro

As políticas de controle de serviço (SCPs) do AWS Organizations não oferecem suporte ao elemento `NotResource`.

Você deve reescrever a lógica para permitir todos os recursos ou para negar todos os recursos listados.

Termos relacionados

- Elemento Resource de SCP
- Avaliação do SCP
- Políticas de controle de serviço do AWS Organizations
- Elementos de política JSON do IAM: Resource

Erro: SCP syntax error principal (Erro de sintaxe de SCP para a entidade)

No AWS Management Console, 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`.

Você pode 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)

No AWS Management Console, 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. Você pode 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

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

A AWS 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

- Chaves contextuais de condições globais da AWS

Erro: Unsupported principal (Sem suporte para a entidade)

No AWS Management Console, 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. Você pode 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

- Elementos de política JSON da AWS: entidade
- Acesso a recursos entre contas no IAM

Erro: ARN de recurso incompatível na política

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

- Elementos de política JSON da AWS: entidade

Erro: Missing brace in variable (Chave ausente na variável)

No AWS Management Console, 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)

No AWS Management Console, 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 `DOC-EXAMPLE-BUCKET` do Amazon S3 de nome `DOC-EXAMPLE-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 `DOC-EXAMPLE-BUCKET-company-wide`, onde podem visualizar informações gerais, como instruções para ingressar em uma equipe.

```
"Resource":"arn:aws:s3:::DOC-EXAMPLE-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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

Você pode 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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
- Políticas de controle de serviço do AWS Organizations

Erro: Invalid service principal format (Formato de entidade de serviço inválido)

No AWS Management Console, 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. Você pode 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
- Chaves de condições globais da AWS
- Chave da condição `kms:ViaService`

Erro: Missing tag key in condition (Chave de etiqueta ausente na condição)

No AWS Management Console, 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
- Chaves de condição do produto da AWS

Erro: formato de vpc inválido

No AWS Management Console, 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

- Chaves globais de condição da AWS: aws:SourceVpc

Erro: formato de vpce inválido

No AWS Management Console, 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

- Chaves globais de condição da AWS: aws:SourceVpce

Erro: a entidade principal federada é incompatível

No AWS Management Console, 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

No AWS Management Console, 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

No AWS Management Console, 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

No AWS Management Console, 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

No AWS Management Console, 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

Você pode especificar os 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. 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

Aviso geral: Create SLR with NotResource (Crie uma SLR com NotResource)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

Políticas gerenciadas pela AWS 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)

No AWS Management Console, 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 AWSrecomenda que você especifique ARNs permitidos no elemento `Resource`.

- Operação CreateServiceLinkedRole
- Elementos de política JSON do IAM: Action
- Elementos de política JSON do IAM: Resource

Políticas gerenciadas pela AWS 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

A AWS 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)

No AWS Management Console, 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

A AWS 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)

No AWS Management Console, 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)

No AWS Management Console, 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

A AWS 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. Você pode 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)

No AWS Management Console, 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

Políticas gerenciadas pela AWS 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)

No AWS Management Console, 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

Você pode anexar até dez políticas gerenciadas a uma identidade do IAM (usuário, grupo de usuários ou função). 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

Políticas gerenciadas pela AWS 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

No AWS Management Console, 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 atributos
- 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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.
```

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."
```

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. Você pode 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.

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)

No AWS Management Console, 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 de segurança: Allow with NotPrincipal (Permitir com NotPrincipal)

No AWS Management Console, 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”`.

- Elementos de política JSON da AWS: entidade
- Elementos de política JSON da AWS: NotPrincipal

Aviso de segurança: ForAllValues with single valued key (ForAllValues com chave de valor único)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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

Políticas gerenciadas pela AWS 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 AWSque 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)

No AWS Management Console, 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 AWSrecomenda 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

Políticas gerenciadas pela AWS 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)

No AWS Management Console, 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)

No AWS Management Console, 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

- Chave de condição global da `aws:VpcSourceIp`
- Chave de condição global da `aws:SourceVPC`
- 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)

No AWS Management Console, 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 AWSrecomenda 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)

No AWS Management Console, 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 ter 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

No AWS Management Console, 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

Você pode especificar os Serviços da AWS no elemento `Principal` de uma política baseada em recursos usando a entidade principal de um serviço, que é um identificador desse serviço. Ao conceder acesso à entidade principal de serviço para atuar em seu nome, restrinja o acesso. Você pode 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

- Entidades principais do AWS service (Serviço da AWS)
- Chaves globais de condição da AWS: aws:SourceAccount
- Chaves globais de condição da AWS: aws:SourceArn
- Chaves de condições globais da AWS: aws:SourceOrgId
- Chaves de condições globais da AWS: aws:SourceOrgPaths
- O problema de "confused deputy"

Aviso de segurança: Missing condition keys for oidc principal (Chaves de condição ausentes para entidade principal do oidc)

No AWS Management Console, 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)

No AWS Management Console, 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. Você pode 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

Sugestão: Empty array action (Ação de matriz vazia)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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 um elemento `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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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. Você pode 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: Improve IP range (Melhorar o intervalo IP)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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. Você pode 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

Políticas gerenciadas pela AWS 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)

No AWS Management Console, 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). Você pode remover o valor `2700` redundante.

- Operadores de condição numéricos
- Elementos de política JSON do IAM: Condition

Sugestão: Redundant resource (Recurso redundante)

No AWS Management Console, 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`. Você pode 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

Políticas gerenciadas pela AWS 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)

No AWS Management Console, 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. Você pode 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)

No AWS Management Console, 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)

No AWS Management Console, 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)

No AWS Management Console, 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 ter 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

No AWS Management Console, 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

- Entidades principais do AWS service (Serviço da AWS)
- Chaves globais de condição da AWS: aws:SourceAccount
- Chaves globais de condição da AWS: aws:SourceArn
- Chaves de condições globais da AWS: aws:SourceOrgId
- Chaves de condições globais da AWS: aws:SourceOrgPaths
- O problema de "confused deputy"

Sugestão: chave de condição irrelevante na política

No AWS Management Console, 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.

Você pode usar a chave de condição em uma política baseada em identidade para controlar o acesso aos recurso.

Termos relacionados

- Políticas baseadas em identidade e em recurso

Sugestão: entidade principal redundante na política de confiança da função

No AWS Management Console, 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: confirme o tipo de reivindicação do público

No AWS Management Console, a descoberta desta verificação inclui a seguinte mensagem:

```
Confirm audience claim type: The 'aud' (audience) claim key identifies the recipients
 that the JSON web token is intended for. Audience claims can be multivalued or single-
valued. If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If
 the 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 'aud' (audience) claim key identifies the recipients that the
 JSON web token is intended for. Audience claims can be multivalued or single-valued.
 If the claim is multivalued, use a ForAllValues or ForAnyValue qualifier. If the 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

Verificações de política personalizadas do IAM Access Analyzer

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. Há dois tipos de verificações de políticas personalizadas que você pode executar:

- 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.
- Compare uma lista de ações do IAM: você pode verificar para garantir que ações específicas do IAM não são permitidas pela sua política. É possível executar essa verificação ao criar ou editar uma política usando a AWS CLI ou a API.

Uma cobrança é associada a cada verificação de política personalizada. Para obter mais detalhes sobre preços, consulte Preços do IAM Access Analyzer.

Como funcionam as verificações de políticas personalizadas

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 ser interpretado 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`.

Inspecionando falhas nas verificações de políticas personalizadas

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 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 AWS Management Console 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. Você pode 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

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.

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

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 uma função de serviço existente ou criar uma nova. A função 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 logs dos eventos 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 AWS Management Console, 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 recurso 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 para AWS Control Tower: a geração de políticas não oferece suporte ao AWS Control Tower para a geração de políticas.

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.

Configuração pela primeira vez

Ao gerar uma política pela primeira vez, você deve escolher uma função de serviço existente adequada em sua conta ou criar uma nova função de serviço. A função de serviço dá 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 a função de serviço durante a primeira configuração. Para saber mais sobre as permissões necessárias para criar funções de serviço, consulte Criar uma função para delegar permissões a um produto da AWS.

Permissões necessárias para a função de serviço

Ao criar uma função de serviço, você configura duas políticas para a função. 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 o principal que pode usar a função.

O primeiro exemplo de política mostra a política de permissões para a função de serviço necessária para gerar uma política. O segundo exemplo de política mostra a política de confiança da função que é necessária para a função de serviço. Você pode usar essas políticas para ajudar na criação de uma função de serviço ao usar a API AWS ou AWS CLI para gerar uma política. Ao usar o console do IAM para criar uma função de serviço como parte do processo de geração de políticas, nós geramos essas políticas para você.

```
{
    "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:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "access-analyzer.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Usos subsequentes

Para gerar políticas no AWS Management Console, um usuário do IAM deve ter uma política de permissões que permita transmitir a função de serviço usada 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 da função a ser transmitida. 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 funções do IAM

para produtos da AWS, consulte Conceder permissões a um usuário para passar uma função para um produto da AWS.

```
{
  "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 AWS Management Console, 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 AWS Management Console 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 função 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 AWS Management Console.

Gerar uma política para uma função do IAM

1. Faça login no AWS Management Console 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 logs de eventos 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).

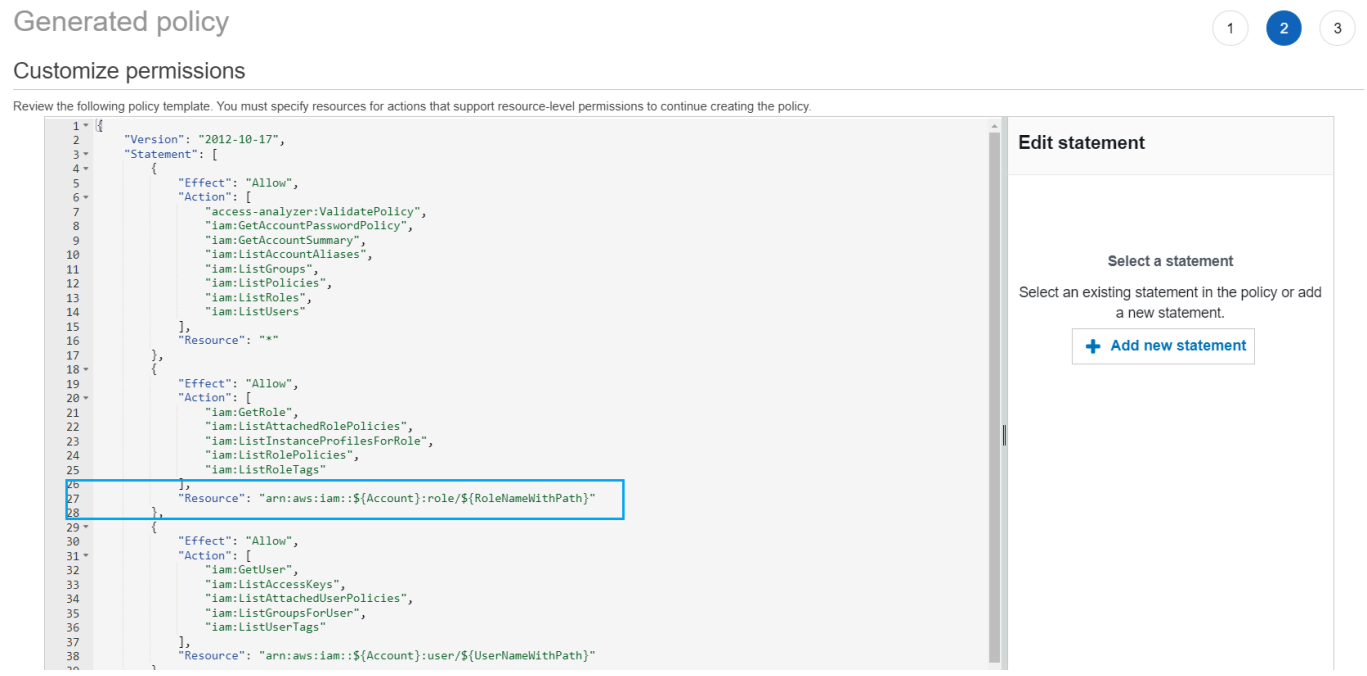
Etapa 3: personalizar ainda mais a política gerada

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 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 Attach policy to ***seuRolename*** (Anexar política ao) (seu nome completo). 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 uma função existente ou crie uma nova função 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 uma nova função 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 AWS Management Console 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 `DOC-EXAMPLE-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`. O uso de `aws:PrincipalArn` como uma `Condition` no elemento `Resource` garante que a função só possa acessar a atividade da conta se ela pertencer à conta A. É possível substituir `DOC-EXAMPLE-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:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/optional-prefix/AWSLogs/organization-id/
${aws:PrincipalAccount}/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "organization-id"
        },
        "StringLike": {
          "aws:PrincipalArn": "arn:aws:iam::${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"
          ]
          "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.

Serviço	Prefixo do serviço
AWS Identity and Access Management Access Analyzer	access-analyzer
AWS Account Management	conta
AWS Certificate Manager	acm
Amazon Managed Workflows for Apache Airflow	airflow
AWS Amplify	amplify
AWS Amplify UI Builder	amplifyuibuilder

Serviço	Prefixo do serviço
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 AppStream 2.0	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	batch (lote)
Amazon Braket	braket
AWS Budgets	orçamentos
AWS Cloud9	cloud9
AWS CloudFormation	cloudformation

Serviço	Prefixo do serviço
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
Notificações do AWS CodeStar	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
AWS Compute Optimizer	compute-optimizer
AWS Config	config
Amazon Connect	conectar

Serviço	Prefixo do serviço
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
AWS Directory Service	ds
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
Amazon Elastic Kubernetes Service	eks
Amazon Elastic Inference	elastic-inference

Serviço	Prefixo do serviço
Amazon ElastiCache	elasticache
AWS Elastic Beanstalk	elasticbeanstalk
Amazon Elastic File System	elasticfilesystem
Elastic Load Balancing	elasticloadbalancing
Amazon Elastic Transcoder	elastictranscoder
Amazon EMR em EKS (EMR Containers)	emr-containers
Amazon EMR Serverless	emr-serverless
Amazon OpenSearch Service	es
Amazon EventBridge	eventos
Amazon CloudWatch Evidently	evidently
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	gamelift
Amazon Location Service	geo
Amazon S3 Glacier	glacier
Amazon Managed Grafana	grafana

Serviço	Prefixo do serviço
AWS IoT Greengrass	greengrass
AWS Ground Station	groundstation
Amazon GuardDuty	guardduty
AWS HealthLake	healthlake
Amazon Honeycode	honeycode
AWS Identity and Access Management	iam
Armazenamento de identidades do AWS	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

Serviço	Prefixo do serviço
Amazon Managed Streaming para 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	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
Amazon Lookout for Vision	lookoutvision
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 MediaStore	mediastore
AWS Elemental MediaTailor	mediatailor
Amazon MemoryDB para Redis	memorydb
AWS Application Migration Service	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 Panorama	panorama
AWS Performance Insights	pi
Amazon EventBridge Pipes	pipes
Amazon Polly	polly
Amazon Connect Customer Profiles	profile

Serviço	Prefixo do serviço
Amazon QLDB	qldb
AWS Resource Access Manager	ram
AWSLixeira	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
AWS Resource Groups	resource-groups
AWS RoboMaker	robomaker
AWS Identity and Access Management Roles Anywhere	rolesanywhere
Amazon Route 53	route53
Amazon Route 53 Recovery Controls	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

Serviço	Prefixo do serviço
Amazon S3 on Outposts	s3-outposts
Recursos geoespaciais do Amazon SageMaker	sagemaker-geospatial
Savings Plans	savingsplans
Amazon EventBridge Schemas	schemas
Amazon SimpleDB	sdb
AWS Secrets Manager	secretsmanager
AWS Security Hub	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	snowball
Amazon Simple Queue Service	sqs

Serviço	Prefixo do serviço
AWS Systems Manager	ssm
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
Amazon Connect Voice ID	voiceid
Amazon VPC Lattice	vpc-lattice
AWS WAFV2	wafv2
AWS Well-Architected Tool	wellarchitected
Amazon Connect Wisdom	wisdom
Amazon WorkLink	worklink

Serviço	Prefixo do serviço
Amazon WorkSpaces	espaços de trabalho
AWS X-Ray	xray

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 nível organizacional por tipo de analisador por região da Conta da AWS	5	20[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
Arquivos de log do AWS CloudTrail processados por gerações de política	100.000	100.000

Recurso	Cota padrão	Cota máxima
Gerações simultâneas de política	1	1
Tamanho dos dados de geração de política do AWS CloudTrail	25 GB	25 GB
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 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 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.

Solução de problemas do IAM

Se você encontrar problemas de acesso negado ou dificuldades semelhantes ao trabalhar com o AWS Identity and Access Management (IAM), consulte os tópicos nesta seção.

Tópicos

Solução de problemas gerais 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

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. 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.

Para obter mais informações, consulte Redefinição de senhas perdidas ou esquecidas ou chaves de acesso para a AWS.

As variáveis da política não estão funcionando

- 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. Todas as políticas que não incluírem variáveis ainda funcionarão se você incluir 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. 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”.
- Verifique se as variáveis de política estão no caso certo. Para obter mais 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 datacenters em todo o mundo, o IAM usa um modelo de computação distribuído chamado consistência final. Qualquer alteração feita no IAM (ou outros serviços da AWS), incluindo etiquetas usadas no attribute-based access control (ABAC – Controle de acesso baseado em atributo), leva tempo para se tornar visível em todos os endpoints possíveis. Parte do atraso resulta do tempo necessário para enviar os dados de um servidor para outro, de uma zona de replicação para outra e de uma região para outra em todo o mundo. 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 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 sobre como alguns outros serviços da AWS são afetados por isso, consulte os seguintes recursos:

- Amazon DynamoDB: Qual é o modelo de consistência do Amazon DynamoDB? nas Perguntas frequentes sobre o DynamoDB e Consistência de leitura 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 Elastic MapReduce for ETL Workflows no AWS Big Data Blog
- 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. Você deve excluir o dispositivo de 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`
   - API da AWS: `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`
   - API da AWS: `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 a seguir para criar um novo usuário com segurança no IAM:

1. Crie um novo usuário usando o AWS Management Console. Conceda acesso ao AWS Management Console 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 ver 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.

Solução de problemas de mensagens de erro de acesso negado

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 ter mais informações, consulte A diferença entre negações explícitas e implícitas.

Se várias políticas do mesmo tipo de política negarem uma solicitação de autorização, a AWS não especificará o número de políticas na mensagem de erro de acesso negado. 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 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 AWS Support. 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 de política responsável por negar o acesso. Por exemplo, se o erro mencionar que o acesso é negado devido a uma política de controle de serviço (SCP), você poderá se concentrar na solução de problemas de SCP. Quando você conhece o tipo de política, também pode verificar se há uma instrução de negação ou se falta uma permissão na ação específica em políticas desse tipo de política. 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 Gerenciamento de políticas do IAM.
- Se o AWS Management Console 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 Serviços da AWS 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 ter 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 de segurança do IAM (usuário ou função) 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.

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 ter 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 Serviços da AWS 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 Uso de credenciais temporárias com recursos da AWS.
- Verifique se suas credenciais de segurança temporárias não expiraram. Para ter 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 Controle de permissões para 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. Você pode passar um único documento de política JSON de sessão em linha usando o parâmetro `Policy`. Você

pode 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 um usuário federado, a sessão poderá ser limitada pelas políticas de sessão. Você se torna um usuário federado fazendo login na AWS como um usuário do IAM e solicitando um token de federação. Para obter mais informações sobre usuários federados, consulte GetFederationToken: federação por meio de um agente 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 `MyBucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "S3BucketPolicy",
    "Effect": "Allow",
    "Principal": {"AWS": ["arn:aws:iam::111122223333:role/MyRole"]},
    "Action": ["s3:PutObject"],
    "Resource": ["arn:aws:s3:::MyBucket/*"]
  }]
}
```

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, *`user`* (usuário) é o nome do recurso da Amazon (ARN) que não recebe acesso, *`action`* (ação) é a ação de serviço que a política nega e *`resource`* (recurso) é o ARN do recurso no qual a política atua. O campo *`context`* (contexto) representa contexto adicional sobre o tipo de política que explica por que o acesso é negado.

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. 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::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories because no service control policy allows the
 codecommit:ListRespositories action
```

Acesso negado devido a uma política de controle de serviço: negação explícita

1. Verifique se há uma instrução `Deny` para a ação em suas políticas de controle de serviços (SCPs). No exemplo a seguir, a ação é `codecommit:ListRepositories`.
2. Atualize a SCP removendo a instrução `Deny`. Para obter mais informações, consulte Atualizar uma SCP no Guia do usuário do AWS Organizations.

```
User: arn:aws:iam::777788889999:user/JohnDoe is not authorized to perform:
codecommit:ListRepositories with an explicit deny in a service 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/JohnDoe 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/JohnDoe 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 Edição de políticas do IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe 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. Verifique se há uma instrução `Deny` explícita para a ação em seu limite de permissões. No exemplo a seguir, a ação é `sagemaker:ListModels`.
2. 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 Edição de políticas do IAM.

```
User: arn:aws:iam::777788889999:user/JohnDoe 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 Edição de políticas do IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe 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. Verifique se há uma instrução `Deny` explícita para a ação em suas políticas de sessão. No exemplo a seguir, a ação é `codedeploy:ListDeployments`.
2. Atualize a política de sessão removendo a instrução `Deny`. Para obter mais informações, consulte Políticas de sessão e Edição de políticas do IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe is not authorized to perform:
codedeploy:ListDeployments on resource: arn:aws:codedeploy:us-
east-1:123456789012:deploymentgroup:* with an explicit deny in a sessions 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 Edição de políticas do IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe 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 Edição de políticas do IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe 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 Edição de políticas do IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe 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 Edição de políticas do IAM.

```
User: arn:aws:iam::777788889999:user/JohnDoe 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. Para o exemplo a seguir, a ação é `codecommit:ListRepositories` anexada ao usuário `JohnDoe`.
2. Atualize a política adicionando a instrução `Allow`. Para obter mais informações, consulte Políticas baseadas em identidade e Edição de políticas do IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe 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. Verifique se há uma instrução `Deny` explícita para a ação em políticas baseadas em identidade anexadas à identidade. Para o exemplo a seguir, a ação é `codedeploy:ListDeployments` anexada ao usuário `JohnDoe`.
2. Atualize a política removendo a instrução `Deny`. Para obter mais informações, consulte Políticas baseadas em identidade e Edição de políticas do IAM.

```
User: arn:aws:iam::123456789012:user/JohnDoe 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
```

Acesso negado quando uma solicitação de VPC falha devido a outra política

1. Verifique se há uma instrução `Deny` explícita para a ação em suas políticas de controle de serviços (SCPs). No exemplo a seguir, a ação é `SNS:Publish`.
2. Atualize a SCP removendo a instrução `Deny`. Para obter mais informações, consulte Atualizar uma SCP no Guia do usuário do AWS IAM Identity Center.

```
User: arn:aws:sts::111122223333:assumed-role/role-name/role-session-name is not
 authorized to perform:
SNS:Publish on resource: arn:aws:sns:us-east-1:444455556666:role-name-2
with an explicit deny in a VPC endpoint policy transitively through a service control
 policy
```

Solução de problemas de políticas 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. Você pode anexar uma política baseada em identidade a uma entidade (ou identidade), como um grupo, 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 criar e editar políticas gerenciadas pelo cliente no AWS Management Console usando ambas as opções de editor Visual e JSON. Quando você visualiza uma política no AWS Management Console, 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 AWS Management Console. 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

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 AWS apresenta a política em resposta a uma consulta de usuário ou a exibe no

console, a AWS converte a política de volta a um formato legível sem alterar as permissões concedidas por ela. Isso pode resultar em diferenças no que diz respeito ao que você vê no editor visual de política ou na guia JSON: no editor visual, blocos de permissões podem ser adicionados, removidos ou reordenados, e o conteúdo em 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 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 AWS Management Console, você pode optar por trabalhar exclusivamente no editor JSON. Se você nunca fizer alterações no editor Visual e escolher Avançar no editor JSON, a probabilidade de que a política seja reestruturada é menor. Porém, se você criar uma política e usar o editor Visual para fazer qualquer modificação ou escolher Avançar na opção de editor Visual, o IAM poderá reestruturar a política para otimizar sua aparência no editor visual.

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

Usar essa opção de editor	Se você editar a política	E depois escolher Avançar nesta guia	Quando você escolher Salvar alterações
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.

- Usar o gerador de ARN: dependendo do tipo de recurso, você pode ver campos diferentes para criar 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`. Será preciso fornecer os valores Região, Conta e InstanceId para o recurso. Se você fornecer o ID de sua conta mas escolher Any (Qualquer) para a região e para o ID da instância, a política concederá permissões para qualquer instância em sua conta.
- 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 como melhor prática de segurança que você conceda permissões somente às ações e aos recursos dos quais o usuário precisa. Você só precisa criar uma instrução para negar permissões, se quiser 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. 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.

Se você estiver familiarizado com a sintaxe JSON, também poderá 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 e não permitirá que você revise e salve sua 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, e você pode usar essa informação para ajudá-lo a reduzir o tamanho da política.

Corrigir serviços, ações ou tipos de recursos desconhecidos no editor visual

Quando você cria ou edita uma política no editor visual, pode receber um aviso de que a política inclui um serviço, ação ou tipo de recurso desconhecido.

> **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 você estiver participando da pré-visualização, poderá ignorar o aviso e prosseguir. Entretanto, você deverá inserir manualmente as ações e os ARNs dos recursos para concluir sua política. 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 você estiver usando um serviço personalizado, poderá ignorar o aviso e prosseguir. Entretanto, você deverá inserir manualmente as ações e os ARNs dos recursos para concluir sua política. Ou então, você pode escolher a opção de editor JSON e digitar ou colar um documento de política JSON.
- O serviço não oferece suporte ao editor visual: se sua política incluir um serviço disponível para o público (GA) que não oferece suporte ao editor visual, você poderá ignorar o aviso e prosseguir. Entretanto, você deverá digitar manualmente as ações e os ARNs dos recursos para concluir sua política. 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 oferece suporte ao editor visual: se sua política incluir um serviço compatível com uma ação incompatível, você poderá ignorar o aviso e prosseguir. Entretanto, você deverá digitar manualmente os ARNs dos recursos para concluir sua política. 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. Para evitar isso, convém usar editor visual, selecionando entre os serviços e as ações na lista e, em seguida, conclua a seção de recursos de acordo com os prompts. No entanto, se um serviço não oferece suporte total ao editor visual, pode ser necessário que você insira 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 a ortografia de nomes de serviço, ação e tipos de recurso. Por exemplo, você pode usar `s2` em vez de `s3` e `ListMyBuckets` em vez de `ListAllMyBuckets`. Outro erro de ortografia comum é a inclusão de texto desnecessário nos ARNs, como, por exemplo, `arn:aws:s3: : :*`ou a falta de pontuação, como `iam.CreateUser`. Você pode avaliar uma política que pode incluir erros de digitação selecionando Avançar para revisar o resumo da política e confirmar se a política fornece as permissões que você pretendia.

Solução de problemas usando 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. Você pode visualizar o resumo da política de qualquer política anexada a uma entidade na página Detalhes da política daquela política. Visualize o resumo de políticas para políticas gerenciadas na página Políticas. Se a AWS for incapaz de renderizar um resumo para uma política, você verá o documento da política JSON, em vez do resumo, e receberá a seguinte mensagem de 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**
>
> É necessário 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 elementos de política JSON do IAM.

Você pode criar uma política que não fornece permissões eficazes se você fornecer serviços e recursos incompatíveis. Isso pode ocorrer se você especificar 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. Para compreender melhor suas políticas, sempre teste-as com o simulador de políticas.

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, uma ação ou um 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

Quando um serviço disponível para o público (GA) ou uma ação não é reconhecido pelos resumos de políticas do IAM ou pelo editor visual, é possível que o serviço não ofereça suporte a esses recursos. 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 AWS Management Console 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 AWS Management Console, 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 AWS Management Console 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 AWS Management Console, 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 elementos 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.

Para saber se sua política define uma ação, um recurso ou uma condição que não concede permissões, você pode visualizar o resumo da política para a 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 suportam permissões no nível de recurso, consulte Serviços da AWS que funcionam com o IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "ec2:Describe*",
```

```
        "Resource": "arn:aws:ec2:us-east-2:ACCOUNT-ID: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`.

```
{
    "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.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "cloudfront:*",
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::examplebucket"
        ]
    }]
}
```

Esta política fornece permissões para todas as ações do CloudFront. Mas como a política define o recurso do S3 `examplebucket` 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.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "cloudfront:*",
            "s3:CreateBucket",
            "s3:ListBucket*",
            "s3:PutBucket*",
            "s3:GetBucket*"
        ],
        "Resource": [
            "arn:aws:cloudfront:*",
            "arn:aws:s3:::examplebucket"
        ]
    }]
}
```

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 Ações, recursos e chaves de condição de serviços da AWS e siga o link para a documentação do serviço de chaves de condição.

```
{
    "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`.

```
{
    "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`.

```
{
    "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 de uma 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. Você pode descobrir que um usuário ou função está tentando executar uma ação na AWS e essa solicitação está sendo negada. Nesse caso, você pode considerar que o usuário ou a função tem permissão para executar 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. Opcionalmente, se o usuário ou a função tiver permissões que eles não estão usando, você poderá considerar remover essas permissões da sua política. 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 políticas 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 AWS Management Console, 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 apenas um único objeto JSON. Você denota um objeto colocando chaves { } em torno. Embora você possa aninhar outros objetos dentro de um objeto JSON incorporando { } adicionais dentro do par de chaves externas, uma política pode conter apenas um par mais externo de { } chaves. O exemplo a seguir é incorreto, pois contém dois objetos no nível superior (destacados em *vermelho*):

```
{
      "Version": "2012-10-17",
      "Statement":
      {
         "Effect":"Allow",
         "Action":"ec2:Describe*",
         "Resource":"*"
      }
   }
   {
      "Statement": {
         "Effect": "Allow",
         "Action": "s3:*",
```

```
        "Resource": "arn:aws:s3:::my-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):

```
{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":" *"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::my-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`*):

```
{
```

```
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "ec2:Describe*",
        "Resource": "*"
      },
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-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):

```
{
      "Version": "2012-10-17",
      "Statement": [    {
          "Effect": "Allow",
          "Action": "ec2:Describe*",
          "Resource":"*"
        },
        {
          "Effect": "Allow",
          "Action": "s3:*",
          "Resource": "arn:aws:s3:::my-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` no objeto de valor de `Statement`:

```
{
      "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 única lista.
> - Elementos `Resource` múltiplos: todos os elementos `Resource` são combinados internamente e tratados como se fossem uma única lista.
>
> O mecanismo da política não permite salvar qualquer política com erros de sintaxe. Você deve corrigir os erros na política antes de salvá-la. Recomendamos revisar e corrigir todas as recomendações de validação de política para as 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:

```
{
      "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*):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": "arn:aws:s3:::my-bucket",
        "Resource": "arn:aws:s3:::my-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):

```
{
      "Version": "2012-10-17",
      "Statement": {
        "Effect": "Allow",
        "Action": "s3:*",
        "Resource": [
          "arn:aws:s3:::my-bucket",
          "arn:aws:s3:::my-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. 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”.

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):

```
{
  "Version": "2012-10-17",
  "Statement":
  {
    "Action": "iam:*AccessKey*",
    "Effect": "Allow",
    "Resource": "arn:aws:iam::123456789012:user/${aws:username}"
  }
}
```

Solução de problemas de 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

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?

  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 segurança FIDO.

- Você está usando o Mozilla Firefox?

  As versões atuais do Firefox são compatíveis com WebAuthn por padrão. Para habilitar a compatibilidade com WebAuthn no Firefox, faça o seguinte:

  1. Na barra de endereços do Firefox, digite **`about:config`**.
  2. Na barra de pesquisa da tela que é aberta, digite **`webauthn`**.
  3. Escolha security.webauth.webauthn e altere o valor para true.

- Você está usando algum plug-in de navegador?

  A AWS 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. Você deve desabilitar todos os plug-ins que podem ser incompatíveis e reiniciar 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

Se você for um administrador e os usuários do IAM não conseguirem habilitar as chaves de segurança FIDO apesar de usarem uma configuração compatível, certifique-se de que eles tenham as permissões apropriadas. 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 é um usuário do IAM e não consegue fazer login no AWS Management Console usando a chaves de segurança FIDO, primeiro consulte Configurações compatíveis com o uso de chaves de segurança FIDO. 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 AWS Management Console. Substituir uma chave de segurança FIDO é semelhante a substituir um token de hardware TOTP. Para obter informações sobre o que fazer se você perder ou quebrar qualquer tipo de dispositivo MFA, consulte O que acontece se um dispositivo com MFA for perdido ou parar de funcionar?.

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.

Solução de problemas das funções 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

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 Modificação de uma política de confiança de função (console).

- Ao assumir uma função usando o AWS Management Console, 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.
- 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 “Recursos de etiquetas 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 Usar funções vinculadas 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 ao serviço, consulte Serviços da AWS 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 ao serviço, consulte Serviços da AWS 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 Serviços da AWS 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 `AssumeRole*` ou as operações da CLI `assume-role*` do AWS STS 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 Visualizar a configuração de duração máxima da sessão para uma função.

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. Você pode passar um único documento de política JSON de sessão em linha usando o parâmetro `Policy`. Você pode 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 AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha 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 ter 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 a função.

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 escopo 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 Serviços da AWS 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 ao 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 Criação de uma função para delegar 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.

Solução de problemas do IAM e Amazon EC2

Use as informações contidas aqui para ajudar você a solucionar problemas e corrigir acesso negado ou outros problemas que você pode encontrar ao trabalhar com o Amazon EC2 e o IAM.

Tópicos

- Ao tentar iniciar uma instância, não vejo a função que esperava ver na lista de funções 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 uma função, 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 a função que esperava ver na lista de funções 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 informações sobre as permissões necessárias para trabalhar com funções, consulte “Permissões necessárias para usar funções com o Amazon EC2” em Uso de uma função do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2. Para obter informações sobre como adicionar permissões a um usuário, consulte Gerenciamento de 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 uma função usando a CLI ou a API do IAM, verifique se criou um perfil da instância e adicionou a função a esse perfil. Além disso, se você nomear sua função e o perfil da instância de forma diferente, não verá o nome correto da função na lista de funções do IAM 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 funções, consulte “Como faço para começar?” no Uso de uma função do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2. Para obter informações sobre como adicionar permissões a um usuário, consulte Gerenciamento de políticas do IAM.

Amazon EC2: Quando tento iniciar uma instância com uma função, 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 Criação de funções do IAM.

Para obter mais informações sobre as permissões necessárias para trabalhar com funções, consulte "Como faço para começar?" no Uso de uma função do IAM para conceder permissões a aplicações em execução em instâncias do Amazon EC2. Para obter informações sobre como adicionar permissões a um usuário, consulte Gerenciamento de 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 Uso de uma função 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.

```
{"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.

Solução de problemas do IAM e do 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 usuário raiz da Conta da AWS. Por que não consigo acessar um bucket do Amazon S3 na 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 conta da AWS como entidade principal, o usuário raiz terá o acesso negado ao 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" }
```

Solução de problemas da federação SAML 2.0 com a AWS

Use estas informações para ajudar você a diagnosticar e corrigir problemas que encontrar ao trabalhar com o SAML 2.0 e a federação com o IAM.

Tópicos

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

Para ter mais informações, consulte Configurar declarações SAML para a resposta de autenticação. Para visualizar a resposta do SAML no navegador, siga as etapas listadas em Como visualizar uma resposta do SAML no navegador para solução de problemas.

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 ter mais informações, consulte Configurar declarações SAML para a resposta de autenticação. Para visualizar a resposta do SAML no navegador, siga as etapas listadas em Como visualizar uma resposta do SAML no navegador para solução de problemas.

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 pode ocorrer se os usuários federados não tiverem permissões para assumir a função. 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 Estabelecer permissões na AWS para usuários federados e Configurar declarações SAML para a resposta de autenticação. Para visualizar a resposta do SAML no navegador, siga as etapas listadas em Como visualizar uma resposta do SAML no navegador para solução de problemas.

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 ter mais informações, consulte Configurar declarações SAML para a resposta de autenticação. Para visualizar a resposta do SAML no navegador, siga as etapas listadas em Como visualizar uma resposta do SAML no navegador para solução de problemas.

Erro: A identidade-fonte deve corresponder a [a-zA-Z_0-9+=,.@-]{2,64} 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 64 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 Como visualizar uma resposta do SAML no navegador para solução de problemas.

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: 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 AWS Management Console, 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” (Não foi possível analisar metadados).

Conforme definido pelo Perfil de Interoperabilidade de Metadados SAML V2.0 Versão 1.0, o IAM não avalia nem toma medidas em relação à expiração do certificado X.509 do documento de metadados.

Erro: o provedor especificado não existe.

Esse erro poderá ocorrer se o nome do provedor que você especifica na declaração SAML não corresponder ao nome do provedor configurado 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 Visualizar a configuração de duração máxima da sessão para uma função.

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 visualizar uma resposta do SAML no navegador para solução de problemas

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

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 Method (Método) não estiver visível no painel de log Network (Rede) de Developer Tools (Ferramentas de desenvolvimento), clique com o botão direito do mouse em qualquer rótulo de coluna e escolha Method (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 Method (Método) não estiver visível no painel de log Network (Rede) de Web Developer Tools (Ferramentas de desenvolvimento na Web), clique com o botão direito do mouse em qualquer rótulo de coluna e escolha Method (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 Method (Método) não estiver visível no painel de log Network (Rede) de Web Inspector (Inspetor da Web), clique com o botão direito do mouse em qualquer rótulo de coluna e escolha Method (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

Assim que você encontrar o elemento de resposta do SAML codificado em Base64 no navegador, copie e use sua ferramenta de decodificação Base-64 de sua preferência para extrair a resposta marcada do 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
```

Informações de referência do 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

Nomes de recurso da Amazon (ARN)

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.

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/*`johndoe`*

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 (`*`). Por exemplo, se você estiver escrevendo uma política do IAM, poderá especificar todos os usuários do IAM que tenham o caminho `product_1234` usando um curinga desta maneira:

```
arn:aws:iam::123456789012:user/Development/product_1234/*
```

Da mesma forma, poderá especificar `user/*` para indicar todos os usuários ou `group/*` para indicar todos os grupos, como nos exemplos a seguir:

```
"Resource":"arn:aws:iam::123456789012:user/*"
"Resource":"arn:aws:iam::123456789012:group/*"
```

O exemplo a seguir mostra ARNs para um bucket do Amazon S3 em que o nome do recurso inclui um caminho:

```
arn:aws:s3:::my_corporate_bucket/*
arn:aws:s3:::my_corporate_bucket/Development/*
```

Uso incorreto de curingas

Você não pode usar um curinga na parte do ARN que especifica o tipo de recurso, como o termo `user` em um ARN do IAM. Por exemplo, não é permitido fazer as ações abaixo.

```
arn:aws:iam::123456789012:u*   <== not allowed
```

Identificadores do IAM

O IAM usa alguns identificadores diferentes para os usuários, grupos de usuários, funções, políticas e certificados de servidor. Esta seção descreve os identificadores e quando usar cada um deles.

Tópicos

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.

Ao usar o AWS CloudFormation para criar recursos, você pode especificar um caminho para usuários, grupos de usuários e funções, além de 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 qualquer limite entre usuários ou grupos de usuários 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 ter 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
```

Muitos dos seguintes exemplos incluem caminhos na parte de recurso do ARN. Caminhos não podem ser criados ou manipulados no AWS Management Console. 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/JohnDoe
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
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/JohnDoe
arn:aws:sts::123456789012:assumed-role/Accounting-Role/JaneDoe
arn:aws:sts::123456789012:self
arn:aws:iam::123456789012:mfa/JaneDoeMFA
arn:aws:iam::123456789012:u2f/user/JohnDoe/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
```

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/JohnDoe
```

- Outro usuário com um caminho que reflete um gráfico da organização:

```
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/JaneDoe
```

- 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
```

- Um usuário federado identificado 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
```

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.

```
{
    "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 tempo de execução (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, grupos de usuários ou políticas. 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, grupos de usuários ou políticas na 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/JohnDoe
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/JohnDoe
arn:aws:iam::111122223333:user/division_abc/subdivision_xyz/JaneDoe
```

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*
```

Example Exemplo de uso de caminhos e ARNs para um grupo de usuários baseado em projeto

Caminhos não podem ser criados ou manipulados no AWS Management Console. 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.

```
{
  "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	Token de portador de serviço do AWS STS
ACCA	Credencial específica de contexto
AGPA	Grupo de usuários
AIDA	IAM user (Usuário do IAM)
AIPA	Perfil de instância do Amazon EC2
AKIA	Chave de acesso
ANPA	Política gerenciada

Prefixo	Tipo de recurso
ANVA	Versão em uma política gerenciada
APKA	Chave pública
AROA	Função
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 Como usar um ID externo ao conceder acesso aos seus recursos da AWS a 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), vá até 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 fazer login naAWS no Guia do usuário do AWS Sign-In para entrar no AWS Management Console.
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 a área de aumento de cotas pode ser aprovada automaticamente.

Cotas ajustáveis para recursos do IAM

Recurso	Cota padrão	Cota máxima
Políticas gerenciadas pelo cliente por conta	1500	5000
Grupos por conta	300	500
Perfis de instância por conta	1000	5000
Políticas gerenciadas por perfil	10	20
Políticas gerenciadas por usuário	10	20
Duração da política de confiança da função	2048 caracteres	4096 caracteres
Funções por conta	1000	5000
Certificados de servidor por conta	20	1000

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.

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. **Note** O IAM; não conta espaços em branco ao calcular o tamanho de uma política em relação a esses limites.
Para políticas gerenciadas	• O tamanho de cada política gerenciada não pode exceder 6,144 caracteres.

Descrição	Limite
	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
Path	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 AWS Management Console, `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 especific ar um valor para o parâmetro `DurationS econds` , 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 visualiza r o valor máximo para sua função, consulte Visualizar a configuração de duração máxima da sessão para uma função.
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. • Você pode passar apenas um documento de política JSON ao criar uma sessão temporária de forma programática para uma função ou para um usuário federado. • Além disso, uma conversão da AWS compacta as políticas de sessão e as tags de sessão transmitidas em um formato binário compactado que tem uma cota separada. O elemento de resposta `PackedPolicySize` indica, em porcentagem, o quão perto as políticas e tags da sua solicitação estão do limite de tamanho superior. • Recomendamos que você passe políticas de sessão usando a AWS CLI ou API da AWS. O AWS Management Console pode adicionar informações adicionais da sessão do console à política compactada.

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. • Uma conversão da AWS compacta as políticas de sessão e as tags de sessão passadas em um formato binário compactad o que têm um limite separado. Você pode passar tags de sessão usando o AWS CLI ou API da AWS. O elemento de resposta `PackedPolicySize` indica, em porcentag em, o quão perto as políticas e tags da sua solicitação estão do limite de tamanho superior.
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 do usuário	64 caracteres

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 Security Token Service (AWS STS). Você pode usar essa conexão para permitir que 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 AWS STS, você define um VPC endpoint de interface para o AWS STS. O endpoint fornece uma conectividade confiável e escalável 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 VPC endpoints de interface são desenvolvidos pelo AWS PrivateLink, uma tecnologia da AWS permite 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 produtos 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.

Disponibilidade

No momento, o AWS STS oferece suporte a VPC endpoints nas seguintes regiões:

- Leste dos EUA (Ohio)
- Leste dos EUA (N. da Virgínia)
- Oeste dos EUA (N. da Califórnia)
- Oeste dos EUA (Oregon)
- África (Cidade do Cabo)
- Ásia-Pacífico (Hong Kong)
- Ásia-Pacífico (Mumbai)
- Ásia-Pacífico (Osaka)
- Ásia-Pacífico (Seul)
- Ásia-Pacífico (Singapura)
- Ásia-Pacífico (Sydney)
- Ásia-Pacífico (Tóquio)
- Canadá (Central)
- China (Pequim)
- China (Ningxia)
- Europa (Frankfurt)
- Europa (Irlanda)
- Europa (Londres)

- Europa (Milão)
- Europa (Paris)
- Europa (Estocolmo)
- Oriente Médio (Bahrein)
- South America (São Paulo)
- AWS GovCloud (Leste dos EUA)
- AWS GovCloud (Oeste dos EUA)

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 `setEndpoint` e `setRegion` 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.

Serviços da AWS 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 recurso, 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 recurso, 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 recursos 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 ter 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 oferecer suporte às 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 O que é ABAC para a AWS?. Para visualizar um tutorial com etapas para configurar o ABAC, consulte Use attribute-based access control (ABAC) (Usar controle de acesso baseado em atributos [ABAC]).
- Credenciais temporárias: é possível usar credenciais de curto prazo obtidas ao fazer login usando o IAM Identity Center ou alternar perfis no console, ou ainda credenciais que você gera usando o AWS STS na AWS CLI ou a API da AWS. Você pode 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 ter mais informações, consulte Credenciais de segurança temporárias no IAM.
- Funções vinculadas ao serviço: uma função vinculada ao serviço é um tipo especial de função 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 funções de serviço padrão. Para ter mais informações, consulte Usar funções vinculadas ao serviço.

- Mais informações – Se um serviço não é totalmente compatível com um recurso, 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	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Account Management	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
AWS Activate Console	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
AWS Amplify Admin	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
AWS Amplify	Yes (Sim)	Yes (Sim)	No (Não)	Parci	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Amplify UI Builder	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
APIs do Apache Kafka para clusters do Amazon MSK	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Amazon API Gateway	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Sim
Amazon API Gateway Management	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon API Gateway Management V2	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS App2Container	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS AppConfig	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS AppFabric	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon AppFlow	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon AppIntegrations	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Application Auto Scaling	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Application Cost Profiler	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Application Discovery Arsenal	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Application Discovery Service	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 Sim
AWS Application Migration Service	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Application Transformation Service	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS App Mesh	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Demonstração do AWS App Mesh	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
AWS App Runner	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon AppStream 2.0	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS AppSync	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Artifact	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Athena	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Audit Manager	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Auto Scaling	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Intercâmbio de dados B2B AWS	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
AWS Backup	Yes (Sim)	Yes (Sim)	Yes (Sim)	Yes (Sim)	Yes (Sim)	Sim
AWS Backup Gateway	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Armazenamento do AWS Backup	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
AWS Batch	Yes (Sim)	Parci	No (Não)	Yes (Sim)	Yes (Sim)	Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Bedrock	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Billing and Cost Management	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 Sim
Exportações de dados AWS Billing and Cost Management	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Billing Conductor	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Braket	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Budget Service	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 No (Não)
AWS BugBust	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Certificate Manager (ACM)	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Chatbot	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
Amazon Chime	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Clean Rooms	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
ML AWS Clean Rooms	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Client VPN	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
AWS Cloud9	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Sim
API de controle do Nuvem AWS	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Cloud Directory	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
AWS CloudFormation	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon CloudFront	Yes (Sim)	Yes (Sim)	No (Não)	Parci	Sim	Parcial (Informações)
KeyValueStore do Amazon CloudFront	Sim	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
AWS CloudHSM	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Cloud Map	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon CloudSearch	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS CloudShell	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS CloudTrail	 Yes (Sim)	 Sim	 Parcial (Informações)	 Parcial (Informações)	 Sim	 Sim
Dados do AWS CloudTrail	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon CloudWatch	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Sim	Parcial (Informações)
Amazon CloudWatch Application Insights	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
Amazon CloudWatch Application Signals	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon CloudWatch Evidently	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon CloudWatch Internet Monitor	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon CloudWatch Logs	Yes (Sim)	Yes (Sim)	Yes (Sim)	Parci	Yes (Sim)	Sim
Amazon CloudWatch Network Monitor	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon CloudWatch Observability Access Manager	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon CloudWatch RUM	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon CloudWatch Synthetics	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS CodeArtifact	 Yes (Sim)	 Yes (Sim)	 Sim	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS CodeBuild	 Yes (Sim)	 Sim	 Sim (Informações)	 Parcial (Informações)	 Sim	 No (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)	 No (Não)
AWS CodeConnections	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS CodeDeploy	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Serviço de comandos de host seguro do AWS CodeDeploy	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon CodeGuru Profiler	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon CodeGuru Reviewer	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon CodeGuru Security	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS CodePipeline	Yes (Sim)	Parci	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
AWS CodeStar	Yes (Sim)	Parci	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Conexões do AWS CodeStar	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim
Notificações do AWS CodeStar	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim
Amazon CodeWhisperer	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Cognito	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Cognito Sync	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
Grupos de usuários do Amazon Cognito	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Comprehend	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Comprehend Medical	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Compute Optimizer	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 Sim
AWS Config	 Sim	 Parcial (Informações)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Connect	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Connect Cases	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Connect Customer Profiles	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Comunicações de saída de alto volume do Amazon Connect	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Connect Voice ID	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Console Mobile Application	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Faturamento consolidado da AWS	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Catálogo de controle da AWS	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Control Tower	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS Cost and Usage Report	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 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	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Serviço de verificação de clientes da AWS	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Database Migration Service	Yes (Sim)	Sim	Não (Informações)	Sim	Yes (Sim)	Sim
Database Query Metadata Service	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
AWS Data Exchange	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon Data Lifecycle Manager	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
AWS Data Pipeline	Yes (Sim)	Yes (Sim)	No (Não)	Parci	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS DataSync	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon DataZone	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Deadline Cloud	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS DeepComposer	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS DeepRacer	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Detective	Yes (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	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	Sim
Ferramentas de diagnóstico AWS	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
AWS Direct Connect	Yes (Sim)	Yes (Sim)	No (Não)	Sim	Yes (Sim)	Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Directory Service	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon DocumentDB Elastic Clusters	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim
Amazon DynamoDB Accelerator (DAX)	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	Sim
Amazon DynamoDB	Yes (Sim)	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	No (Não)
Amazon Elastic Compute Cloud (Amazon EC2)	Sim	Parci	No (Não)	Sim	Sim	Parcial (Informações)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
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)	Não	No (Não)	Yes (Sim)	Sim
Amazon ElastiCache	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim
AWS Elastic Beanstalk	Yes (Sim)	Parci	No (Não)	Sim	Yes (Sim)	Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Elastic Block Store (Amazon EBS)	Yes (Sim)	Parci	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon Elastic Container Registry (Amazon ECR)	Yes (Sim)	Yes (Sim)	Yes (Sim)	Yes (Sim)	Yes (Sim)	Sim
Amazon Elastic Container Registry Público (Amazon ECR Público)	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon Elastic Container Service (Amazon ECS)	Sim	Parcial (Informações)	No (Não)	Yes (Sim)	Yes (Sim)	Sim
AWS Elastic Disaster Recovery	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Elastic File System (Amazon EFS)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Parci	 Yes (Sim)	 Sim
Amazon Elastic Inference	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Elastic Kubernetes Service (Amazon EKS)	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Elastic Kubernetes Service (Amazon EKS) Auth	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS Elastic Load Balancing	 Sim  Parci	 No (Não)  Parci	 Yes (Sim)	 Sim		

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Elastic Transcoder	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS Elemental Appliances e serviço de ativação de software	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Dispositivos e software do AWS Elemental	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Elemental MediaConnect	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
AWS Elemental MediaConvert	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Sim	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
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)	 Sim	 Parcial (Informações)
AWS Elemental MediaPackageV2	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
VOD do AWS Elemental MediaPackage	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Sim	 Parcial (Informações)
AWS Elemental MediaStore	 Sim	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Elemental MediaTailor	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Casos de suporte do Elemental	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Conteúdo de suporte do Elemental	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon EMR	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon EMR no EKS	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon EMR Serverless	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Entity Resolution	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon EventBridge	 Yes (Sim)	 Yes (Sim)	 Sim	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon EventBridge Pipes	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Agendador do Amazon EventBridge	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon EventBridge Schemas	 Yes (Sim)	 Yes (Sim)	 Sim	 Yes (Sim)	 Yes (Sim)	 No (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
API do Amazon FinSpace	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS Firewall Manager	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Parcial

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Fleet Hub for AWS IoT Device Management	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Forecast	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Fraud Detector	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
FreeRTOS	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Nível gratuito da AWS	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon FSx	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim
Amazon GameLift	Yes (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)	Yes (Sim)	Yes (Sim)	Parci	Yes (Sim)	No (Não)
AWS Glue DataBrew	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Ground Station	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Ground Truth Labeling	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon GuardDuty	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
APIs e notificações do AWS Health	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS HealthImaging	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS HealthLake	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
AWS HealthOmics	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon Honeycode	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
AWS IAM Identity Center	Yes (Sim)	Yes (Sim)	No (Não)	Parci	Yes (Sim)	Sim
IAM Identity Center Directory	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
IAM Identity Center Identity Store	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Serviço OIDC do IAM Identity Center	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS Identity and Access Management (IAM)	 Yes (Sim)	 Sim	 Parcial (Informações)	 Parcial (Informações)	 Parcial (Informações)	 No (Não)
AWS Identity and Access Management Access Analyzer	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Parcial
AWS Identity and Access Management Roles Anywhere	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Identity Store Auth	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Identity Sync	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS Import/Export	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Inspector	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Inspector Classic	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon InspectorScan	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Interactive Video Service	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Interactive Video Service Chat	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Faturamento da AWS	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS IoT 1-Click	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS IoT Analytics	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
AWS IoT	Sim	Sim	Parcial (Informações)	Sim	Yes (Sim)	No (Não)
AWS IoT Core Device Advisor	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
AWS IoT Device Tester	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
AWS IoT Events	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
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)	No (Não)
AWS IoT GreengrassV2	Yes (Sim)	Yes (Sim)	No (Não)	Parci	Yes (Sim)	No (Não)
AWS IoT Jobs DataPlane	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
AWS IoT RoboRunner	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
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)	 No (Não)
AWS IQ	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
Permissões do AWS IQ	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Kendra	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (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)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Keyspaces (for Apache Cassandra)	 Yes (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)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Managed Service for Apache Flink V2	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Data Firehose	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Kinesis Data Streams	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Kinesis Video Streams	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Lake Formation	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Lambda	 Yes (Sim)	 Yes (Sim)	 Sim	 Parcial (Informações)	 Sim	 Parcial (Informações)
AWS Launch Wizard	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Lex	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Lex V2	 Yes (Sim)	 Yes (Sim)	 Sim	 Yes (Sim)	 Yes (Sim)	 Sim
AWS License Manager	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS License Manager Linux Subscriptions Manager	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS License Manager Assinaturas de usuário	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 Sim
Amazon Lightsail	 Sim	 Parcial (Informações)	 No (Não)	 Parcial (Informações)	 Sim	 Sim
Amazon Location Service	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Lookout for Equipment	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Lookout for Metrics	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon Lookout for Vision	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon Machine Learning	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
Amazon Macie	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim
AWS Mainframe Modernization	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Managed Blockchain	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Consulta ao Amazon Managed Blockchain	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Managed Grafana	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Managed Service for Prometheus	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Managed Streaming for Apache Kafka (MSK)	 Yes (Sim)	 Sim	 Parcial (Informações)	 Sim	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Managed Streaming for Kafka Connect	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
Amazon Managed Workflows for Apache Airflow	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Marketplace	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 Sim
Catálogo do AWS Marketplace	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Marketplace Commerce Analytics	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 No (Não)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Serviço de implantação doAWS Marketplace	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Descoberta do AWS Marketplace	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Marketplace Entitlement Service	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Marketplace Image Building Service	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Portal de gerenciamento do AWS Marketplace	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Marketplace Metering Service	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
AWS Marketplace Private Marketplace	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
AWS Marketplace Procurement Systems Integration	Sim	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
Relatórios do vendedor do AWS Marketplace	Sim	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
AWS Marketplace Vendor Insights	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Mechanical Turk	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon MediaImport	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 No (Não)	 No (Não)
Amazon MemoryDB for Redis	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Serviço de entrega de mensagens da Amazon	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Serviço Amazon Message Gateway	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Microservice Extractor for .NET	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWSCréditos do Programa de Aceleração da Migração	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS Migration Hub	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
AWS Migration Hub Orchestrator	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Migration Hub Refactor Spaces	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Migration Hub Strategy Recommendations	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 Sim
Amazon Monitron	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon MQ	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Neptune	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
Análise do Amazon Neptune	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Network Firewall	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Network Manager	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Sim	 Sim (Informações)
Chat AWS Network Manager	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Nimble Studio	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon One Enterprise	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon OpenSearch Ingestion	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon OpenSearch Serverless	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon OpenSearch Service	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS OpsWorks	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Gerenciamento de configuração AWS OpsWorks	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Organizations	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Sim
AWS Outposts	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Panorama	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Gerenciamento de contas da AWS Partner Central	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Payment Cryptography	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Payments	Sim	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
AWS Performance Insights	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
Amazon Personalize	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
Amazon Pinpoint	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Serviço de e-mail do Amazon Pinpoint	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
SMS e serviço de voz do Amazon Pinpoint	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Pinpoint SMS and Voice Service V2	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Polly	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS Price List	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS 5G privado	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Private CA Connector for Active Directory	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Private Certificate Authority (AWS Private CA)	 Yes (Sim)	 Yes (Sim)	 Sim	 Yes (Sim)	 Yes (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)	 No (Não)
Amazon Q Business	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Aplicações do Amazon Q Business Q	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Q Developer	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 Sim
Amazon Q in Connect	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Quantum Ledger Database (Amazon QLDB)	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon QuickSight	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
API Data do Amazon RDS	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Autenticação do IAM do Amazon RDS	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWSLixeira	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Redshift	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
API de dados do Amazon Redshift	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Redshift Serverless	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Rekognition	 Yes (Sim)	 Sim	 Parcial (Informações)	 Sim	 Yes (Sim)	 No (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)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Resource Access Manager (AWS RAM)	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Explorador de recursos da AWS	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Resource Groups	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Sim	 Parcial (Informações)	 No (Não)
AWS Resource Groups Tagging API	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon RHEL Knowledgebase Portal	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS RoboMaker	Yes (Sim)	Yes (Sim)	No (Não)	Sim	Yes (Sim)	Sim
Amazon Route 53	Yes (Sim)	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
Amazon Route 53 Application Recovery Controller - Zonal Shift	Sim	Yes (Sim)	Não	No (Não)	Yes (Sim)	No (Não)
Domínios do Amazon Route 53	Yes (Sim)	Não	No (Não)	No (Não)	No (Não)	No (Não)
Perfis do Amazon Route 53	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Route 53 Recovery Cluster	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Route 53 Recovery Control Config	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Route 53 Recovery Readiness	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Route 53 Resolver	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon S3 Express	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon S3 Glacier	Yes (Sim)	Yes (Sim)	Yes (Sim)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon SageMaker	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Sim	Parcial (Informações)
Recursos geoespaciais do Amazon SageMaker	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon SageMaker Ground Truth Synthetic	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
AWS Savings Plans	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Secrets Manager	 Yes (Sim)	 Yes (Sim)	 Sim	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Security Hub	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon Security Lake	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
AWS Security Token Service (AWS STS)	 Sim	 Parcial (Informações)	 No (Não)	 Sim	 Parcial (Informações)	 No (Não)
AWS Serverless Application Repository	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Service Catalog	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Service Quotas	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Shield	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Signer	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Acessar a AWS	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon SimpleDB	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (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	 Sim	 Parcial (Informações)	 Sim
Amazon Simple Notification Service (Amazon SNS)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Simple Queue Service (Amazon SQS)	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 Parci	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Simple Storage Service (Amazon S3)	 Yes (Sim)	 Yes (Sim)	 Sim	 Parcial (Informações)	 Sim	 Parcial (Informações)
Objeto Lambda do Amazon Simple Storage Service (Amazon S3)	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Simple Storage Service (Amazon S3) no AWS Outposts	 Yes (Sim)	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Sim
Amazon Simple Workflow Service (Amazon SWF)	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS SimSpace Weaver	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Site-to-Site VPN	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 Sim
AWS Snowball	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Snowball Edge	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Snow 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)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
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)
Cadeia de Suprimentos AWS	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
AWS Support App in Slack	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	No (Não)
AWS Support	Yes (Sim)	Não	No (Não)	No (Não)	Yes (Sim)	Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Support Planos	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Recomendações do AWS Support	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Sustainability	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Systems Manager	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS Systems Manager para SAP	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Systems Manager GUI Connect	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Systems Manager Incident Manager	 Yes (Sim)	 Yes (Sim)	 Sim	 Yes (Sim)	 Yes (Sim)	 Sim
Contatos do AWS Systems Manager Incident Manager	 Yes (Sim)	 Yes (Sim)	 Sim	 No (Não)	 Yes (Sim)	 No (Não)
Tag Editor	 Sim	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
AWS Tax Settings	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
AWS Telco Network Builder	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Textract	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Timestream	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon Timestream Influxdb	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
API do AWS Tiros (para o Reachability Analyzer)	 Sim	 Não	 No (Não)	 No (Não)	 No (Não)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon Transcribe	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
AWS Transfer Family	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
Amazon Translate	Yes (Sim)	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	No (Não)
AWS Trusted Advisor	Parcial (Informações)	Sim	Não	No (Não)	Parci	Sim
Notificações ao usuário da AWS	Sim	Yes (Sim)	No (Não)	Yes (Sim)	Yes (Sim)	Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Contatos de notificações de usuários da AWS	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS Assinaturas de usuário	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Acesso Verificado pela AWS	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Verified Permissions	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
Amazon Virtual Private Cloud (Amazon VPC)	 Sim	 Parcial (Informações)	 Parcial (Informações)	 Sim	 Sim	 Parcial (Informações)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon VPC Lattice	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Serviços do Amazon VPC Lattice	 Sim	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)
AWS WAF	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS WAF Classic	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
AWS WAF Regional	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
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)	 No (Não)
Amazon WorkDocs	 Yes (Sim)	 Não	 No (Não)	 No (Não)	 Yes (Sim)	 No (Não)
Amazon WorkMail	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Amazon WorkMail Message Flow	 Yes (Sim)	 Yes (Sim)	 Não	 No (Não)	 Yes (Sim)	 No (Não)

Serviço	Ações	Permissões em nível de recurso	Políticas baseadas em atributos	ABAC	Credenciais temporárias	Funções vinculadas ao serviço
Amazon WorkSpaces	 Yes (Sim)	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
Amazon WorkSpaces Secure Browser	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 Sim
Thin Client Amazon WorkSpaces	 Sim	 Yes (Sim)	 No (Não)	 Yes (Sim)	 Yes (Sim)	 No (Não)
AWS X-Ray	 Sim	 Parcial (Informações)	 No (Não)	 Parcial (Informações)	 Sim	 No (Não)

Mais informações

Amazon CloudFront

O CloudFront não tem perfis vinculados a serviço, mas o Lambda@Edge tem. Para obter mais informações, consulte Funções vinculadas ao serviço para o Lambda@Edge no Guia do desenvolvedor do Amazon CloudFront.

AWS CloudTrail

O CloudTrail oferece suporte a políticas baseadas em recursos somente em canais do CloudTrail usados para integrações do CloudTrail Lake com origens de eventos fora da AWS.

O CloudTrail oferece suporte ao controle de acesso baseado em tags para armazenamentos de dados e canais de eventos do CloudTrail Lake. O CloudTrail não oferece suporte a controles de acesso baseados em tags para trilhas.

Amazon CloudWatch

Os perfis vinculados ao serviço do CloudWatch não podem ser criados usando o AWS Management Console 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

Você pode 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, frotas do Amazon EC2 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 ter mais informações, consulte Concessão de permissões a um usuário para alternar funções.

O IAM é compatível com controle de acesso baseado em etiquetas para a maioria dos recursos do IAM. Para ter mais informações, consulte Recursos de etiquetas do IAM.

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. Você pode 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 ações de API que usam uma função do Lambda como o recurso obrigatório. Camadas, mapeamentos de origem de eventos e recursos de configuração de assinatura de código não são aceitos.

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

O Lightsail 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.

AWS Network Manager

O AWS 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 Relational Database Service

O Amazon Aurora é um mecanismo de banco de dados relacional gerenciado compatível com o MySQL e o PostgreSQL. Você pode 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.

AWS Resource Groups

Os usuários podem assumir um perfil com uma política que permita operações de grupos de recurso.

Amazon SageMaker

Os perfis vinculados a serviço estão atualmente disponíveis para trabalhos de treinamento do SageMaker Studio e do SageMaker.

AWS Security Token Service

O AWS STS 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 etiqueta apenas para recursos de objeto.

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 AWS Support e é controlado por políticas do IAM do AWS Support.

Amazon Virtual Private Cloud

Em uma política de usuário do IAM, não é possível restringir permissões para um endpoint da Amazon VPC específico. Qualquer elemento `Action` que inclua as ações de API `ec2:*VpcEndpoint*` ou `ec2:DescribePrefixLists` devem especificar `""Resource": "*""`. Para obter mais informações, consulte Gerenciamento de identidade e acesso para VPC endpoints e serviços de VPC endpoint no Guia do AWS PrivateLink.

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.

A Amazon VPC não tem perfis vinculados ao serviço, mas o AWS Transit Gateway sim. Para obter mais informações, consulte Use funções vinculadas ao serviço para o transit gateway no Guia do AWS Transit Gateway para Amazon VPC.

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.

Assinar solicitações de API do AWS

> **Important**
>
> Caso use um AWS SDK (consulte Código de exemplo e Bibliotecas) ou a ferramenta AWS Command Line Interface (CLI) para enviar solicitações de API para a AWS, você pode pular esta seção, pois os clientes de SDK e CLI autenticam 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.

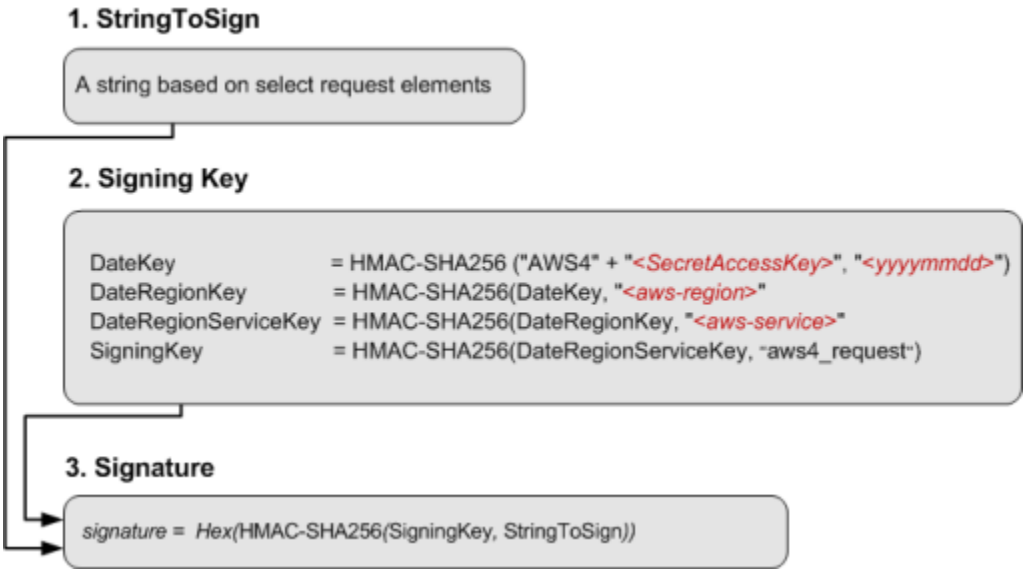
As informações de autenticação enviadas em uma solicitação devem incluir uma assinatura. Para calcular uma assinatura, primeiro concatene os elementos de solicitação selecionados para formar uma string, chamada de string a ser assinada. Em seguida, use uma chave de assinatura para calcular o código de autenticação de mensagens por hash (HMAC) da string para assinar.

No AWS Signature Version 4, você não usa sua chave de acesso secreta apenas para assinar a solicitação. Em vez disso, use primeiro sua chave de acesso secreta para derivar uma chave de assinatura. A chave de assinatura derivada é específica para data, serviço e região. Para obter mais

informações sobre como criar uma chave de assinatura em diferentes linguagens de programação, consulte Exemplos de assinatura de solicitação.

A versão 4 do Signature é o protocolo de assinatura da AWS. A AWS também oferece suporte a uma extensão, a Signature Version 4A, que oferece suporte a assinaturas para solicitações de API multirregionais. Para obter mais informações, consulte o projeto sigv4a-signing-examples no GitHub.

O diagrama a seguir ilustra o processo geral de computação de uma assinatura.



- 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 variável 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.
- Para chave de assinatura, o diagrama mostra uma série de cálculos, em que o resultado de cada etapa é inserido na etapa seguinte. A etapa final é a chave de assinatura.
- 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.

Conteúdo

- Crie uma solicitação assinada de API da AWS
- Exemplos de assinatura de solicitação
- Solucionar problemas de solicitações assinadas de APIs da AWS

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.

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.

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: use `AWS4-HMAC-SHA256` para especificar o Signature Version 4 com o algoritmo de hash `HMAC-SHA256`.
- Credencial: uma string que consiste no ID da chave de acesso, na data no formato AAAAMMDD, no código da região, no código do serviço e na string de término `aws4_request`, 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
  ```

- Cabeçalhos assinados: os cabeçalhos HTTP a serem incluídos na assinatura, separados por ponto e vírgula (;). Por exemplo, `host;x-amz-date`.

- Assinatura: uma string codificada em hexadecimal que representa a assinatura calculada. Você deve calcular a assinatura com o algoritmo especificado no parâmetro `Algorithm`.

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. Para obter mais informações sobre o valor do cabeçalho de autorização, como calcular a assinatura e opções relacionadas, consulte a página Authenticating Requests: Using the Authorization Header (AWS Signature Version 4) na Referência de APIs do Amazon S3.

Veja a seguir um exemplo de um valor de cabeçalho `Authorization`. 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.

```
Authorization: AWS4-HMAC-SHA256
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
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. É necessário informar esse valor ao usar o AWS Signature Version 4 para autentica ção. A string especifica o AWS Signature Version 4 (`AWS4`) e o algoritmo de assinatura (`HMAC-SHA256`).
Credential	Seu ID de chave de acesso e as informações do escopo, como 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` Em que: o valor da `<date>` é especificado usando o formato AAAAMMDD. O valor de `<aws-service>` é s3 ao enviar uma solicitaç ão ao Amazon S3.
SignedHeaders	Uma lista separada por ponto e vírgula dos cabeçalhos de solicitação usados para computar a `Signature` . A lista contém somente nomes de cabeçalho, e os nomes dos cabeçalhos devem estar em letras minúsculas. Por exemplo: host;range;x-amz-date
Assinatura	A assinatura de 256 bits expressa como 64 caracteres hexadecimais minúsculo s. Por exemplo:`fe5f80f77d5fa3beca 038a248ff027d0445342fe2855d dc963176630326f1024`

Componente	Descrição
	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.

O exemplo a seguir é um URL pré-assinado. As quebras de linha foram adicionadas a este exemplo somente para facilitar a leitura:

```
https://s3.amazonaws.com/examplebucket/test.txt ?
X-Amz-Algorithm=AWS4-HMAC-SHA256 &
X-Amz-Credential=<your-access-key-id>/20130721/us-east-1/s3/aws4_request &
X-Amz-Date=20130721T201207Z &
X-Amz-Expires=86400 &
X-Amz-SignedHeaders=host &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.

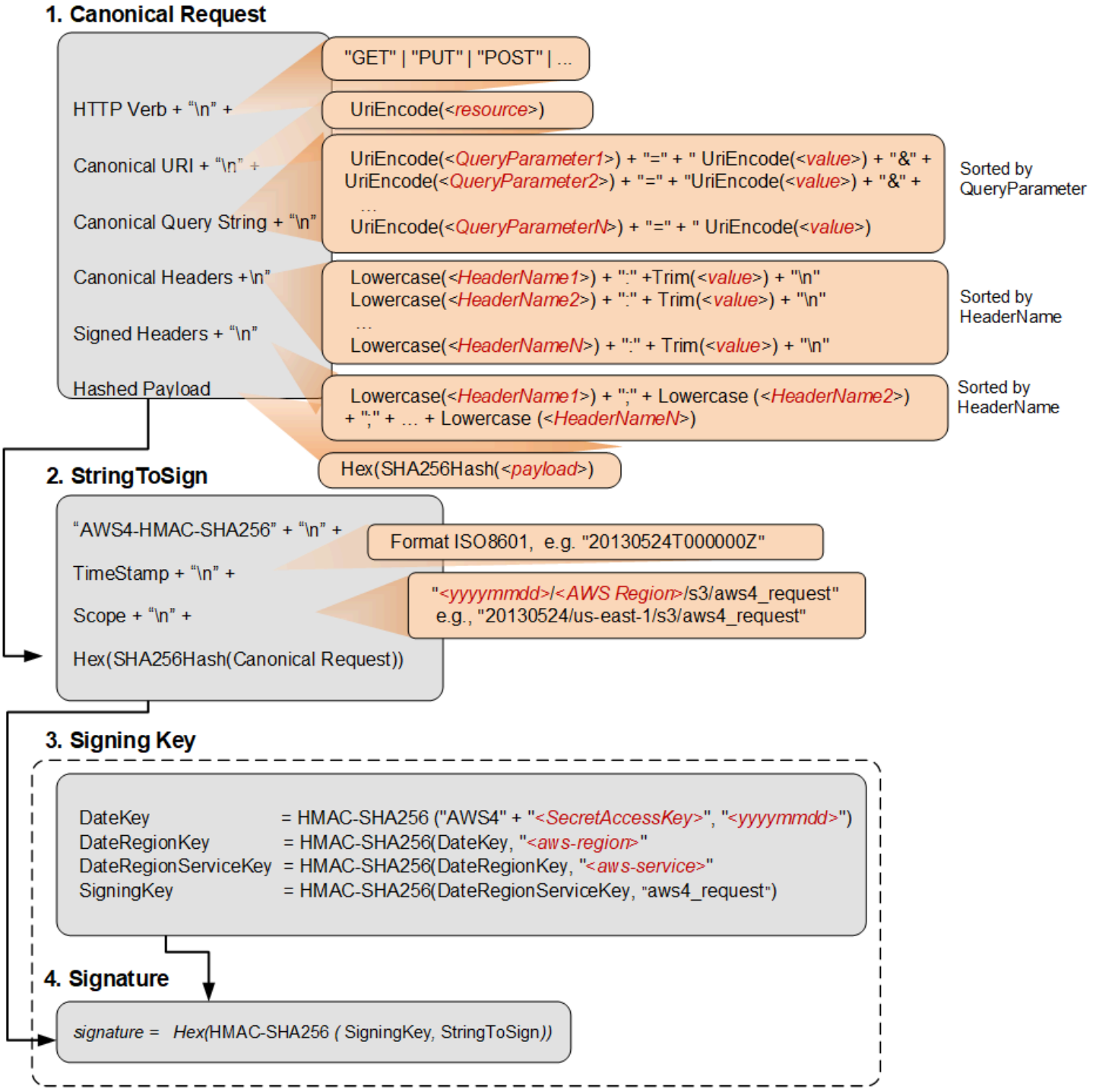
Nome de parâmetro de string de consulta	Descrição
X-Amz-Algorithm	Identifica a versão do AWS Signature e o algoritmo usado para calcular a assinatura. No AWS Signature Version 4, defina esse valor de parâmetro como `AWS4-HMAC-SHA256`. Essa string identifica o AWS Signature Verion 4 (AWS4) e o algoritmo HMAC-SHA256 (`HMAC-SHA256`).
X-Amz-Credential	Além do ID da chave de acesso, esse parâmetro também fornece o escopo (região e serviço da AWS) para os quais a assinatura é válida. O valor deve corresponder ao escopo usado nos cálculos de assinatura, abordados na seção a seguir. A forma geral para esse valor de parâmetro é: `<your-access-key-id>/<date>/ <AWS Region>/<AWS-service>/aws4_ request` Por exemplo: `AKIAIOSFODNN7EXAMP LE/20130721/us-east-1/s3/aw s4_request` Para obter uma lista de strings regionais da AWS, consulte Regional Endpoints na Referência geral da AWS.
X-Amz-Date	O formato de data e hora deve seguir o padrão ISO 8601 e deve ter a formatação `yyyyMMddT HHmmssZ`. 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 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.
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 AWS SDK (consulte Código de exemplo e Bibliotecas) ou a ferramenta AWS Command Line Interface (CLI) para enviar solicitações de API para a AWS, você pode pular esta seção, pois os clientes de SDK e CLI autenticam 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.

Veja abaixo uma visão geral do processo para criar uma solicitação assinada. Para calcular uma assinatura, primeiro é necessário usar uma string para assinar. Em seguida, calcule um hash HMAC-SHA256 da string a ser assinada usando uma chave de assinatura. O diagrama a seguir ilustra o processo, incluindo os vários componentes da string criada para assinatura.



A tabela a seguir descreve as funções exibidas no diagrama. 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. Essa é a assinatura final.
`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 “+”). • 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.

Função	Descrição
	Important As funções UriEncode padrão fornecidas por 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 garantir que a codificação funcione. Para ver um exemplo de uma função UriEncode em Java, consulte Java Utilities no site do GitHub.

Note

Ao assinar suas solicitações, você pode usar o AWS Signature Version 4 ou o AWS Signature Version 4A. A principal diferença entre os dois está na forma como a assinatura é calculada. Com o AWS Signature Version 4A, a assinatura não inclui informações específicas da região e é calculada usando o algoritmo `AWS4-ECDSA-P256-SHA256`.

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 `X-Amz-Security-Token` ao cabeçalho de autorização ou à 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 detalhes.

Resumo das etapas de assinatura

Etapa 1: 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 uma string para assinar. Para obter detalhes, consulte Elementos de uma assinatura de solicitação de API da AWS.

Etapa 2: criar um hash para a solicitação canônica

Derive uma chave de assinatura executando uma série de operações de hash com chave (operações HMAC) na data, na região e no serviço da solicitação com sua chave de acesso secreta da AWS como a chave para a operação de hash inicial.

Etapa 3: criar uma string para assinar

Crie uma string para assinar com a solicitação canônica e informações adicionais, como o algoritmo, data de solicitação, gama de credenciais e o resumo (hash) da solicitação canônica.

Etapa 4: calcular a assinatura

Depois de derivar a chave de assinatura, calcule esta executando uma operação de hash de chave na string a ser assinada. Use a chave de assinatura derivada como a chave de hash para esta operação.

Etapa 5: adicionar a assinatura à solicitação

Depois de calcular a assinatura, adicione-a a um cabeçalho de HTTP ou à string de consulta da solicitação.

Etapa 1: criar uma solicitação canônica

Crie uma solicitação canônica concatenando as strings a seguir, separadas por caracteres de linha nova. Isso ajuda a garantir que a assinatura que você calcular e a assinatura que a AWS calcular sejam correspondentes.

```
<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 por URI do URI do componente de caminho absoluto, começando pelo "/" após o nome do domínio e seguindo até o final da string ou até o caractere de ponto de interrogação ("?"), se houver parâmetros de string de consulta. Se o caminho absoluto estiver vazio, use um caractere de barra inclinada (/). O URI no seguinte exemplo, `/examplebucket/myphoto.jpg`, é o caminho absoluto, e você não codifica o "/" no caminho absoluto:

  ```
  http://s3.amazonaws.com/examplebucket/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/examplebucket?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 `examplebucket`:

  ```
  http://s3.amazonaws.com/examplebucket?acl
  ```

  A CanonicalQueryString nesse caso é a seguinte:

  ```
  UriEncode("acl") + "=" + ""
  ```

Se o URI não contém 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 “\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 nova linha (“\n”). Veja a seguir um exemplo de 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*.

> **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, convém incluir todos os cabeçalhos no cálculo da assinatura.

- *`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`. Por exemplo, 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 da seguinte forma:

```
Hex(SHA256Hash(""))
```

O hash retorna o seguinte valor:

```
e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
```

Por exemplo, ao carregar um objeto usando uma solicitação PUT, forneça dados do objeto no corpo. Ao recuperar um objeto usando um solicitação GET, calcule o hash da string vazia.

Etapa 2: 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.

Etapa 3: criar uma string para assinar

Crie uma string concatenando 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. Para SHA-256, o algoritmo é AWS4-HMAC-SHA256.
- *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 da credencial. Isso restringe a assinatura resultante à região e ao serviço especificados. A string tem o seguinte formato: *AAAAMMDD*/*região*/*serviço*/aws4_request.
- *HashedCanonicalRequest*: o hash da solicitação canônica. Esse valor é calculado na Etapa 2.

Veja a seguir um exemplo de string para assinar.

```
"AWS4-HMAC-SHA256" + "\n" +
```

```
timeStampISO8601Format + "\n" +
<Scope> + "\n" +
Hex(SHA256Hash(<CanonicalRequest>))
```

Etapa 4: calcular a assinatura

No AWS Signature Version 4, em vez de usar suas chaves de acesso da AWS para assinar uma solicitação, crie uma chave de assinatura que tenha como escopo uma região e um serviço específicos como as informações de autenticação que você adicionará à solicitação.

```
DateKey = HMAC-SHA256("AWS4"+"<SecretAccessKey>", "<YYYYMMDD>")
DateRegionKey = HMAC-SHA256(<DateKey>, "<aws-region>")
DateRegionServiceKey = HMAC-SHA256(<DateRegionKey>, "<aws-service>")
SigningKey = HMAC-SHA256(<DateRegionServiceKey>, "aws4_request")
```

Para obter uma lista de strings regionais, consulte Regional Endpoints na Referência geral da AWS.

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.

Entrada obrigatória

- Uma string, `Key`, que contém sua chave de acesso secreta
- Uma string, `Date`, que contém a data usada no escopo da credencial, no formato AAAAMMDD
- Uma string, `Region`, que contém o código da região (por exemplo, `us-east-1`)
- Uma string, `Service`, que contém o código do serviço (por exemplo, `ec2`)
- A string para assinar que você criou na etapa anterior.

Para calcular a assinatura

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.

   ```
   kDate = 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.

```
kRegion = 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.

```
kService = hash(kRegion, Service)
```

4. Chame a função de hash com o resultado da chamada anterior como a chave e “aws4_request” como os dados.

```
kSigning = hash(kService, "aws4_request")
```

5. Chame a função de hash com o resultado da chamada anterior como a chave e a string para assinar como os dados. O resultado é a assinatura como valor binário.

```
signature = hash(kSigning, string-to-sign)
```

6. Converta a assinatura da representação binária em hexadecimal, em caracteres minúsculos.

Etapa 5: adicionar a assinatura à solicitação

Example Exemplo: cabeçalho de autorização

O exemplo a seguir mostra um cabeçalho `Authorization` para a ação `DescribeInstances`. 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
```

Example Exemplo: solicitação com parâmetros de autenticação na string de consulta

O exemplo a seguir mostra uma consulta para a ação `DescribeInstances` 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
```

Código-fonte nos AWS SDKs

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 for .NET: AWS4Signer.cs
- AWS SDK for C++: AWSAuthV4Signer.cpp
- AWS SDK for Go: v4.go
- AWS SDK for Java: BaseAws4Signer.java
- AWS SDK for JavaScript: v4.js
- AWS SDK for PHP: SignatureV4.php
- AWS SDK for Python (Boto): signers.py
- AWS SDK for Ruby: signer.rb

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

Usar o Signature Version 4 com o Amazon Translate mostra como usar um programa Python para adicionar informações de autenticação às solicitações do Amazon Translate. O exemplo faz uma solicitação POST, cria uma estrutura JSON que contém o texto a ser traduzido no corpo (carga) da solicitação e repassa as informações de autenticação em um cabeçalho de autorização.

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 S3 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 S3 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 Sigv4a 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 solicitações assinadas de APIs 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 Solução de problemas de mensagens de erro de acesso negado.

Erros de solicitação canônica e string de assinatura

Se você calculou incorretamente a solicitação canônica em Etapa 2: criar um hash para a solicitação canônica ou Etapa 3: 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

Certifique-se de que o cabeçalho de autorização SigV4 que você adicionou a Etapa 4: calcular a assinatura inclua a chave de credencial correta, semelhante à seguinte:

```
Authorization: AWS4-HMAC-SHA256
```

```
Credential=AKIAIOSFODNN7EXAMPLE/20130524/us-east-1/s3/aws4_request,
SignedHeaders=host;range;x-amz-date,
Signature=example-generated-signature
```

Se a chave de credencial estiver ausente ou incorreta, poderá ser recebido o erro: `Authorization header requires 'Credential' parameter. Authorization header requires 'Signature' parameter.` Certifique-se de que a solicitação de autorização SigV4 também inclua a data da solicitação usando `HTTP Date` ou o cabeçalho `x-amz-date`.

Erros de escopo de credenciais

O escopo da credencial criado em Etapa 3: 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 ver 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 elementos 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.
- Políticas gerenciadas pela AWS 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.
- Chaves de contexto de condição globais da AWS: 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 elementos 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 Serviços da AWS 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 políticas 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

> **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.

```
{
  "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.

```
{
  "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.

```
{
  "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:::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
        "arn:aws:s3:::BUCKET-NAME/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.

```
{
  "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.

Elementos da política JSON da AWS: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

- Especificar um principal
- Entidades principais da Conta da AWS
- Entidades de segurança de função do IAM
- Entidades principais da sessão de função
- Entidades principais de usuário do IAM

- Entidades principais do Centro de identidade do IAM
- Entidades principais de sessão de usuário federado do AWS STS
- Responsáveis pelos serviços da AWS
- Entidades principais de serviço da AWS nas regiões de aceitação
- Todas as entidades principais
- Mais informações

Especificar um 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
- IAM users
- Sessões de usuário federado
- Serviços da AWS
- 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.

Entidades principais da Conta da AWS

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.

As funções do IAM são identidades que existem no IAM. Funções confiam em outra identidade autenticada, como uma entidade principal na AWS ou um usuário de um provedor de identidade externo. Quando uma entidade principal ou uma identidade assume uma função, elas recebem credenciais de segurança temporárias. Em seguida, elas podem usar essas credenciais como uma entidade principal de sessão de função para executar operações na AWS.

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 função contiver um ARN que aponte para uma função específica do IAM, esse ARN será transformado no ID exclusivo da entidade principal da função 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 final 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.

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.

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 ter mais informações, consulte Entidades de segurança de função do IAM.

Entidades principais da sessão de função assumida

Uma entidade principal da sessão de função assumida é uma entidade principal de sessão que resulta do uso da operação `AssumeRole` do AWS STS. Para mais informações sobre quais entidades principais podem assumir uma função usando essa operação, consulte Comparação das operações de API do AWS STS.

Para especificar o ARN da sessão de função assumida no elemento `Principal`, use o seguinte formato:

```
"Principal": { "AWS": "arn:aws:sts::AWS-account-ID:assumed-role/role-name/role-session-name" }
```

Ao especificar uma sessão de função assumida em um elemento `Principal`, não é possível usar um curinga “*” para indicar todas as sessões. Os principais devem sempre nomear uma sessão específica.

Entidades principais de sessão do OIDC

Uma entidade principal de sessão do OIDC é uma entidade principal de sessão que resulta do uso da operação `AssumeRoleWithWebIdentity` do AWS STS. Você pode usar um provedor OIDC

externo (IdP) para se conectar e, em seguida, assumir um perfil do IAM usando essa operação. Isso tira proveito da federação de identidades e emite uma sessão de função. Para mais informações sobre quais entidades principais podem assumir uma função usando essa operação, consulte Comparação das operações de API do AWS STS.

Ao emitir uma função de um provedor OIDC, você obtém esse tipo especial de entidade principal de sessão que inclui informações sobre o provedor OIDC.

Use esse tipo de entidade principal em sua política para permitir ou negar acesso com base no provedor confiável de identidade da Web. Para especificar o ARN da sessão do perfil de OIDC no elemento `Principal` de uma política de confiança de perfil, use o seguinte formato:

```
"Principal": { "Federated": "cognito-identity.amazonaws.com" }
```

```
"Principal": { "Federated": "www.amazon.com" }
```

```
"Principal": { "Federated": "graph.facebook.com" }
```

```
"Principal": { "Federated": "accounts.google.com" }
```

Entidades principais de sessão SAML

Uma entidade principal da sessão SAML é uma entidade principal de sessão que resulta do uso da operação AWS STS `AssumeRoleWithSAML`. Você pode usar um provedor de identidade (IdP) SAML externo para se conectar e, em seguida, assumir uma função do IAM usando essa operação. Isso tira proveito da federação de identidades e emite uma sessão de função. Para mais informações sobre quais entidades principais podem assumir uma função usando essa operação, consulte Comparação das operações de API do AWS STS.

Ao emitir uma função de um provedor de identidade SAML, você obtém esse tipo especial de entidade principal de sessão que inclui informações sobre o provedor de identidade SAML.

Use esse tipo de entidade principal em sua política para permitir ou negar acesso com base no provedor confiável de identidade SAML. 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.

Entidades principais de sessão de usuário federado do AWS STS

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 usar as sessões de usuários federados do AWS STS somente quando necessário, como quando é necessário ter acesso de usuário raiz. Em vez disso, sugerimos usar funções para delegar permissões.

Uma entidade principal de sessão de usuário federado do AWS STS é uma entidade principal de sessão que resulta do uso da operação AWS STS `GetFederationToken`. Nesse caso, o AWS STS usa a federação de identidades como o método para obter tokens temporários de acesso em vez de usar funções do IAM.

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 Comparação das operações de API do AWS STS.

- Usuário federado do IAM: um usuário do IAM se federa usando a operação `GetFederationToken`, que resulta em uma entidade principal de sessão de usuário federado para esse usuário do IAM.
- Usuário raiz federado: um usuário raiz se federa usando a operação `GetFederationToken`, que resulta em uma entidade principal de 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" }
```

Responsáveis pelos serviços da AWS

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": "*"`.

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 Serviços da AWS que funcionam com o IAM, verificando se o serviço tem Yes (Sim) na coluna Service-linked role (Função vinculada ao serviço) e abrindo o link Yes (Sim) para visualizar a documentação da função vinculada a esse serviço específico. 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"
   ]
}
```

Entidades principais de serviço da AWS 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 atributos 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 Especificar `NotPrincipal` com `Deny` para negar explicitamente todas as entidades principais exceto as especificadas no elemento `Condition`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "UsePrincipalArnInsteadOfNotPrincipalWithDeny",
      "Effect": "Deny",
      "Action": "s3:*",
      "Principal": "*",
      "Resource": [
        "arn:aws:s3:::BUCKETNAME/*",
        "arn:aws:s3:::BUCKETNAME"
      ],
      "Condition": {
        "ArnNotEquals": {
          "aws:PrincipalArn": "arn:aws:iam::444455556666:user/user-name"
        }
      }
    }
  ]
}
```

Mais informações

Para obter mais informações, 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

Você pode usar o elemento do `NotPrincipal` para negar o acesso a todas as entidades principais exceto o usuário do IAM, usuário federado, perfil do IAM, Conta da AWS, serviço da AWS ou outra entidade principal especificado no elemento do `NotPrincipal`.

Você pode usá-lo 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.

`NotPrincipal` deve ser usado com `"Effect":"Deny"`. O uso de `"Effect":"Allow"` não é compatível.

> **Important**
>
> Muitos poucos cenários exigem o uso de `NotPrincipal`. Recomendamos que você explore outras opções de autorização antes de decidir usar `NotPrincipal`. 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. Para obter mais informações, consulte Todas as entidades principais.

Especificar **`NotPrincipal`** com **`Deny`**

Quando você usar `NotPrincipal` com `Deny`, você também deve especificar o ARN da conta 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.

> **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 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.

> **Note**
>
> Como prática recomendada, você deve incluir os ARNs para a conta em sua política. Alguns serviços exigem o ARN da conta, embora isso não seja necessário em todos os casos. Todas as políticas existentes sem o ARN necessário continuarão funcionando, mas as novas políticas que incluírem esses serviços deverão atender a esse requisito. O IAM não rastreia esses serviços e, portanto, recomenda que você sempre inclua o ARN da conta.

Os exemplos a seguir mostram como usar `NotPrincipal` e `"Effect": "Deny"` na mesma declaração de política com eficácia.

Example Exemplo de usuário do IAM na mesma conta ou em outra conta

No exemplo a seguir, todas as entidades principais, exceto o usuário de nome Bob na Conta da AWS 444455556666, têm o acesso explicitamente negado a um recurso. Observe que, como prática recomendada, o elemento `NotPrincipal` contém o ARN do usuário Bob e da Conta da AWS à qual Bob pertence (`arn:aws:iam::444455556666:root`). Se o elemento `NotPrincipal` contivesse apenas o ARN de Bob, o efeito da política poderia ser negar explicitamente o acesso à Conta da AWS que contém esse usuário. Em alguns casos, um usuário não pode ter mais permissões do que sua conta pai, portanto, se a conta de Bob tem o acesso explicitamente negado, Bob também pode ser incapaz de acessar o recurso.

Este exemplo funciona conforme o esperado quando faz parte de uma instrução de política em uma política baseada em recurso que está anexada a um recurso na mesma conta ou em outra Conta da AWS (diferente de 444455556666). Este exemplo por si só não concede acesso a Bob, ele apenas omite Bob da lista de principais que são explicitamente negados. Para conceder a Bob acesso ao recurso, outra declaração de política deve explicitamente permitir o acesso usando `"Effect": "Allow"`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "NotPrincipal": {"AWS": [
            "arn:aws:iam::444455556666:user/Bob",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::BUCKETNAME",
            "arn:aws:s3:::BUCKETNAME/*"
        ]
    }]
}
```

Example Exemplo de função do IAM na mesma conta ou em outra conta

No exemplo a seguir, todas as entidades principais, exceto o usuário de perfil assumido denominado cross-account-audit-app na Conta da AWS 444455556666, têm o acesso explicitamente negado a um recurso. Como prática recomendada, o elemento `NotPrincipal` contém o ARN do usuário de perfil assumido (cross-account-audit-app), o perfil (cross-account-read-only-role) e a Conta da

AWS à qual o perfil pertence (444455556666). Se o elemento `NotPrincipal` não continha o ARN da função, o efeito da política poderia negar explicitamente o acesso à função. Da mesma forma, se o elemento `NotPrincipal` não contivesse o ARN da Conta da AWS à qual o perfil pertence, o efeito da política poderia ser negar explicitamente o acesso à Conta da AWS e todas as entidades nessa conta. Em alguns casos, os usuários de perfil assumido não podem ter mais permissões do que seu perfil pai, e perfis não podem ter mais permissões do que sua Conta da AWS pai, de forma que quando o perfil ou a conta tem o acesso explicitamente negado, o usuário de perfil assumido poderia ser incapaz de acessar o recurso.

Este exemplo funciona conforme o esperado quando faz parte de uma instrução de política em uma política baseada em recurso anexada a um recurso em outra Conta da AWS (exceto 444455556666). Este exemplo por si só não concede acesso ao usuário de função assumida cross-account-audit-app, ele apenas omite o cross-account-audit-app da lista de principais que são explicitamente negados. Para conceder a cross-account-audit-app acesso ao recurso, outra declaração de política deve explicitamente permitir o acesso usando `"Effect": "Allow"`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "NotPrincipal": {"AWS": [
            "arn:aws:sts::444455556666:assumed-role/cross-account-read-only-role/cross-
account-audit-app",
            "arn:aws:iam::444455556666:role/cross-account-read-only-role",
            "arn:aws:iam::444455556666:root"
        ]},
        "Action": "s3:*",
        "Resource": [
            "arn:aws:s3:::Bucket_AccountAudit",
            "arn:aws:s3:::Bucket_AccountAudit/*"
        ]
    }]
}
```

Ao especificar uma sessão de função assumida em um elemento `NotPrincipal`, não é possível usar um curinga (*) para significar "todas as sessões". Os principais devem sempre nomear uma sessão específica.

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.

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 EC2

```
"Action": "ec2:StartInstances"
```

Ação do IAM

```
"Action": "iam:ChangePassword"
```

Ação do Amazon S3

```
"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 um curinga (*) 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 (*) 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. Isso não permite que os

usuários usem a operação da API `ListAllMyBuckets` do S3, pois essa ação requer o recurso"*". 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.

```
{
```

```
    "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` especifica o objeto ou objetos que a instrução abrange. As instruções devem incluir um elemento `Resource` ou um elemento `NotResource`. Você especifica um recurso usando um ARN. Para obter mais informações sobre o formato de ARNs, consulte ARNs do IAM.

Cada serviço tem seu próprio conjunto de recursos. Embora você sempre use um nome de recurso da Amazon (ARN) para especificar um recurso, os detalhes do ARN para um recurso dependem do serviço e do recurso. Para obter informações sobre como especificar um recurso, consulte a documentação do serviço para o qual deseja escrever uma instrução.

> **Note**
>
> Alguns serviços não permitem que você especifique ações para recursos individuais; em vez disso, qualquer ação que você listar no elemento `Action` ou `NotAction` se aplica a todos os recursos naquele serviço. Nesses casos, você deve usar o 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 curingas como parte do ARN do recurso. Você pode usar caracteres curinga (* e ?) dentro de segmentos do ARN (as partes separadas por dois pontos) para representar qualquer combinação de caracteres por um asterisco (*) e qualquer caractere único por um ponto de interrogação (?). 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 Nomes de recurso da Amazon (ARN)

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:::DOC-EXAMPLE-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:::DOC-EXAMPLE-BUCKET/*/test/*"
```

Os caracteres curinga no ARN se aplicam a todos os objetos a seguir no bucket, não apenas ao primeiro objeto listado.

```
DOC-EXAMPLE-BUCKET/1/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test/3/object.jpg
DOC-EXAMPLE-BUCKET/1/2/3/test/4/object.jpg
DOC-EXAMPLE-BUCKET/1///test///object.jpg
DOC-EXAMPLE-BUCKET/1/test/.jpg
DOC-EXAMPLE-BUCKET//test/object.jpg
DOC-EXAMPLE-BUCKET/1/test/
```

Considere os dois últimos objetos na lista anterior. Um nome de objeto do Amazon S3 pode validamente 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:

```
DOC-EXAMPLE-BUCKET/1-test/object.jpg
DOC-EXAMPLE-BUCKET/test/object.jpg
DOC-EXAMPLE-BUCKET/1/2/test.jpg
```

Especificação de vários recursos

Você pode especificar múltiplos recursos. 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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "dynamodb:*",
    "Resource": "arn:aws:dynamodb:us-east-2:account-id: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.

```
{
  "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 em `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 a ação também incluísse ações do Amazon EC2, a política não negaria acesso a nenhum recurso do EC2. 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 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 `johndoe` sejam aceitas. Os usuários chamados `JohnDoe` não têm permissão de acesso.

```
"Condition" : { "StringEquals" : { "aws:username" : "johndoe" }}
```

A seguinte condição usa o operador `StringEqualsIgnoreCase` para corresponder usuários chamados `johndoe` ou `JohnDoe`.

```
"Condition" : { "StringEqualsIgnoreCase" : { "aws:username" : "johndoe" }}
```

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.
- Os serviços da AWS 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 Chaves de contexto de múltiplos valores.
- Para obter uma lista de todas as chaves de contexto disponíveis globalmente, consulte Chaves de contexto de condição globais da AWS.
- 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. As informações são usadas para avaliar e autorizar a solicitação. 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.

Quando uma solicitação é enviada, a AWS avalia cada chave de contexto na política e retorna um valor true, false, not present e, ocasionalmente, null (uma string de dados vazia). Uma chave de contexto que não está na solicitação é considerada uma não correspondência. Por exemplo,

a política a seguir permite remover seu próprio dispositivo de autenticação multifator (MFA), mas somente se você fez login usando MFA na última hora (3.600 segundos).

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "AllowRemoveMfaOnlyIfRecentMfa",
        "Effect": "Allow",
        "Action": [
            "iam:DeactivateMFADevice"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}",
        "Condition": {
            "NumericLessThanEquals": {"aws:MultiFactorAuthAge": "3600"}
        }
    }
}
```

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.
- Null: para chaves de contexto definidas pelo usuário, como passar etiquetas em uma solicitação, é possível incluir uma string vazia. Nesse caso, o valor no contexto da solicitação é null.
  Um valor null pode retornar true em alguns casos. Por exemplo, se você usar o operador de condição `ForAllValues` de múltiplos valores com a chave de contexto `aws:TagKeys`, poderá ter resultados inesperados se o contexto da solicitação retornar como null. Para obter mais informações, consulte aws:TagKeys e Chaves de contexto de múltiplos valores.

O bloco de condição

O exemplo a seguir mostra o formato básico de um elemento `Condition`:

```
"Condition": {"StringLike": {"s3:prefix": ["janedoe/*"]}}
```

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 `janedoe/*`. 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`.

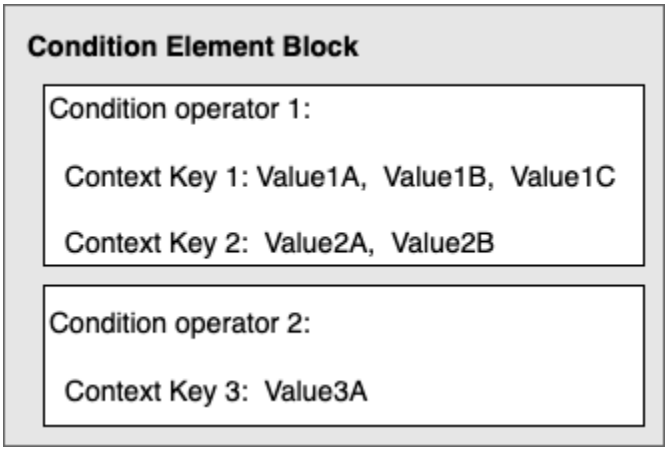
```
"Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}
```

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 ter mais informações, consulte Chaves de contexto de múltiplos 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 ter mais informações, consulte Chaves de contexto de múltiplos 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 Chaves de contexto de condição globais da AWS. 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
- Numeric
- Data e hora
- Booleano
- Binário
- IP address
- 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.

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 ter mais informações, consulte Chaves de contexto de múltiplos valores.
`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.

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`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id: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.

Você pode usar uma variável de política com o 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.

```
{
  "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:::BUCKET-NAME",
      "Condition": {"StringLike": {"s3:prefix": [
        "",
        "home/",
        "home/${aws:username}/"
      ]}}
    },
    {
      "Effect": "Allow",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}",
        "arn:aws:s3:::BUCKET-NAME/home/${aws:username}/*"
      ]
    }
  ]
}
```

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.

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. Quando você usa os operadores StringLike ou StringNotLike para correspondências parciais de string de um ARN ou data, a correspondência ignora qual parte da estrutura está marcada como curinga.

Por exemplo, as condições a seguir pesquisam uma correspondência parcial de um ARN usando operadores de condição diferentes.

Quando o ArnLike é usado, as partes partição, serviço, ID da conta, tipo de recurso e ID parcial do recurso do ARN devem ter a correspondência exata com o ARN no contexto da solicitação. Somente a região e o caminho do recurso permitem a correspondência parcial.

```
"Condition": {"ArnLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/
*"}}
```

Quando StringLike é usado em vez de ArnLike, a correspondência ignora a estrutura do ARN e permite a correspondência parcial, independentemente da parte que foi marcada como curinga.

```
"Condition": {"StringLike": {"aws:SourceArn": "arn:aws:cloudtrail:*:111122223333:trail/
*"}}
```

ARN	ArnLike	StringLike
arn:aws:cloudtrail:us-west-2:111122223333:trail/finance	Match	Match
arn:aws:cloudtrail:us-east-2:111122223333:trail/finance/archive	Match	Match
arn:aws:cloudtrail:us-east-2:444455556666:user/111122223333:trail/finance	Nenhuma correspondência	Match

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.

Operador de condição	Descrição
`NumericEquals`	Correspondência
`NumericNotEquals`	Correspondência negativa
`NumericLessThan`	Correspondência "menor que"
`NumericLessThanEquals`	Correspondência "menor ou igual a"

Operador de condição	Descrição
`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 `example_bucket` por vez.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::example_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.

Você não pode usar uma variável de política com o operador de condição `Numeric`.

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).

> **Note**
>
> Curingas não são permitidos para operadores de condição de data.

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 declaração 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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id: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.

Você não pode usar uma variável de política com o operador de condição `Date`.

Operadores de condição booliana

Operadores de condição boolianos permitem que você construa elementos `Condition` que restringem o acesso com base na comparação de uma chave a um "verdadeiro" ou "falso".

Operador de condição	Descrição
`Bool`	Correspondência booliana

Por exemplo, essa política baseada em identidade usará o operador de condição `Bool` com a chave `aws:SecureTransport` para negar a replicação de objetos e tags de objetos para o 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BooleanExample",
      "Action": "s3:ReplicateObject",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
      ],
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "false"
```

```
        }
      }
    }
  ]
}
```

Se a chave especificada em uma condição de política não estiver presente no contexto de solicitação, os valores não corresponderão. O contexto da solicitação `aws:SecureTransport` retorna true ou false.

Você pode usar uma variável de política com o operador de condição `Boolean`.

Operadores de condição binários

O operador de condição `BinaryEquals` permite que você construa elementos `Condition` que testam valores de chave em formato binário. Ele compara o valor da chave especificada byte por byte à uma representação base-64 codificada do valor binário na política.

```
"Condition" : {
  "BinaryEquals": {
    "key" : "QmluYXJ5VmFsdWVJbkJhc2U2NA=="
  }
}
```

Se a chave especificada em uma condição de política não estiver presente no contexto de solicitação, os valores não corresponderão.

Você não pode usar uma variável de política com o operador de condição `Binary`.

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.

Operador de condição	Descrição
`IpAddress`	O endereço IP ou intervalo especificado
`NotIpAddress`	Todos os endereços IP, exceto o endereço IP ou intervalo especific ado

Por exemplo, a instrução a seguir usa o operador de condição `IpAddress` com a chave `aws:SourceIp` para especificar que a solicitação deve partir de um intervalo de IP 203.0.113.0 a 203.0.113.255.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "iam:*AccessKey*",
    "Resource": "arn:aws:iam::account-id: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.

Você não pode usar uma variável de política com o operador de condição `IpAddress`.

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.

```
{
```

```
  "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.

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.

Você pode usar uma variável de política com o 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`.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"AWS": "123456789012"},
    "Action": "SQS:SendMessage",
    "Resource": "arn:aws:sqs:REGION:123456789012:QUEUE-ID",
    "Condition": {"ArnEquals": {"aws:SourceArn":
 "arn:aws:sns:REGION:123456789012:TOPIC-ID"}}
  }
}
```

Se a chave especificada em uma condição de política não estiver presente no contexto de solicitação, os valores não corresponderão. 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.

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 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." 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 tag estiver faltando.

Exemplo do uso de **`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:

```
{
  "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.

```
{
    "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": "*"
        }]
}
```

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, você pode usar esse operador de condição para determinar se um usuário está usando suas próprias credenciais para a operação ou credenciais temporárias. 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 não deve usar credenciais temporárias (a chave não deve existir) para que o usuário use a API do Amazon EC2.

```
{
  "Version": "2012-10-17",
  "Statement":{
      "Action":"ec2:*",
      "Effect":"Allow",
      "Resource":"*",
      "Condition":{"Null":{"aws:TokenIssueTime":"true"}}
  }
}
```

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 AWS Management Console, 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

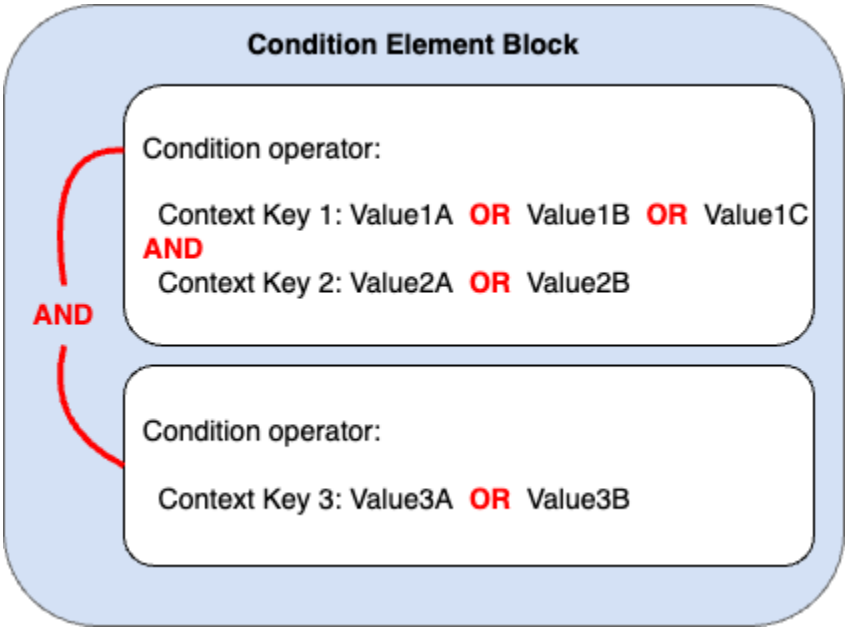
- Lógica de avaliação para múltiplas chaves de contexto ou valores
- Lógica de avaliação para os operadores de conjuntos de condições com correspondência negada

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

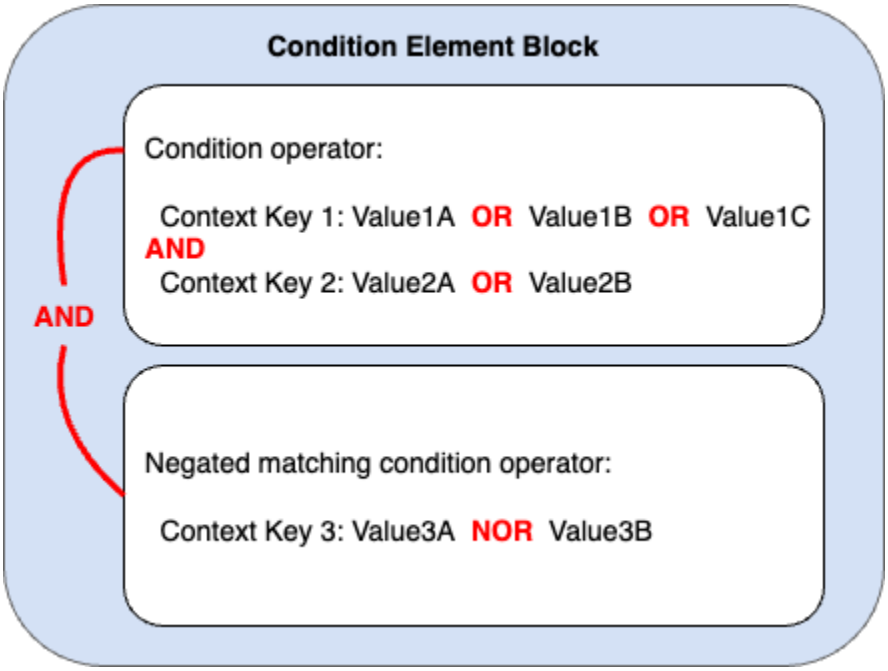
Deny desejado, todas as chaves de contexto de um bloco de condições devem ser resolvidas como verdadeiro. O usuário que faz a solicitação deve ter ambas as chaves de tag da entidade principal, departamento e função, que incluem um dos valores de chave de tag especificados na política. Além disso, o ARN 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-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"
          ]
        }
      }
    }
  ]
}
```

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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ExamplePolicy",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::222222222222:root"
      },
      "Action": "s3:ListBucket",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-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"
          ]
        }
      }
    }
  ]
}
```

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, 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, você pode marcar recursos do na AWS. As tags de recurso são armazenadas como pares de chave-valor de tag. Uma chave de tag de recurso pode ter um único

valor de tag. Portanto, 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, você pode marcar recursos na AWS e incluir vários pares de chave-valor de tag em uma 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.

> **Important**
>
> 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. Para saber mais sobre operadores de conjunto de condições, consulte Chaves de contexto de múltiplos valores.

As classificações Valor único e Valores múltiplos constam na descrição de cada chave de contexto de condição como Tipo de valor no tópico Chaves de contexto de condição globais da AWS. A Referência de autorização de serviço usa uma classificação de tipo de valor diferente para chaves de contexto de múltiplos valores no seguinte formato: um prefixo `ArrayOf` seguido pelo tipo de categoria do operador de condição. Por exemplo, o `ArrayOfString` ou o `ArrayOfARN`.

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, aws:PrincipalServiceNamesList é uma chave de contexto de múltiplos valores.

É possível usar qualquer chave de contexto de valor único disponível como variável de política. Você não pode usar uma chave de contexto de múltiplos valores como variável de política. Para obter mais informações sobre variáveis de política, consulte Elementos de política do IAM: variáveis e etiquetas.

As chaves de contexto de múltiplos valores exigem operadores de conjunto de condições `ForAllValues` ou `ForAnyValue`. Chaves de contexto que incluem pares de chave-valor, como the section called “RequestTag” e the section called “ResourceTag”, podem causar confusão porque podem haver múltiplos valores de *`tag-key`*. Porém, como cada *`tag-key`* pode ter apenas um valor, `aws:RequestTag` e `aws:ResourceTag` são chaves de contexto de valor único. O uso de

operadores de conjunto de condições com chaves de contexto de valor único pode levar a políticas excessivamente permissivas.

Chaves de contexto de múltiplos 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 para que você possa testar 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`. Vários valores de chave de condição devem ser colocados entre colchetes, como uma matriz. Por exemplo, `"Key2": ["Value2A", "Value2B"]`.

- `ForAllValues`: esse qualificador testa se o valor de cada membro do conjunto de solicitações é um subconjunto do conjunto de chaves de contexto de condição. A condição retornará verdadeiro se cada valor de chave de contexto na solicitação corresponder a pelo menos um valor de chave de contexto na política. Também retornará verdadeiro se não houver chaves de contexto na 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 verdadeiros, você pode incluir o operador de condição `Null` em sua política com um valor falso para verificar se a chave de contexto existe e se seu valor não é nulo.

  > **Important**
  >
  > Tenha cuidado ao usar `ForAllValues` com um efeito `Allow` porque ele pode ser excessivamente permissivo se chaves de contexto ausentes ou chaves de contexto com valores vazios no contexto da solicitação forem inesperadas. Você pode incluir o operador de condição `Null` na política com um valor falso 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.

- `ForAnyValue`: esse qualificador testa se pelo menos um membro do conjunto de valores da chave de contexto da solicitação corresponde a pelo menos um membro do conjunto de valores da

chave de contexto de sua condição de política. A chave de contexto retornará como verdadeiro se qualquer um dos valores na solicitação corresponder a algum dos valores na política. A condição retornará como falsa se nenhuma chave de contexto corresponder ou se houver um conjunto de dados nulo.

> **Note**
>
> 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.

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. (Visualizar este exemplo.)
- Um bloco de condição com múltiplos valores e chaves de contexto de valor único. (Visualizar este exemplo.)

Exemplos de políticas de condição: chaves de contexto de múltiplos valores

- Política de negação com operador de conjunto de condições `ForAllValues`. (Visualizar este exemplo.)
- Política de negação com operador de conjunto de condições `ForAnyValue`. (Visualizar este exemplo.)

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

O exemplo de política baseada em identidade a seguir nega o uso de ações de marcação do IAM quando há prefixos de chave de etiqueta específicos na solicitação. Todo valor para a chave de contexto `aws:TagKeys` inclui um curinga (*) para correspondência parcial de strings. 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` retornar o valor verdadeiro, cada valor da solicitação deverá corresponder a pelo menos um valor na política.

O operador de conjunto `ForAllValues` também retornará como verdadeiro se não houver chaves de contexto na 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 verdadeiro, inclua 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.

> **Important**
>
> Esta política não permite qualquer ação. Use essa política em combinação com outras políticas que permitam ações específicas.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyRestrictedTags",
      "Effect": "Deny",
      "Action": [
        "iam:Tag*",
        "iam:UnTag*"
      ],
      "Resource": [
        "*"
      ],
```

```
      "Condition": {
        "Null": {
          "aws:TagKeys": "false"
        },
        "ForAllValues:StringLike": {
          "aws:TagKeys": [
            "key1*",
            "key2*",
            "key3*"
          ]
        }
      }
    }
  ]
}
```

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.

```
{
    "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": ["environment", "webserver"]
                }
            }
        }
    ]
}
```

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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CreateVolume",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:::volume/*",
      "Condition": {
        "StringLike": {
          "aws:RequestTag/project": "*"
        }
```

```
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:CreateTags",
      "Resource": "arn:aws:ec2:region:account:*/*",
      "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 `pre-prod` 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.

```
{
  "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"
          ]
        }
      }
    }
  ]
}
```

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

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 `DOC-EXAMPLE-BUCKET` para projetos de `marketing`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
      "Condition": {"StringLike": {"s3:prefix": ["marketing/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-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 AWSChaves de contexto de condição globais da .

⚠ 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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:ListBucket"],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET"],
      "Condition": {"StringLike": {"s3:prefix": ["${aws:PrincipalTag/team}/*"]}}
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/${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:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3::DOC-EXAMPLE-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.

```
{
  "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.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowOnlyDeptS3Prefix",
    "Effect": "Allow",
    "Action": ["s3:GetObject"],
    "Resource": ["arn:aws:s3:::DOC-EXAMPLE-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 Recursos de etiquetas do IAM. 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 AWS Management Console 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

Você pode usar variáveis de política no elemento `Resource` e em comparações de string no elemento `Condition`.

Elemento de recurso

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:::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.

```
{
  "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.

```
{
  "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 Chaves de contexto de condição globais da AWS.

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.

```
{
  "Version": "2012-10-17",
  "Statement": [
   {
    "Effect": "Deny",
    "Action": "s3:GetObject",
    "Resource": "arn:aws:s3:::/example-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`
- Usuário federado
  - `aws:username`: (não está presente)
  - `aws:userid`: *`account`*:*`caller-specified-name`*
  - `aws:PrincipalType`: `FederatedUser`
- Usuário federado da Web e usuário federado SAML

> **Note**
>
> Para obter informações sobre as chaves de política que estão disponíveis ao usar a federação OIDC, consulte Identificar usuários com a 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 usuários federados

Usuários federados são usuários que são autenticados usando um sistema que não seja o 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 mais informações, 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 `DOC-EXAMPLE-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 `DOC-EXAMPLE-BUCKET-company-wide`, onde podem visualizar informações gerais, como instruções para ingressar em uma equipe.

```
"Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET-${aws:PrincipalTag/team, 'company-wide'}"
```

Para obter mais informações

Para obter mais informações sobre políticas, consulte o seguinte:

- Políticas e permissões no IAM
- Exemplos de políticas baseadas em identidade do IAM
- Referência de elementos 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)
- Booliano
- Nulo
- 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.

Type (Tipo)	JSON
String	String
Inteiro	Número
Float	Número
Booliano	verdadeiro falso
Nulo	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 AWS Management Console, 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 de segurança 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: a AWS processa as informações coletadas na solicitação para determinar quais políticas se aplicam à solicitação.
3. Avaliação de políticas em uma única conta: a AWS avalia todos os tipos de política, que afetam a ordem em que as políticas são avaliadas.

4. Determinar se uma solicitação é permitida ou negada em uma conta: a AWS então processa as políticas em relação ao contexto da solicitação para determinar se a solicitação é permitida ou negada.

Processamento do contexto da solicitação

A AWS processa a solicitação para reunir as seguintes informações em um contexto de solicitação:

- Ações (ou operações): as ações ou as operações que a entidade de segurança deseja executar.
- Recursos: o objeto de recurso da AWS no qual as ações ou operações são executadas.
- Entidade de segurança: o usuário, função, usuário federado ou aplicativo que enviou a solicitação. As informações sobre a entidade principal incluem as políticas que estão associadas à entidade principal.
- Dados do ambiente – Informações sobre o endereço IP, o agente de usuário, o status do SSL habilitado ou a hora do dia.
- 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.

A AWS usa essas informações para localizar as políticas que se aplicam ao contexto da solicitação.

Avaliação de políticas em uma única conta

A forma como o AWS avalia as políticas depende dos tipos de políticas que se aplicam ao contexto da solicitação. Os seguintes tipos de políticas, listados em ordem de frequência, estão disponíveis para uso em uma única Conta da AWS. Para obter mais informações sobre esses tipos de política, consulte Políticas e permissões no IAM. Para saber como a AWS avalia políticas para acesso entre contas, consulte Lógica de avaliação de política entre contas.

1. 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`.
2. Políticas baseadas em recurso: as políticas baseadas em recurso concedem permissões à entidade principal (conta, usuário, função e entidades principais de sessão, como sessões de função e usuários federados do IAM) especificada como entidade principal. As permissões

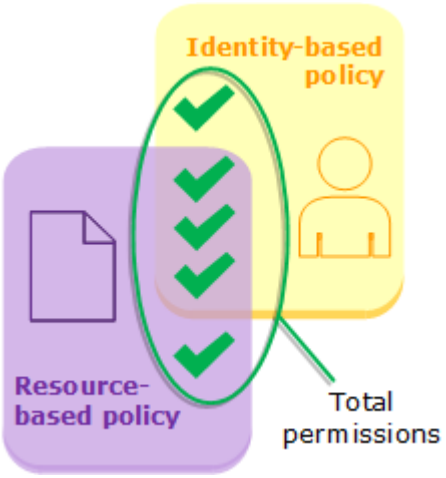
definem o que a entidade principal pode fazer com o recurso ao qual a política está anexada. Se as políticas baseadas em recurso e as políticas baseadas em identidade se aplicarem a uma solicitação, o AWS verificará todas as políticas para pelo menos um `Allow`. Quando políticas baseadas em recursos são avaliadas, o ARN da entidade principal especificado na política determina se as negações implícitas em outros tipos de política são aplicáveis à decisão final.

3. Limites de permissões do IAM: limites de permissões são um recurso avançado que define as permissões máximas que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou função). Quando você definir um limite de permissões para uma entidade, a entidade poderá executar apenas as ações que são permitidas por ambas as políticas baseadas em identidade e seus 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 saber mais, consulte Determinar se uma solicitação é permitida ou negada em uma conta mais adiante neste tópico.
4. Políticas de controle de serviço (SCPs) do AWS Organizations: as SCPs do Organizations especificam as permissões máximas para uma organização ou unidade organizacional (UO). O máximo de SCP se aplica a entidades principais em contas-membro, incluindo cada Usuário raiz da conta da AWS. Se uma SCP estiver presente, as políticas baseadas em identidade e baseadas em recurso concedem permissões a entidades principais em contas-membro somente se essas políticas e a SCP permitirem a ação. Se um limite de permissões e uma SCP estiverem presentes, o limite, o SCP e a política baseada em identidade devem permitir a ação.
5. Políticas de sessão: as políticas de sessão são políticas avançadas que você passa como parâmetros ao criar uma sessão temporária de forma programática para uma função ou um 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`. Uma política baseada em recurso tem um efeito diferente na avaliação das permissões da política de sessão. A diferença depende de se o usuário ou o ARN da função ou o ARN da sessão está listado como a entidade principal na política baseada em recurso. Para obter mais informações, consulte Políticas de sessão.

Lembre-se de que uma negação explícita em qualquer uma dessas políticas substitui a permissão.

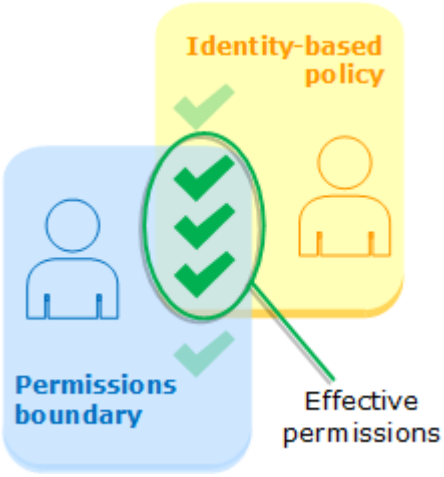
Avaliação das políticas baseadas em identidade com políticas baseadas em recurso

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 as permissões totais 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 do Organizations

Quando um usuário pertence a uma conta que é membro de uma organização, as permissões resultantes são a interseção das políticas do usuário e a SCP. Isso significa que uma ação deve ser permitida tanto pela política baseada em identidade quanto pelo SCP. Uma negação explícita em qualquer uma dessas políticas substitui a permissão.

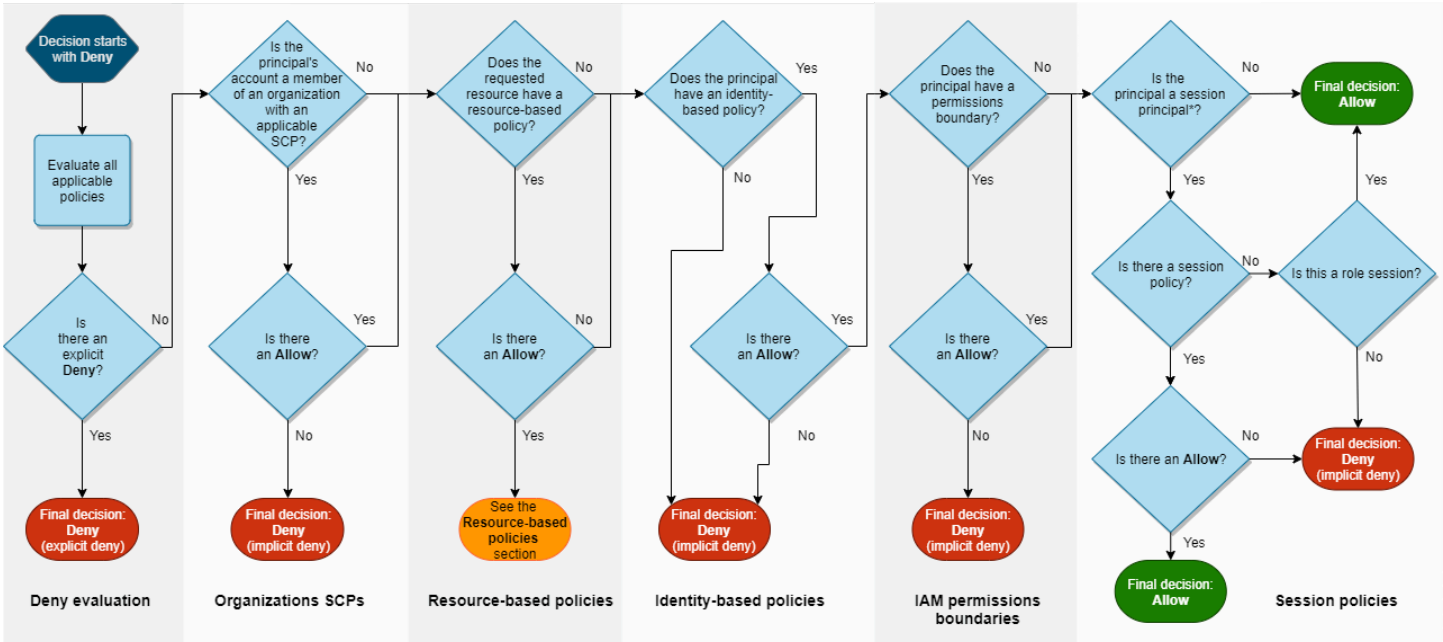
Você pode saber se sua conta é membro de uma organização no AWS Organizations. Os membros da organização podem ser afetados por uma SCP. Para visualizar esses dados usando o comando da AWS CLI ou a operação da API da AWS, você deve ter permissões para a ação `organizations:DescribeOrganization` da sua entidade do Organizations. Você deve ter permissões adicionais para executar a operação no console do 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.

Determinar se uma solicitação é permitida ou negada em uma conta

Suponha que uma entidade principal envie uma solicitação para a AWS para acessar um recurso na mesma conta que a entidade principal. O código de imposição da AWS decide se a solicitação deve ser permitida ou negada. A AWS avalia todas as políticas que são aplicáveis ao contexto da solicitação. A seguir apresentamos um resumo da lógica de avaliação da AWS para políticas em uma única conta.

- Por padrão, todas as solicitações são implicitamente negadas, com exceção do Usuário raiz da conta da AWS, que tem acesso total.
- Uma permissão explícita em uma política baseada em recurso ou identidade substitui esse padrão.
- Se houver um limite de permissões, uma SCP do Organizations ou uma política de sessão, isso poderá substituir a permissão com uma negação implícita.
- Uma negação explícita em qualquer política substitui todas as permissões.

O seguinte fluxograma fornece detalhes sobre como a decisão é tomada. Este fluxograma não aborda o impacto de políticas baseadas em recursos e negações implícitas em outros tipos de políticas.



*A session principal is either a role session or an IAM federated user session.

1. 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 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 uma instrução `Deny` que se aplica à solicitação. Esse processo é chamado de negação explícita. Se o código de imposição encontrar uma única negação explícita aplicável, o código retornará uma decisão final Deny (Negação). Se não houver uma negação explícita, a avaliação do código de imposição continuará.

2. SCPs do Organizations: em seguida, 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 uma declaração `Allow` aplicável nas SCPs, a solicitação será negada implicitamente, mesmo que a negação seja implícita. O código de imposição retorna uma decisão final de Deny (Negação). Se não houver uma SCP ou se a SCP permitir a ação solicitada, a avaliação do código de imposição continuará.
3. 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 uma permissão explícita 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.

   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 as sessões de função do IAM ou uma sessão de usuário federado do IAM. 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.

   A tabela a seguir ajuda a entender o impacto de políticas baseadas em recurso para diferentes tipos de entidades principais quando negações implícitas estão presentes em políticas baseadas em identidade, limites de permissões e políticas de sessão.

Políticas baseadas em recursos e negações implícitas em outros tipos de política (mesma conta)

Entidade principal fazendo a solicitação	Política baseada em recurso	Política baseada em identidade	Limite de permissões	Política de sessão	Result	Motivo
Perfil do IAM	Não aplicável	Não aplicável	Não aplicável	Não aplicável	Não aplicável	Uma função não pode fazer uma solicitação por conta própria. As solicitações são feitas com a sessão de função depois que uma função é assumida.

Entidade principal fazendo a solicitação	Política baseada em recurso	Política baseada em identidade	Limite de permissões	Política de sessão	Result	Motivo
Sessão de função do IAM	Permite o ARN de função	Negação implícita	Negação implícita	Negação implícita	DENY	O limite de permissões e a política de sessão são avaliados como parte da decisão final. Uma negação implícita em qualquer política resulta em uma decisão DENY (negar).

Entidade principal fazendo a solicitação	Política baseada em recurso	Política baseada em identidade	Limite de permissões	Política de sessão	Result	Motivo
Sessão de função do IAM	Permite ARN de sessão de função	Negação implícita	Negação implícita	Negação implícita	PERMISSÃC	As permissões são concedidas diretamente à sessão. Outros tipos de política não afetam a decisão.
Usuário do IAM	Permite o ARN de usuário do IAM	Negação implícita	Negação implícita	Não aplicável	PERMISSÃC	As permissões são concedidas diretamente ao usuário. Outros tipos de política não afetam a decisão.

Entidade principal fazendo a solicitação	Política baseada em recurso	Política baseada em identidade	Limite de permissões	Política de sessão	Result	Motivo
Usuário federado do IAM (`GetFederationToken`)	Permite o ARN de usuário do IAM	Negação implícita	Negação implícita	Negação implícita	DENY	Uma negação implícita seja no limite de permissões ou na política de sessão resulta em DENY (negar).
Usuário federado do IAM (`GetFederationToken`)	Permite ARN de sessão de usuário federado do IAM	Negação implícita	Negação implícita	Negação implícita	PERMISSÃC	As permissões são concedidas diretamente à sessão. Outros tipos de política não afetam a decisão.

Entidade principal fazendo a solicitação	Política baseada em recurso	Política baseada em identidade	Limite de permissões	Política de sessão	Result	Motivo
usuário raiz	Permite o ARN de usuário raiz	Não aplicável	Não aplicável	Não aplicável	PERMISSÃC	O usuário raiz tem acesso completo e irrestrito a todos os recursos na sua Conta da AWS. Para aprender como controlar o acesso de usuários raiz em contas no AWS Organizations, consulte Políticas de controle de serviços (SCPs), no Guia do usuário do Organizations.

Entidade principal fazendo a solicitação	Política baseada em recurso	Política baseada em identidade	Limite de permissões	Política de sessão	Result	Motivo
Principal do serviço da AWS	Permite uma entidade principal de serviço da AWS	Não aplicável	Não aplicável	Não aplicável	PERMISSÃC	Quando uma política baseada em recursos concede permissões diretamente a uma entidade principal de serviço da AWS, outros tipos de política não afetam a decisão.

- 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.

  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.

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
  ```

- Sessões de usuário federado do IAM: uma sessão de usuário federado do IAM 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 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 do IAM

  ```
  arn:aws:sts::111122223333:federated-user/exampleuser
  ```

4. Políticas baseadas em identidade: o código verifica as políticas baseadas em identidade para a entidade de segurança. 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

declarações em políticas baseadas em identidade que permitam a ação solicitada, a solicitação será implicitamente negada e o código retornará uma decisão final de Deny (Negar). Se uma declaração em qualquer política aplicável baseada em identidade permitir a ação solicitada, o código continuará.

5. Limites de permissões do IAM: em seguida, o código confere 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 retorna uma decisão final de Negação. Se não houver um limite de permissões ou se o limite de permissões permitir a ação solicitada, o código continuará.
6. Políticas de sessão: em seguida, o código confere se a entidade principal é uma entidade principal de sessão. As entidades principais de sessão incluem uma sessão de função do IAM ou uma sessão de usuário federado do IAM. 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 confere se uma política de sessão foi transmitida na solicitação. Você pode transmitir uma política de sessão enquanto usa a AWS CLI ou API da AWS para obter credenciais temporárias para uma função ou um usuário federado do IAM.

   - Se houver uma política de sessão presente e ela não permitir a ação solicitada, a solicitação será implicitamente negada. O código retorna uma decisão final de Negação.
   - Se não houver política de sessão, o código vai conferir se a entidade principal é uma sessão de função. Se a entidade principal for uma sessão de função, a solicitação será Permitida. Caso contrário, a solicitação é implicitamente negada e o código retorna uma decisão final de Deny (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).
7. Erros: Se o código de aplicação da AWS encontrar um erro em qualquer ponto durante a avaliação, ele gerará uma exceção e será fechado.

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 `carlossalazar-logs` do Amazon S3.

Suponha também que a política a seguir esteja anexada ao usuário do IAM `carlossalazar`.

```
{
    "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:::carlossalazar/*",
                "arn:aws:s3:::carlossalazar"
            ]
        },
        {
            "Sid": "DenyS3Logs",
            "Effect": "Deny",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::*log*"
        }
    ]
}
```

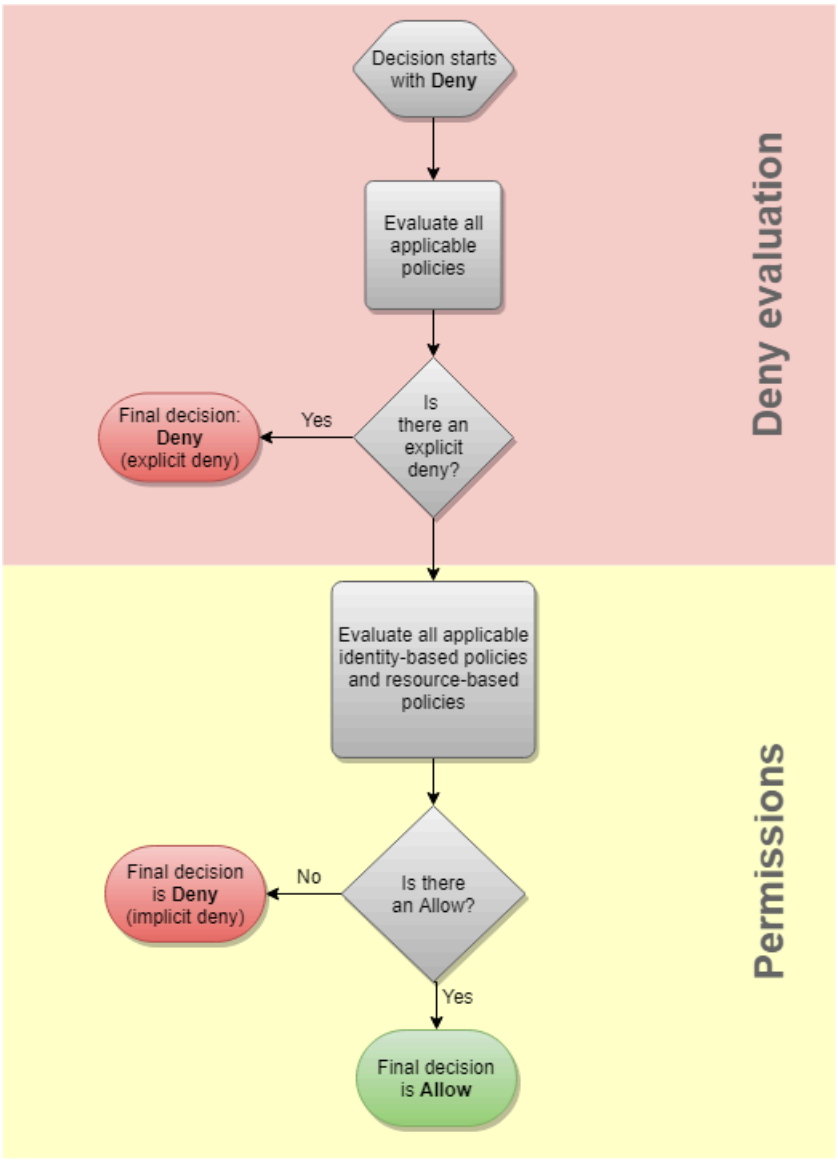
A instrução `AllowS3ListRead` nessa política permite que Carlos visualize uma lista de todos os buckets da conta. A instrução `AllowS3Self` permite a Carlos acesso total ao bucket com o mesmo nome que seu nome de usuário. A instrução `DenyS3Logs` nega a Carlos acesso a qualquer bucket do S3 com `log` em seu nome.

Além disso, a seguinte política baseada em recurso (chamada de política de bucket) está anexada ao bucket `carlossalazar`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::123456789012:user/carlossalazar"
            },
            "Action": "s3:*",
            "Resource": [
                "arn:aws:s3:::carlossalazar/*",
                "arn:aws:s3:::carlossalazar"
            ]
        }
    ]
}
```

Essa política especifica que apenas o usuário `carlossalazar` pode acessar o bucket `carlossalazar`.

Quando Carlos faz sua solicitação para salvar um arquivo no 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 `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.

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.

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

```
            "Action": "iam:*Report",
            "Resource": "*"
        }
    ]
}
```

Lógica de avaliação de política entre contas

Você pode 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. Você pode especificar toda a conta ou os usuários do IAM, usuários federados, funções do IAM ou sessões de função assumida. Você também pode especificar um serviço da AWS como principal. Para obter mais informações, consulte Especificar um principal.

A política baseada em identidade do principal deve permitir o acesso solicitado ao recurso no serviço de confiança. Você pode fazer isso ao especificar o ARN do recurso ou ao permitir acesso a todos os recursos (*).

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.

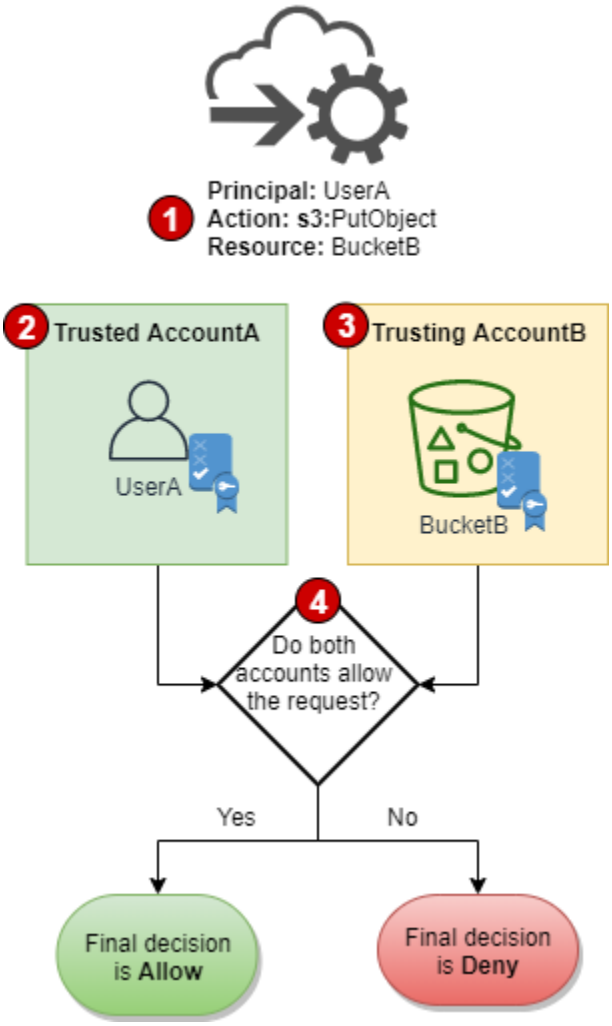
> **Important**
>
> Outros serviços podem afetar a lógica de avaliação da política. Por exemplo, o AWS Organizations oferece suporte a políticas de controle de serviço que podem ser aplicadas a uma ou mais contas dos principais. O AWS Resource Access Manager oferece suporte a

fragmentos de política que controlam quais ações os principais têm permissão para executar nos recursos compartilhados com eles.

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 `AccountB` de confiança. 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 Determinar se uma solicitação é permitida ou negada em uma conta. 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 de políticas em uma única conta.
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 de políticas em uma única conta.
4. A AWS permite 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 usuário 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 usuário do IAM chamada`carlossalazar` na conta 111111111111. Ele quer salvar um arquivo no bucket `Production-logs` do Amazon S3 na conta 222222222222.

Suponha também que a política a seguir esteja anexada ao usuário do IAM `carlossalazar`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowS3ListRead",
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Sid": "AllowS3ProductionObjectActions",
            "Effect": "Allow",
            "Action": "s3:*Object*",
            "Resource": "arn:aws:s3:::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 `Production`. A instrução `DenyS3Logs` nega a Carlos acesso a qualquer bucket do S3 com `log` em seu nome. Ela também nega o acesso a todos os objetos nesses buckets.

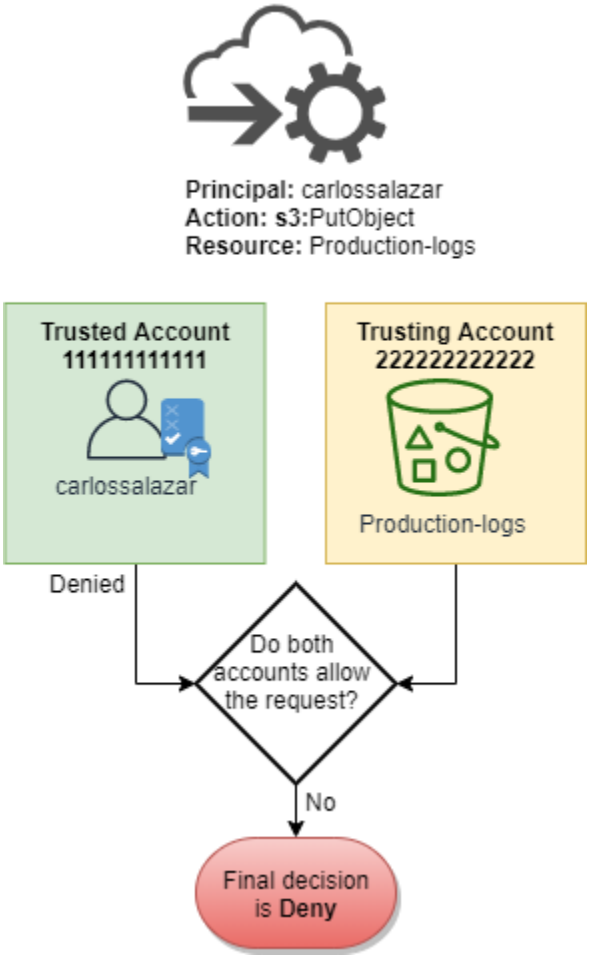
Além disso, a política baseada em recurso a seguir (chamada de política de bucket) está anexada ao bucket `Production` na conta 222222222222.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject*",
                "s3:ReplicateObject",
                "s3:RestoreObject"
            ],
            "Principal": { "AWS": "arn:aws:iam::111111111111:user/carlossalazar" },
            "Resource": "arn:aws:s3:::Production/*"
        }
    ]
}
```

Essa política permite que o usuário `carlossalazar` acesse objetos no bucket de `Production`. Ele poderá criar e editar, mas não excluir os objetos no bucket. Ele não conseguirá gerir o bucket propriamente dito.

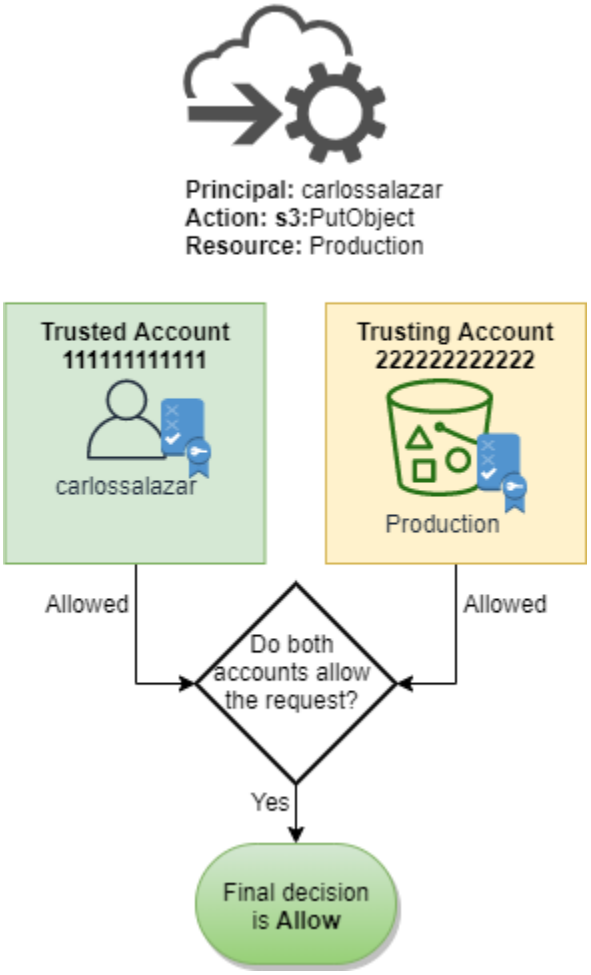
Quando Carlos faz sua solicitação para salvar um arquivo no bucket `Production-logs`, a AWS determina quais políticas são aplicáveis à solicitação. Nesse caso, a política baseada em identidade anexada ao usuário `carlossalazar` é a única política aplicável à conta `111111111111`. Na conta `222222222222`, não há uma política baseada em recursos anexada ao 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 Determinar se uma solicitação é permitida ou negada em uma conta.

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.



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 IAM
- 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 para instâncias do Linux.

- 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

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 políticas 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>, ...])

<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 ter 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 Especificar um 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).

```
"Resource":"arn:aws:iam::123456789012:user/Bob"

"Resource":"arn:aws:s3:::examplebucket/*"
```

`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 Chaves de contexto de condição globais da AWS. 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 para instâncias do Linux.

```
"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"
         ]
  }
}
```

Políticas gerenciadas pela AWS 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 Políticas gerenciadas pela AWS. 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 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.

Políticas gerenciadas pela AWS 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 AWS Management Console. 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 AWS Billing and Cost Management com as permissões nesta política, você deve primeiro ativar o usuário e o acesso à função do IAM. Para fazer isso, siga as instruções na Etapa 1 do tutorial sobre como 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 AWS Billing and Cost Management. 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 AWS Billing and Cost Management com as permissões nesta política, você deve primeiro ativar o usuário e o acesso à função do IAM. Para fazer isso, siga as instruções na Etapa 1 do tutorial sobre como 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.

Funções de serviço opcionais do IAM para a função de trabalho do administrador de banco de dados

Caso de uso	Nome da função (* é um curinga)	Tipo de função 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
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	AWSLambdaDynamoDBExecutionRole

Caso de uso	Nome da função (* é um curinga)	Tipo de função de serviço a selecionar	Selecione esta política gerenciada pela AWS
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	AWSLambdaVPCAccessExecutionRole
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. ConsulteFunções do IAM para oAWS 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 análise de dados 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 realizar análise de dados com ferramentas como o Amazon QuickSight. A política inclui o acesso a serviços adicionais de cientistas de dados, como o 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 função para o SageMaker. 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.

Funções de serviço opcionais do IAM para a função de trabalho do cientista de dados

Caso de uso	Nome da função (* é um curinga)	Tipo de função de serviço a selecionar	Política gerenciada pela AWS 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	Consulte o Blog de Big Data da AWS, que descreve quatro opções possíveis

Caso de uso	Nome da função (* é um curinga)	Tipo de função de serviço a selecionar	Política gerenciada pela AWS a selecionar
		no Blog de Big Data da AWS.	, 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. ConsulteFunções do IAM para oAWS 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	AmazonEC2 RoleforDataPipelineRole

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 Account Management. 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 Organizations para visualizar informações sobre a organização do usuário, incluindo o e-mail da conta de gerenciamento e as limitações da organização. Embora esta política limite o IAM e o Organizations, ela permite que o usuário execute todas as ações do IAM Identity Center, se o IAM Identity Center 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.

Esta função de trabalho requer a capacidade de passar funções 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.

Funções de serviço opcionais do IAM para a função de trabalho do administrador de rede

Caso de uso	Nome da função (* é um curinga)	Tipo de função de serviço a selecionar	Política gerenciada pela AWS 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 gerenciada pela AWS existente, mas a documentação lista as permissões necessárias. Consulte Guia do 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.

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 AWS Billing and Cost Management. Visualize a política para obter a lista completa de serviços e ações a que esta política oferece suporte.

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 pela AWS: SupportUser

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 AWS Support. Para obter mais informações sobre a política gerenciada, consulte SupportUser no Guia de referência de políticas gerenciadas pela 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.

Esta função de trabalho requer a capacidade de passar funções 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.

Funções de serviço opcionais do IAM para a função de trabalho do administrador de sistemas

Caso de uso	Nome da função (* é um curinga)	Tipo de função de serviço a selecionar	Política gerenciada pela AWS 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	AmazonEC2ContainerServiceforEC2Role
Permitir que um usuário monitore bancos de dados	rds-monitoring-role	Função do Amazon RDS para monitoramento aprimorado	AmazonRDSEnhancedMonitoringRole
Permitir que aplicativos em execução em instâncias do EC2 acessem recursos da AWS.	ec2-sysadmin-*	Amazon EC2	Política de amostra para função que concede acesso a um bucket do S3, conforme mostrado no Guia do usuário do Amazon EC2 para instâncias do

Caso de uso	Nome da função (* é um curinga)	Tipo de função de serviço a selecionar	Política gerenciada pela AWS a selecionar
			Linux; personalizar conforme necessário
Permitir que o Lambda leia fluxos do DynamoDB e grave no CloudWatch Logs	lambda-sysadmin-*	AWS Lambda	AWSLambdaDynamoDBExecutionRole

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 AWS Management Console 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 política de permissão alguma, 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 Marcar recursos do IAM no Guia do usuário do IAM.
13. Reveja a função e escolha Create role (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. Você pode 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 AWS Management Console 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 da função, 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 AWS Management Console 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 Attach policy (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.

```
{
  "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, erros ou avisos gerais gerados durante a validação de política e depois escolha Avançar.

> **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 ter 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 da função, 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 Roles (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.

Chaves de contexto de condição globais da AWS

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 ter mais informações, consulte Chaves de contexto de múltiplos valores.

Propriedades da entidade principal	Propriedades de uma sessão de perfil	Propriedades da rede	Propriedades do recurso	Propriedades da solicitação
aws:PrincipalArn	aws:FederatedProvider	aws:SourceIp	aws:ResourceAccount	aws:CalledVia

Propriedades da entidade principal	Propriedades de uma sessão de perfil	Propriedades da rede	Propriedades do recurso	Propriedades da solicitação
aws:PrincipalAccount aws:PrincipalOrgPaths aws:PrincipalOrgID aws:PrincipalTag/tag-key aws:PrincipalIsAWSService aws:PrincipalServiceName aws:PrincipalServiceNamesList aws:PrincipalType aws:userid aws:username	aws:TokenIssueTime aws:MultiFactorAuthAge aws:MultiFactorAuthPresent aws:Ec2InstanceSourceVpc aws:Ec2InstanceSourcePrivateIPv4 aws:SourceIdentity ec2:RoleDelivery ec2:SourceInstanceArn glue:RoleAssumedBy glue:CredentialIssuingService lambda:SourceFunctionArn	aws:SourceVpc aws:SourceVpce aws:VpcSourceIp	aws:ResourceOrgPaths aws:ResourceOrgID aws:ResourceTag/tag-key	aws:CalledViaFirst aws:CalledViaLast aws:ViaAWSService aws:CurrentTime aws:EpochTime aws:referer aws:RequestedRegion aws:RequestTag/tag-key aws:TagKeys aws:SecureTransport aws:SourceArn aws:SourceAccount aws:SourceOrgPaths aws:SourceOrgID aws:UserAgent

Propriedades da entidade principal	Propriedades de uma sessão de perfil	Propriedades da rede	Propriedades do recurso	Propriedades da solicitação
	ssm:SourceInstanceArn identitystore:UserId			

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 Especificar um principal.

Sumário

- aws:PrincipalArn
- aws:PrincipalAccount
- aws:PrincipalOrgPaths
- aws:PrincipalOrgID
- aws:PrincipalTag/tag-key
- aws:PrincipalIsAWSService
- aws:PrincipalServiceName
- aws:PrincipalServiceNamesList
- aws:PrincipalType
- aws:userid
- aws:username

aws:PrincipalArn

Use essa chave para comparar o nome de 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. Você pode especificar os seguintes tipos de entidades principais nesta chave de condição:
  - IAM role (Perfil do IAM)
  - IAM user (Usuário do IAM)
  - Sessão de usuário federado do AWS STS
  - Usuário raiz da Conta da AWS
- Tipo de dados: ARN, String

  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 :
  - 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
    ```
  - Sessões de usuário federado do AWS STS: o contexto da solicitação contém o seguinte valor 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 seguinte valor 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 Entidades principais da Conta da AWS.

```
arn:aws:iam::123456789012:root
```

Você pode 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 SCP effects on permissions (Efeitos das SCPs nas permissões) no Guia do usuário do 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`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAccessFromPrincipalNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:accountID: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, um usuário federado ou um 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 do AWS Organizations é uma representação de texto da estrutura de uma entidade do Organizations. Para obter mais informações sobre como usar e entender os caminhos, consulte Compreender o caminho da entidade do AWS Organizations.

- Disponibilidade: essa chave é incluída no contexto da solicitação somente se a entidade de segurança 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 Chaves de contexto de múltiplos 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 de segurança 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. Você pode 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 `policy-ninja-dev`.

```
{
 "Version": "2012-10-17",
 "Statement": {
   "Sid": "AllowPutObject",
   "Effect": "Allow",
   "Principal": "*",
   "Action": "s3:PutObject",
   "Resource": "arn:aws:s3:::policy-ninja-dev/*",
   "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 de segurança estiver usando um usuário do IAM com etiquetas anexadas. Ela será incluída para uma entidade de segurança usando uma função do IAM com etiquetas 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 Recursos de etiquetas do IAM. Você pode usar `aws:PrincipalTag` para controlar acesso de 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.

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

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iam:*",
      "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 de segurança do produto da AWS. Por exemplo, o AWS CloudTrail usa a entidade de segurança `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 de segurança 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 de segurança do IAM para fazer uma solicitação em nome da entidade de segurança. Também é definida como false se o serviço usar uma função de serviço ou função vinculada ao serviço para fazer uma chamada em nome da entidade de segurança.

- 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

Você pode 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 de segurança de serviço, como o CloudTrail.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Expected-network+service-principal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-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 de segurança de serviço na política com a entidade de segurança de serviço que está fazendo solicitações aos seus recursos. Você pode usar essa chave para verificar se essa chamada é feita por uma entidade de segurança de serviço específica. Quando uma entidade de segurança do serviço faz uma solicitação direta ao seu recurso, a chave `aws:PrincipalServiceName` contém o nome da entidade de segurança do serviço. Por exemplo, o nome da entidade de segurança do serviço AWS CloudTrail é `cloudtrail.amazonaws.com`.

- Disponibilidade: essa chave está presente na solicitação quando a chamada é feita por uma entidade de segurança de produto da AWS. Esta chave não está presente em nenhuma outra situação, incluindo a seguinte:

  - se o serviço usa uma função de serviço ou uma função vinculada ao serviço para fazer uma chamada em nome da entidade de segurança.
  - Se o serviço usar as credenciais de uma entidade de segurança do IAM para fazer uma solicitação em nome da entidade de segurança.
  - Se a chamada for feita diretamente por uma entidade de segurança do IAM.
  - Se a chamada for feita por um solicitante anônimo.
- Tipo de dados: String
- Tipo de valor: valor único

Você pode 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 de segurança de serviço, como o CloudTrail.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "expected-network+service-principal",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-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 de segurança de serviço que pertencem ao serviço. Esta é uma chave de condição avançada. Você pode usá-la para restringir o acesso do serviço ao seu recurso somente de uma região específica. Alguns serviços podem criar entidades de serviço regionais para indicar uma instância específica do serviço dentro de uma região específica. Você pode limitar o acesso a um recurso para uma instância específica do serviço. Quando uma entidade de segurança 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 de segurança de produto da AWS. Esta chave não está presente em nenhuma outra situação, incluindo a seguinte:
  - se o serviço usa uma função de serviço ou uma função vinculada ao serviço para fazer uma chamada em nome da entidade de segurança.
  - Se o serviço usar as credenciais de uma entidade de segurança do IAM para fazer uma solicitação em nome da entidade de segurança.
  - Se a chamada for feita diretamente por uma entidade de segurança 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 Chaves de contexto de múltiplos 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 ter mais informações, consulte Especificar um 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 Especificar um 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 Especificar um 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 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.

Sumário

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 quando a entidade principal é uma entidade principal da sessão do perfil e a sessão foi emitida usando um perfil que foi assumido com `AssumeRoleWithWebIdentity`.
- Tipo de dados: String
- Tipo de valor: valor único

Por exemplo, se o usuário foi autenticado por meio do Amazon Cognito, o contexto da solicitação incluirá o valor `cognito-identity.amazonaws.com`. Da mesma forma, se o usuário foi autenticado por meio do Login with Amazon, o contexto da solicitação incluirá o valor `www.amazon.com`.

É possível usar qualquer chave de condição de valor único disponível como uma variável. O seguinte exemplo de política com base em recursos usa a chave `aws:FederatedProvider` como uma variável de política no ARN de um recurso. Essa política permite que qualquer entidade principal que fez autenticação usando um IdP obtenha objetos de um bucket do Amazon S3 com um caminho que é específico para o provedor de identidade emissor.

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 de segurança 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 Serviços da AWS que funcionam com o IAM.

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 Uso de autenticação multifator (MFA) na AWS.

> **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 Configuração de acesso à API protegido por 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 Configuração de acesso à API protegido por 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 de segurança 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 AWS Management Console.

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 AWS Management Console 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 AWS 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 Serviços da AWS 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:Ec2InstanceSourceVpc

Essa chave identifica a VPC para a qual as credenciais de perfil do IAM do Amazon EC2 foram entregues. Você pode 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.

> **Note**
>
> Essa chave de condição não está disponível no EC2-Classic.

```
{
  "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 garantir que você tenha uma combinação global exclusiva de ID da VPC e IP privado de origem. Use essa chave com `aws:Ec2InstanceSourceVpc` para garantir 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.

Note

Essa chave de condição não está disponível no EC2-Classic.

```
{
    "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 de segurança 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

Você pode usar essa chave em uma política para permitir ações na AWS executadas por entidades de segurança que definiram uma identidade-fonte ao assumir uma função. 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 de segurança inicialmente define uma identidade-fonte enquanto assume uma função usando qualquer comando de assumir função 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 função a seguir para `CriticalRole` na conta `111122223333` contém uma condição para `aws:SourceIdentity` que impede que uma entidade de segurança sem uma identidade-fonte definida como Saanvi ou Diego assuma a função.

```
{
    "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.

```
{
    "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:::confidential-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:::confidential-bucket/*",
    "Condition": {
        "StringEquals": {
            "glue:CredentialIssuingService": "glue.amazonaws.com"
        }
    }
}
```

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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ExampleSourceFunctionArn",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-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

Você pode 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.

Sumário

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 `DOC-EXAMPLE-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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-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 `DOC-EXAMPLE-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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PrincipalPutObjectIfIpAddress",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-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.

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

aws:SourceVpce

Use essa chave para comparar o identificador do VPC endpoint da solicitação com o ID do endpoint especificado na política. Em uma política, é possível usar essa chave para restringir o acesso a um VPC endpoint específico. Para mais informações, consulte Restringir o acesso a um endpoint da VPC específico 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

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 a serviços com VPC endpoints no Guia do usuário da Amazon VPC.

> **Note**
>
> O `aws:VpcSourceIp` 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.

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.

Sumário

aws:ResourceAccount

Use essa chave para comparar o Conta da AWSID 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`
  - Amazon Elastic Block Store: todas as ações
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CopyImage`
    - `ec2:CopySnapshot`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVolume`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
    - `ec2:DeleteTransitGatewayPeeringAttachment`
    - `ec2:DeleteVpcPeeringConnection`

    - `ec2:RejectTransitGatewayPeeringAttachment`
    - `ec2:RejectVpcEndpointConnections`
    - `ec2:RejectVpcPeeringConnection`
  - 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`
    - `es:CreateOutboundConnection`
  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
    - `route53:DeleteVPCAssociationAuthorization`
    - `route53:DisassociateVPCFromHostedZone`
    - `route53:ListHostedZonesByVPC`
  - AWS Security Hub
    - `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:::DOC-EXAMPLE-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.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyInteractionWithResourcesNotInSpecificAccount",
      "Action": "service:*",
      "Effect": "Deny",
      "Resource": [
        "arn:aws:service:region:account:*"
      ],
      "Condition": {
```

```
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "account"
          ]
        }
      }
    }
  ]
}
```

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 de segurança 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`
  - Amazon Elastic Block Store: todas as ações
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CopyImage`
    - `ec2:CopySnapshot`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVolume`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
    - `ec2:DeleteTransitGatewayPeeringAttachment`
    - `ec2:DeleteVpcPeeringConnection`
    - `ec2:RejectTransitGatewayPeeringAttachment`
    - `ec2:RejectVpcEndpointConnections`
    - `ec2:RejectVpcPeeringConnection`
  - 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`
    - `es:CreateOutboundConnection`
  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
    - `route53:DeleteVPCAssociationAuthorization`
    - `route53:DisassociateVPCFromHostedZone`
    - `route53:ListHostedZonesByVPC`
  - AWS Security Hub
    - `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 Chaves de contexto de múltiplos 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. Você pode 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 de segurança 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`
  - Amazon Elastic Block Store: todas as ações
  - Amazon EC2
    - `ec2:AcceptTransitGatewayPeeringAttachment`
    - `ec2:AcceptVpcEndpointConnections`
    - `ec2:AcceptVpcPeeringConnection`
    - `ec2:CopyImage`
    - `ec2:CopySnapshot`
    - `ec2:CreateTransitGatewayPeeringAttachment`
    - `ec2:CreateVolume`
    - `ec2:CreateVpcEndpoint`
    - `ec2:CreateVpcPeeringConnection`
    - `ec2:DeleteTransitGatewayPeeringAttachment`
    - `ec2:DeleteVpcPeeringConnection`
    - `ec2:RejectTransitGatewayPeeringAttachment`
    - `ec2:RejectVpcEndpointConnections`
    - `ec2:RejectVpcPeeringConnection`
  - 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`
    - `es:CreateOutboundConnection`
  - Amazon Route 53
    - `route53:AssociateVpcWithHostedZone`
    - `route53:CreateVPCAssociationAuthorization`
    - `route53:DeleteVPCAssociationAuthorization`
    - `route53:DisassociateVPCFromHostedZone`
    - `route53:ListHostedZonesByVPC`
  - AWS Security Hub
    - `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.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "DenyPutObjectToS3ResourcesOutsideMyOrganization",
    "Effect": "Deny",
    "Action": "s3:PutObject",
    "Resource": "arn:partition: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. Você pode 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 ter 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 e os valores de etiquetas 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.

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.

Sumário

aws:CalledVia

Use essa chave para comparar os serviços na política com os serviços que fizeram solicitações em nome da entidade de segurança 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. A chave `aws:CalledVia` contém uma lista ordenada de cada serviço na cadeia que fez solicitações em nome do principal.

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 peloAWS Key Management Service (AWS KMS).

- Disponibilidade: essa chave está presente na solicitação quando um serviço que oferece suporte a `aws:CalledVia` usa as credenciais de uma entidade de segurança do IAM para fazer uma solicitação a outro serviço. Esta chave não está presente se o serviço usa uma função de serviço ou função vinculada ao serviço para fazer uma chamada em nome do 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 de serviço para conceder ou negar solicitações de serviço da AWS. A AWS oferece suporte ao uso das seguintes entidades principais de serviço com `aws:CalledVia`.

Entidade principal do serviço
aoss.amazonaws.com
athena.amazonaws.com
backup.amazonaws.com
cloud9.amazonaws.com
cloudformation.amazonaws.com
databrew.amazonaws.com
dataexchange.amazonaws.com
dynamodb.amazonaws.com
imagebuilder.amazonaws.com

Entidade principal do serviço
kms.amazonaws.com
mgn.amazonaws.com
nimble.amazonaws.com
omics.amazonaws.com
ram.amazonaws.com
robomaker.amazonaws.com
servicecatalog-appregistry.amazonaws.com
sqlworkbench.amazonaws.com
ssm-guiconnect.amazonaws.com

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 AWS 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 via AWS CloudFormation e, depois, via DynamoDB.

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 é 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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaDynamodb",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:region: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:CalledViaLast` e `aws:CalledViaFirst`. 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 por meio do AWS CloudFormation e a última pelo DynamoDB. Se outros serviços fizerem solicitações no meio da cadeia, a operação ainda será permitida.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "KmsActionsIfCalledViaChain",
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey",
                "kms:DescribeKey"
```

```
            ],
            "Resource": "arn:aws:kms:region: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 de segurança 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 AWS 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` via AWS CloudFormation, depois, `X Service` e depois DynamoDB. Ele foi chamado pela primeira vez por meio do AWS CloudFormation e a última chamada foi feita por meio do DynamoDB.

aws:CalledViaFirst

Use esta chave para comparar os serviços na política com o primeiro serviço que fez uma solicitação em nome da entidade de segurança do IAM (usuário ou função). Para ter mais informações, consulte `aws:CalledVia`.

- Disponibilidade: essa chave está presente na solicitação quando um serviço usa as credenciais de uma entidade de segurança do IAM para fazer pelo menos uma outra solicitação a um serviço diferente. Esta chave não está presente se o serviço usa uma função de serviço ou função vinculada ao serviço para fazer uma chamada em nome do 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 de segurança do IAM (usuário ou função). Para ter mais informações, consulte `aws:CalledVia`.

- Disponibilidade: essa chave está presente na solicitação quando um serviço usa as credenciais de uma entidade de segurança do IAM para fazer pelo menos uma outra solicitação a um serviço diferente. Esta chave não está presente se o serviço usa uma função de serviço ou função vinculada ao serviço para fazer uma chamada em nome do 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 esta chave para verificar se um serviço da AWS faz uma solicitação para outro serviço em seu nome.

A chave de contexto da solicitação retorna `true` quando um serviço usa as credenciais de uma entidade de segurança do IAM para fazer uma solicitação em nome da entidade de segurança. Esta chave retorna `false` se o serviço usa uma função de serviço ou função vinculada ao serviço para fazer uma chamada em nome do principal. 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

Você pode usar essa chave de condição para conceder ou negar acesso baseado no fato de uma solicitação ter sido feita por um serviço.

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. Você pode 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.
- Você pode 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 AWS Management Console. No entanto, eles só podem realizar alterações às instâncias na Irlanda (eu-west-1), Londres (eu-west-2) ou Paris (eu-west-3).

```
{
    "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 ter 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 e os valores de etiquetas 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.

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.

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

```
    "Effect": "Allow",
    "Action": "ec2:CreateTags",
    "Resource": "arn:aws:ec2:::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 ter mais informações, consulte Chaves de contexto de múltiplos 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 este cenário que use `aws:TagKeys`, consulte Criar um snapshot com etiquetas no Guia do usuário do Amazon EC2 para instâncias do Linux.

aws:SecureTransport

Use essa chave para verificar se a solicitação foi enviada usando SSL. O contexto da solicitação retorna `true` ou `false`. Em uma política, será possível permitir ações específicas somente se a solicitação for enviada usando SSL.

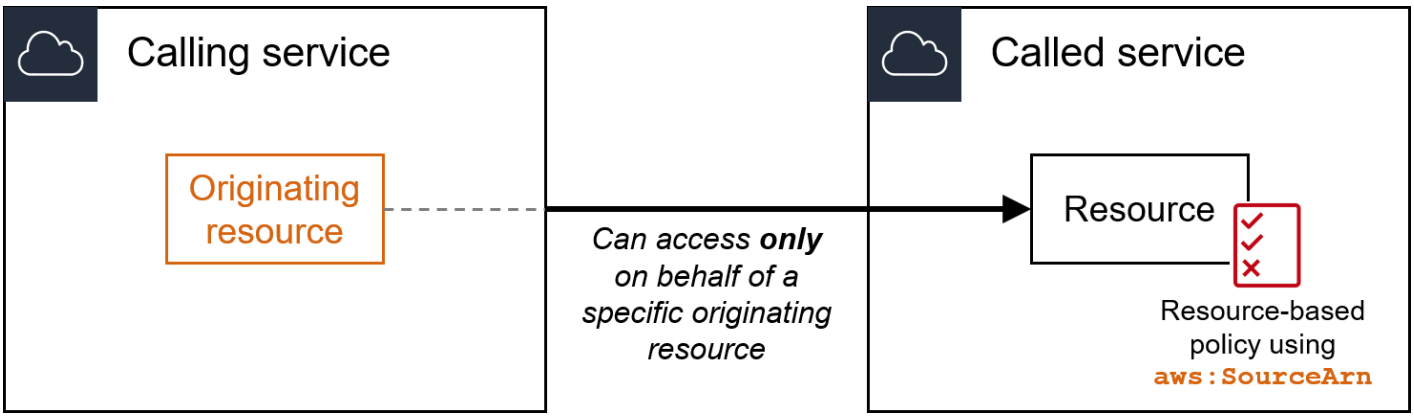
- Disponibilidade: essa chave é sempre incluída no contexto da solicitação.
- Tipo de dados: Booleano
- Tipo de valor: valor único

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.



As seguintes integrações de serviços não são compatíveis com essa chave de condição global:

Serviço que chama (entidade principal do serviço)	Serviço chamado (política baseada em recursos)	Descrição
logdelivery.elb.amazonaws.com	Bucket do Amazon S3	Habilitar o registro em log de acesso do Elastic Load Balancing em um bucket do Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket do Amazon S3	Habilitar o registro em log de acesso do Elastic Load Balancing em um bucket do Amazon S3

> **Note**
>
> Nem todas as integrações de serviços com o AWS Security Token Service (AWS STS) e AWS Key Management Service (AWS KMS) são compatíveis. Consulte a documentação do serviço que faz a chamada para obter mais informações. O uso de `aws:SourceArn` nas políticas de chave do KMS para as chaves usadas pelos Serviços da AWS por meio de concessões de chaves do KMS pode causar um comportamento inesperado.

- Tipo de dados: ARN, String

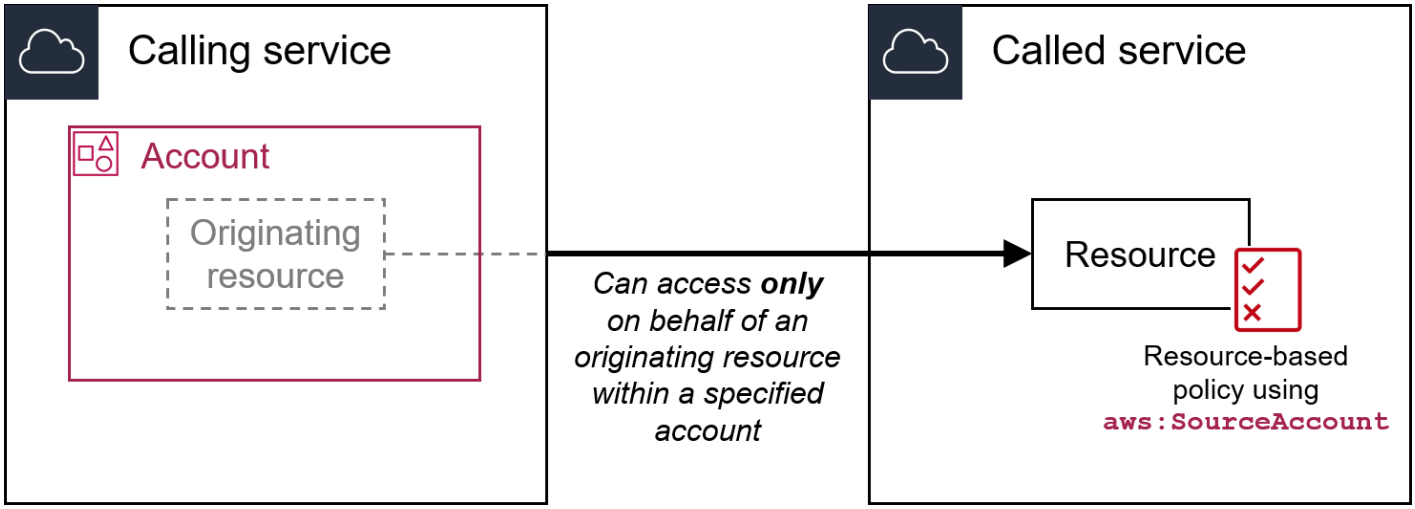
  A AWS recomenda utilizar operadores ARN em vez de operadores string ao comparar ARNs.
- Tipo de valor: valor único

Você pode usar essa chave de condição para impedir que um produto da AWS seja usado como um confused deputy durante transações entre os serviços. Só use essa chave em políticas baseadas em recursos nas quais a `Principal` é a entidade principal do AWS service (Serviço da AWS). Defina o valor desta chave de condição para o ARN do recurso na solicitação. Por exemplo, quando a atualização de um bucket do Amazon S3 aciona a publicação de um tópico do Amazon SNS, o serviço Amazon S3 invoca a operação de API `sns:Publish`. 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 informações sobre como e quando essa chave de condição é recomendada, consulte a documentação dos serviços da AWS que você está usando.

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.



  As seguintes integrações de serviços não são compatíveis com essa chave de condição global:

Serviço que chama (entidade principal do serviço)	Serviço chamado (política baseada em recursos)	Descrição
logdelivery.elb.amazonaws.com	Bucket do Amazon S3	Habilitar o registro em log de acesso do Elastic Load Balancing em um bucket do Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket do Amazon S3	Habilitar o registro em log de acesso do Elastic Load Balancing em um bucket do Amazon S3

> **Note**
>
> Nem todas as integrações de serviços com o AWS Security Token Service (AWS STS) e AWS Key Management Service (AWS KMS) são compatíveis. Consulte a documentação do serviço que faz a chamada para obter mais informações. O uso de `aws:SourceAccount` nas políticas de chave do KMS para as chaves usadas pelos Serviços da AWS por meio de concessões de chaves do KMS pode causar um comportamento inesperado.

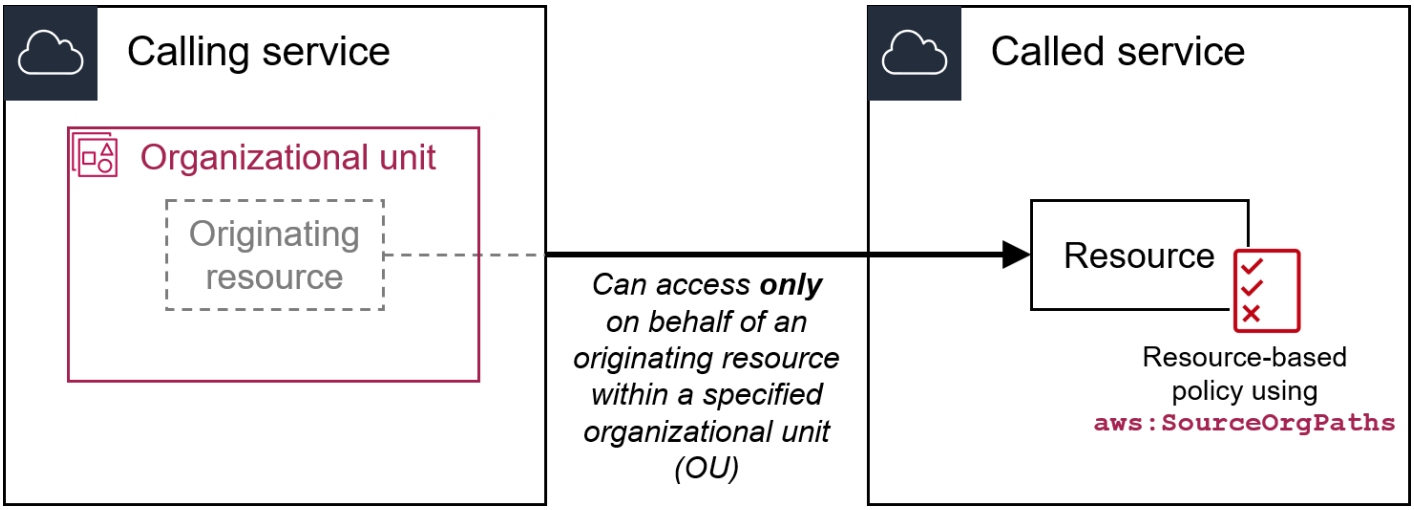
- Tipo de dados: String
- Tipo de valor: valor único

Você pode usar essa chave de condição para impedir que um produto da AWS seja usado como um confused deputy durante transações entre os serviços. Só use essa chave em políticas baseadas em recursos nas quais a `Principal` é a entidade principal do AWS service (Serviço da AWS). Defina o valor dessa chave de condição como o ID da conta do recurso na solicitação. Por exemplo, quando a atualização de um bucket do Amazon S3 aciona a publicação de um tópico do Amazon SNS, o serviço Amazon S3 invoca a operação de API `sns:Publish`. 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. Para obter informações sobre como e quando essas chaves de condições são recomendadas, consulte a documentação dos serviços da AWS que você está usando.

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 do Organizations é uma representação de texto da estrutura de uma entidade do 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.



As seguintes integrações de serviços não são compatíveis com essa chave de condição global:

Serviço que chama (entidade principal do serviço)	Serviço chamado (política baseada em recursos)	Descrição
logdelivery.elb.amazonaws.com	Bucket do Amazon S3	Habilitar o registro em log de acesso do Elastic Load Balancing em um bucket do Amazon S3
logdelivery.elasticloadbalancing.amazonaws.com	Bucket do Amazon S3	Habilitar o registro em log de acesso do Elastic Load

Serviço que chama (entidade principal do serviço)	Serviço chamado (política baseada em recursos)	Descrição
		Balancing em um bucket do Amazon S3
Todas as entidades principais do serviço	Bot do Amazon Lex	Permitir que os Serviços da AWS usem o bot do Amazon Lex

> **Note**
>
> Nem todas as integrações de serviços com o AWS Security Token Service (AWS STS) e AWS Key Management Service (AWS KMS) são compatíveis. Consulte a documentação do serviço que faz a chamada para obter mais informações. O uso de `aws:SourceOrgPaths` nas políticas de chave do KMS para as chaves usadas pelos Serviços da AWS por meio de concessões de chaves do KMS pode causar um comportamento inesperado.

- Tipo de dados: String (lista)
- Tipo de valor: valores múltiplos

Você pode usar essa chave de condição para impedir que um produto da AWS seja usado como um confused deputy durante transações entre os serviços. Só use essa chave em políticas baseadas em recursos nas quais a `Principal` é a entidade principal do AWS service (Serviço da AWS). Defina o valor dessa chave de condição como o caminho da organização do recurso na solicitação. Por exemplo, quando a atualização de um bucket do Amazon S3 aciona a publicação de um tópico do Amazon SNS, o serviço Amazon S3 invoca a operação de API `sns:Publish`. Na política de tópico que permite a operação `sns:Publish`, defina o valor da chave de condição como o caminho da organização do bucket do Amazon S3. Para obter informações sobre como e quando essa chave de condição é recomendada, consulte a documentação dos serviços da AWS que você está usando.

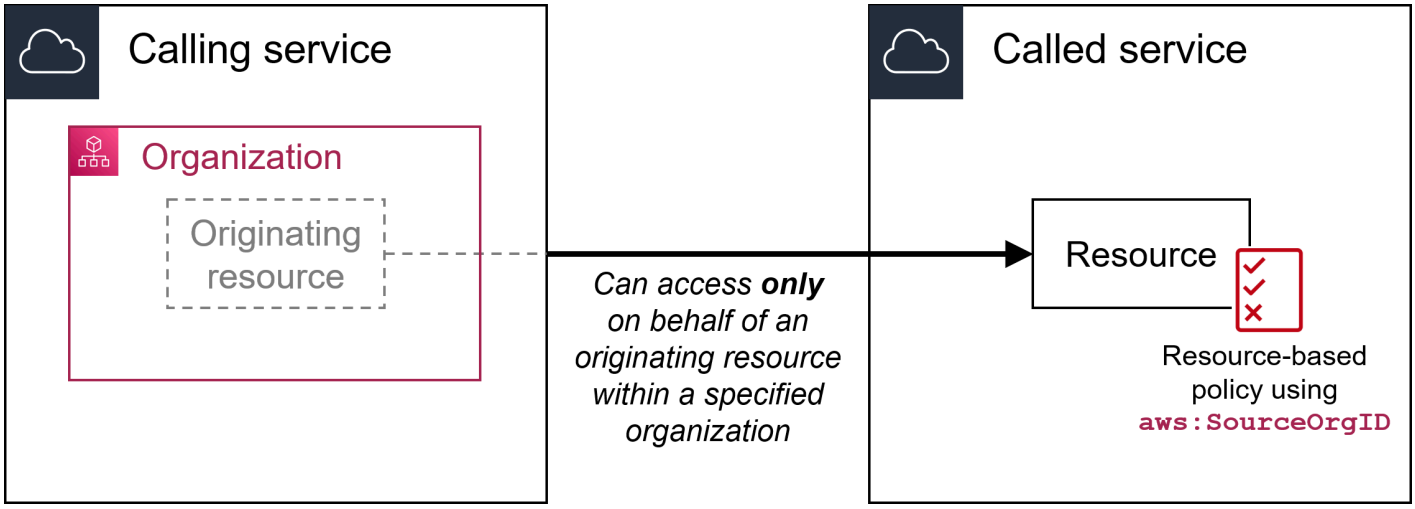
`aws:SourceOrgPaths` é uma chave de condição de vários valores. Chaves de valores múltiplos podem ter vários valores no contexto da solicitação. Você deve usar os operadores de conjunto `ForAnyValue` ou `ForAllValues` com operadores de condição de string para essa chave. Para

obter mais informações sobre chaves de condição de vários valores, consulte Chaves de contexto de múltiplos valores.

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.



As seguintes integrações de serviços não são compatíveis com essa chave de condição global:

Serviço que chama (entidade principal do serviço)	Serviço chamado (política baseada em recursos)	Descrição
logdelivery.elb.amazonaws.com	Bucket do Amazon S3	Habilitar o registro em log de acesso do Elastic Load Balancing em um bucket do Amazon S3

Serviço que chama (entidade principal do serviço)	Serviço chamado (política baseada em recursos)	Descrição
logdelivery.elasticloadbalancing.amazonaws.com	Bucket do Amazon S3	Habilitar o registro em log de acesso do Elastic Load Balancing em um bucket do Amazon S3
Todas as entidades principais do serviço	Bot do Amazon Lex	Permitir que os Serviços da AWS usem o bot do Amazon Lex

> **Note**
>
> Nem todas as integrações de serviços com o AWS Security Token Service (AWS STS) e AWS Key Management Service (AWS KMS) são compatíveis. Consulte a documentação do serviço que faz a chamada para obter mais informações. O uso de `aws:SourceOrgID` nas políticas de chave do KMS para as chaves usadas pelos Serviços da AWS por meio de concessões de chaves do KMS pode causar um comportamento inesperado.

- Tipo de dados: String
- Tipo de valor: valor único

Você pode usar essa chave de condição para impedir que um produto da AWS seja usado como um confused deputy durante transações entre os serviços. Só use essa chave em políticas baseadas em recursos nas quais a `Principal` é a entidade principal do AWS service (Serviço da AWS). Defina o valor dessa chave de condição como o ID da organização do recurso na solicitação. Por exemplo, quando a atualização de um bucket do Amazon S3 aciona a publicação de um tópico do Amazon SNS, o serviço Amazon S3 invoca a operação de API `sns:Publish`. Na política de tópico que permite a operação `sns:Publish`, defina o valor da chave de condição como o ID da organização do bucket do Amazon S3. Para obter informações sobre como e quando essa chave de condição é recomendada, consulte a documentação dos serviços da AWS que você está usando.

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. Você pode usá-lo para permitir apenas aplicações cliente específicas, e somente depois de testar sua política.

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 SAML executa AWS operações em outros serviços.

Chaves de contexto de condição do IAM e do AWS STS

Você pode 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. Você pode 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

Você pode usar as seguintes chaves de condição em políticas que controlam o acesso aos recursos do IAM:

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.

Use essa chave de condição em uma política para permitir que uma entidade transmita uma função, mas somente se essa função estiver associada ao recurso especificado. Você pode usar curingas (*) para permitir operações executadas em um tipo específico de recurso sem restringir a região ou o ID do recurso. Por exemplo, você pode permitir que um usuário ou uma função do IAM passe qualquer função para o serviço Amazon EC2 para ser usado com instâncias na região `us-east-1` ou `us-west-1`. O usuário ou a função do IAM não terá permissão para passar funções para outros serviços. Além disso, ele não permite que o Amazon EC2 use a função com instâncias em outras regiões.

```
{
```

```
    "Effect": "Allow",
    "Action": "iam:PassRole",
    "Resource": "*",
    "Condition": {
        "StringEquals": {"iam:PassedToService": "ec2.amazonaws.com"},
        "ArnLike": {
            "iam:AssociatedResourceARN": [
                "arn:aws:ec2:us-east-1:111122223333:instance/*",
                "arn:aws:ec2:us-west-1:111122223333: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.

Neste exemplo, você permite que uma entidade do crie uma função vinculada ao serviço se o nome de serviço for access-analyzer.amazonaws.com.

```
{
   "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.

```
{
      "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.

```
{
      "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.

```
{
      "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: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.

```
{
      "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: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 as informações do último acesso ao serviço para uma política do 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`.

Você pode 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 garantir que os usuários criem funções de serviço apenas pelos 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 de segurança 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 ter mais informações, consulte Controle de acesso a políticas.

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`.

Você pode 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 “Recursos de etiquetas do IAM”. Você pode 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.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": "iam:DeleteUser",
        "Resource": "*",
        "Condition": {"StringEquals": {"iam:ResourceTag/status": "terminated"}}
    }]
}
```

Chaves disponíveis para federação OIDC da AWS

Você pode 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 por workloads do Amazon Elastic Kubernetes Service.

Você pode usar chaves de contexto de condição do AWS OIDC para criar políticas que limitam o acesso de usuários federados 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.

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. Você pode usar essas chaves para controlar o

acesso a uma função. 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.

Chave da condição AWS STS	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.

Exemplos:

- `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`.

Campo `azp` 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`.

```
{
    "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"
                }
            }
        }
    ]
}
```

Campo `azp` 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`.

```
{
    "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"
                }
            }
        }
    ]
}
```

e-mail

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 exclusiva 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.

Exemplos:

- `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 a função à ramificação do GitHub chamada `demo`.

```
{
   "Version": "2012-10-17",
   "Statement": [
      "Condition": {
         "StringEquals": {
            "token.actions.githubusercontent.com:aud": "sts.amazonaws.com",
            "token.actions.githubusercontent.com:sub": "repo:octo-org/octo-
repo: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. Você pode usar essas chaves para controlar o acesso a uma função. 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.

Chave da condição AWS STS	declaração JWT do IdP	Disponível na sessão
amr	amr	Sim
aud	aud	Sim
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.

```
{
   "Version": "2012-10-17",
   "Statement": [
      "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. Você pode usar essas chaves para controlar o acesso a uma função. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

Chave da condição AWS STS	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. Você pode 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. Você pode usar essas chaves para controlar o acesso a uma função. Para fazer isso, compare as chaves de condição do AWS STS com os valores na coluna de declaração JWT do IdP.

Chave da condição AWS STS	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

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.

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 ter 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.

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. Você pode usar o token do portador para realizar as operações do AWS CodeArtifact. Para obter mais informações sobre tokens de portador, consulte Usar tokens de portador.

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.

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 essa chave para especificar a duração (em segundos) que um principal pode usar ao obter um token de portador 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. Você pode usar o token do portador para realizar as operações do AWS CodeArtifact. Para obter mais informações sobre tokens de portador, consulte Usar tokens de portador.

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. A chave não está presente para operações assume-role do AWS STS.

sts:ExternalId

Funciona com operadores de string.

Use essa chave para exigir que uma entidade de segurança forneça um identificador específico ao assumir uma função do IAM.

Disponibilidade: essa chave está presente na solicitação quando a entidade de segurança fornece um ID externo enquanto assume uma função 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 Como usar um ID externo ao conceder acesso aos seus recursos da AWS a 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 AWS IAM Identity Center.

Você pode 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. Por exemplo, você pode configurar o Amazon Redshift como uma aplicação do IAM Identity Center para acessar os recursos do Amazon S3 em nome da sua força de trabalho ou de identidades federadas.

A política de confiança do perfil a seguir permite que a entidade principal do serviço Amazon Redshift assuma um perfil na conta `111122223333`. Também concede permissão à entidade principal do serviço Amazon Redshift para definir chaves de contexto na solicitação, desde que o valor da chave de contexto `identitystore:UserId` definido seja `1111-22-3333-44-5555`. Depois que o perfil é assumido, a atividade aparece nos logs do AWS CloudTrail no elemento `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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "redshift.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:SetContext"
            ],
            "Condition": {
                "ForAllValues:ArnEquals": {
                    "sts:RequestContextProviders": [
                        "arn:aws:iam::aws:contextProvider/IdentityCenter"
                    ]
                },
                "StringEquals": {
                    "aws:SourceAccount": "111122223333",
                    "sts:RequestContext/identitystore:UserId":
 "1111-22-3333-44-5555"
                }
            }
        }
    ]
}
```

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.

```
      "Condition": {
        "ForAllValues:ArnEquals": {
          "sts:RequestContextProviders": [
            "arn:aws:iam::aws:contextProvider/IdentityCenter"
          ]
        }
      }
```

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 de segurança assume a função usando o AWS Management Console, qualquer comando de assumir função da CLI ou qualquer operação de API `AssumeRole` do AWS STS.

Você pode usar essa chave em uma política de confiança de função para exigir que os usuários forneçam um nome de sessão específico quando assumirem uma função. 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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RoleTrustPolicyRequireUsernameForSessionName",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Principal": {"AWS": "arn:aws:iam::111122223333:root"},
```

```
            "Condition": {
                "StringLike": {"sts:RoleSessionName": "${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 de segurança especifica ao assumir uma função com o valor que é especificado na política.

Disponibilidade:essa chave está presente na solicitação quando a entidade de segurança fornece uma identidade-fonte enquanto assume uma função usando qualquer comando de assumir função da CLI do AWS STS ou operação da API `AssumeRole` do AWS STS.

Você pode usar essa chave em uma política de confiança de função para exigir que seus usuários definam uma identidade-fonte específica ao assumir uma função. Por exemplo, você pode exigir que seu quadro de funcionários ou suas identidades federadas especifiquem um valor para a identidade-fonte. Você pode 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.

Você pode 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`.

```
{
    "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: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.

Você pode usar essa chave de condição em uma política para exigir a configuração de tags de sessão específicas como transitivas ao assumir uma função 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 (usuários, grupos de usuários e funções): 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.

- Gerenciar senhas de usuários na AWS: descreve as opções de gerenciamento de senhas para usuários do IAM em sua conta.
- Gerenciamento de chaves de acesso de 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.

- Uso de autenticação multifator (MFA) na AWS: 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 IAM: 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 elementos de política JSON do IAM: fornece descrições e exemplos de cada elemento da linguagem de política.
- Validação de políticas 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 Serviços da AWS 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 aplicativos da AWS.
- Centro de recursos de conceitos básicos — Saiba como configurar a Conta da AWS, participar da comunidade da AWS e lançar seu primeiro aplicativo.
- Tutoriais práticos — Siga os tutoriais passo a passo para iniciar seu primeiro aplicativo 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.
- AWS Support: a página web principal para obter informações sobre o AWS Support, um canal de suporte de resposta rápida e com atendimento individual para ajudar a construir e a executar aplicativos 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

Índice

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

> **Note**
>
> O AWS STS também dá suporte ao envio de solicitações para endpoints regionais além do endpoint global. 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 Version 4, acesse Processo de assinatura do Signature Version 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 obter mais informações, consulte as informações a seguir.

- Credenciais de segurança da AWS. 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
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.	26 de novembro de 2023

	Essa permissão é exigida pelo IAM Access Analyzer para executar verificações de política em suas políticas.	
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 privilégi o mínimo. O IAM Access Analyzer analisa continuam ente suas contas para identific ar 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 de 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 informações 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 Organizacional (UO) ou organização no AWS Organizations que contém os recursos. Você pode usar as chaves de condições globais `aws:ResourceAccount`, `aws:ResourceOrgID` e `aws:ResourceOrgPaths` 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	Adição de informações sobre como criar usuários administrativos em vez de usar credenciais de usuário raiz, remoção da prática recomendada de usar grupos de usuários para atribuir permissões a usuários do IAM e esclarecimento sobre 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
Atualizações de política gerenciada pela AWS: 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. Você pode 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 AWS Management Console, reduzindo interrupções 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. Você pode 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.	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
Atualização da página de login da AWS	Ao fazer login na página principal da AWS, 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 informações 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 você usar o Organizations para organizar suas contas em UOs, você poderá exigir que as entidades 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. Você pode usar o AWS Management Console, a AWS CLI e a API da AWS para exibir informações sobre quando uma função foi usada 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
Operação GetAccessKeyInfo do AWS STS	Você pode revisar as chaves de acesso da AWS em seu código para determinar se as chaves são de uma conta que você possui. Você pode transmitir um ID da chave de acesso usando o comando `aws sts get-access-key-info` da AWS CLI ou a operação de API `GetAccessKeyInfo` da AWS.	24 de julho de 2019

Visualização das informações de último acesso do serviço Organizations 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 em uma conta do Organizations tentaram acessar mais recentemente e quando. Você pode usar essa informação para identificar permissões desnecessárias, de forma que você possa refinar suas políticas do 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

Compatibilidade de regiões do AWS STS 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 AWS Management Console 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 consultor de acesso	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 AWS Management Console.	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
Processo de login na Conta da AWS	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 AWS Management Console.	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