aws

Guia do desenvolvedor

Amazon Cognito

Amazon Cognito: Guia do desenvolvedor

Table of Contents

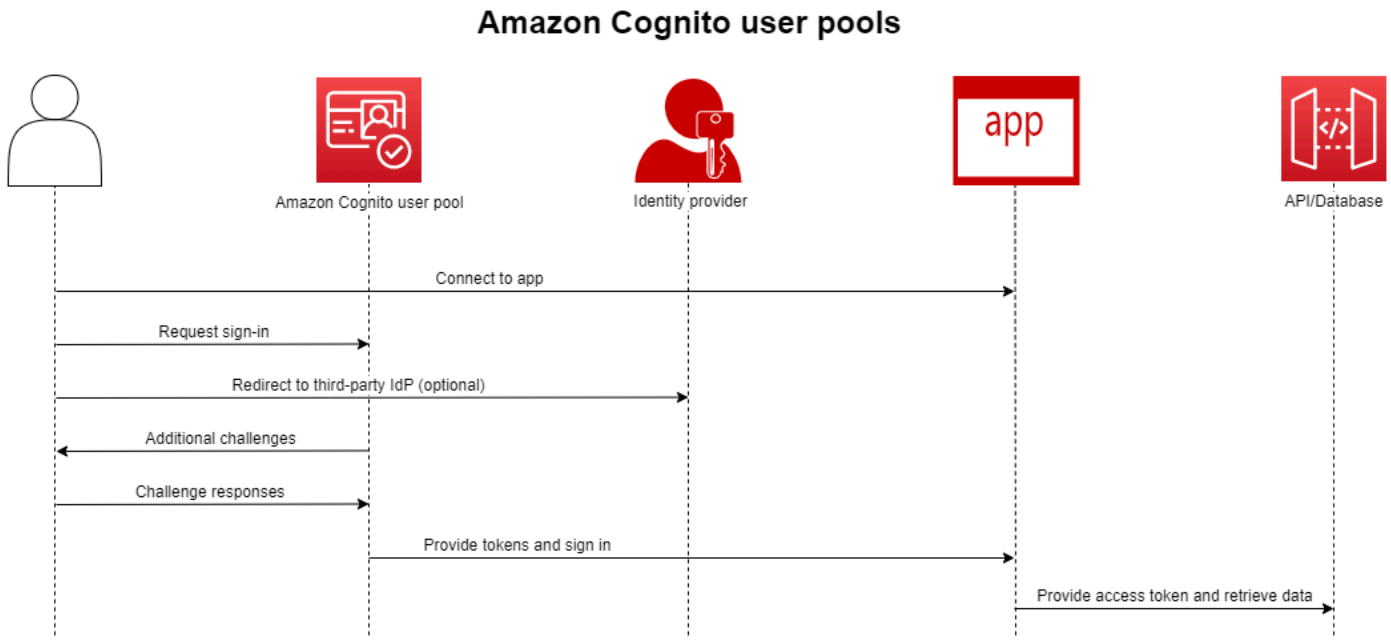
O que é o Amazon Cognito?

O Amazon Cognito é uma plataforma de identidade para aplicações web e aplicativos móveis. É um diretório de usuários, um servidor de autenticação e um serviço de autorização para tokens e AWS credenciais de acesso OAuth 2.0. Com o Amazon Cognito, você pode autenticar e autorizar usuários do diretório de usuários integrado, de seu diretório corporativo e de provedores de identidades de consumidores, como Google e Facebook.

Tópicos

Os dois componentes a seguir formam o Amazon Cognito. Eles operam de maneira independente ou em conjunto, com base nas necessidades de acesso dos usuários.

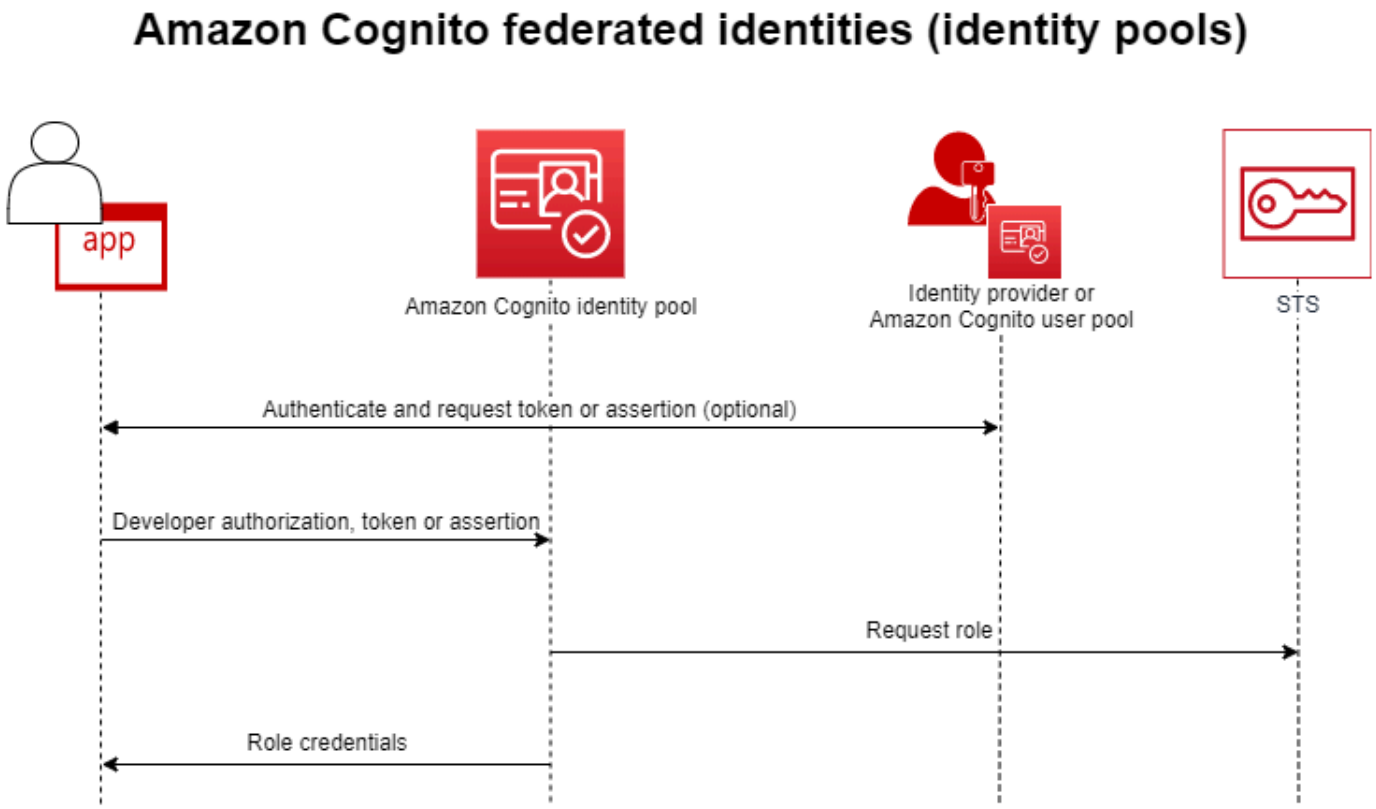
Grupos de usuários



Crie um grupo de usuários quando quiser autenticar e autorizar usuários em sua aplicação ou API. Os grupos de usuários são um diretório de usuários com criação, gerenciamento e autenticação de usuários por autoatendimento e orientados pelo administrador. O grupo de usuários pode ser um diretório independente e um provedor de identidades (IdP) OIDC e um provedor de serviços (SP) intermediário para provedores de terceiros de identidades de funcionários e clientes. Você pode fornecer login único (SSO) em seu aplicativo para as identidades da força de trabalho da sua organização no SAML 2.0 e no OIDC com grupos de usuários. IdPs Você também pode fornecer SSO em seu aplicativo para as identidades de clientes da sua organização nas lojas públicas de identidade OAuth 2.0 Amazon, Google, Apple e Facebook. Para obter mais informações sobre o gerenciamento de identidade e acesso de cliente (CIAM), consulte What is CIAM?.

Os grupos de usuários não exigem integração com um banco de identidades. Em um grupo de usuários, você pode emitir tokens web JSON autenticados (JWTs) diretamente para um aplicativo, um servidor web ou uma API.

Bancos de identidades



Configure um pool de identidade do Amazon Cognito quando quiser autorizar usuários autenticados ou anônimos a acessar seus recursos. AWS Um grupo de identidades emite AWS credenciais para que seu aplicativo forneça recursos aos usuários. Você pode autenticar usuários com um provedor de identidades confiável, como um grupo de usuários ou um serviço SAML 2.0. Ele também pode emitir credenciais para usuários convidados. Os grupos de identidades usam controle de acesso baseado em funções e atributos para gerenciar a autorização dos usuários para acessar seus recursos. AWS

Os bancos de identidades não exigem integração com um grupo de usuários. Um banco de identidades pode aceitar declarações autenticadas diretamente dos fornecedores de identidade de funcionários e consumidores.

Um grupo de usuários do Amazon Cognito e um banco de identidades usados juntos

No diagrama que inicia este tópico, você usa o Amazon Cognito para autenticar o usuário e, depois, conceder a ele acesso a um AWS service (Serviço da AWS).

1. O usuário do seu aplicativo faz login por meio de um grupo de usuários e recebe OAuth 2.0 tokens.
2. Seu aplicativo troca um token de grupo de usuários com um grupo de identidades por AWS credenciais temporárias que você pode usar com AWS APIs e o AWS Command Line Interface (AWS CLI).
3. Seu aplicativo atribui a sessão de credenciais ao seu usuário e fornece acesso autorizado ao Amazon S3 e ao Serviços da AWS Amazon DynamoDB.

Para ter mais exemplos que usam bancos de identidades e grupos de usuários, consulte Cenários comuns do Amazon Cognito.

No Amazon Cognito, a obrigação de segurança da nuvem do modelo de responsabilidade compartilhada está em conformidade com SOC 1 a 3, PCI DSS e ISO 27001 e é elegível para HIPAA-BAA. Você pode projetar sua segurança na nuvem no Amazon Cognito para ser compatível com SOC1 -3, ISO 27001 e HIPAA-BAA, mas não com o PCI DSS. Para mais informações, consulte Serviços da AWS no escopo. Consulta também Considerações sobre dados regionais.

Recursos do Amazon Cognito

Grupos de usuários

Um grupo de usuários do Amazon Cognito é um diretório de usuários. Com um grupo de usuários, os usuários podem fazer login na aplicação web ou no aplicativo móvel por meio do Amazon Cognito ou federar por meio de um IdP de terceiros. Os usuários federados e locais têm um perfil de usuário no grupo de usuários.

Os usuários locais são aqueles que se inscreveram ou que você criou diretamente no grupo de usuários. Você pode gerenciar e personalizar esses perfis de usuário no Console de gerenciamento da AWS, em um AWS SDK ou no AWS Command Line Interface (AWS CLI).

Os grupos de usuários do Amazon Cognito aceitam tokens e afirmações de terceiros IdPs e coletam os atributos do usuário em um JWT que ele emite para seu aplicativo. Você pode padronizar seu aplicativo em um conjunto de JWTs enquanto o Amazon Cognito lida IdPs com as interações, mapeando suas reivindicações em um formato de token central.

Um grupo de usuários do Amazon Cognito pode ser um IdP independente. O Amazon Cognito usa o padrão OpenID Connect (OIDC) para gerar autenticação e autorização. JWTs Quando você faz login

de usuários locais, o grupo de usuários é oficial para esses usuários. Você tem acesso aos recursos a seguir ao autenticar usuários locais.

- Implemente um front-end web próprio que chama a API de grupos de usuários do Amazon Cognito para autenticar, autorizar e gerenciar os usuários.
- Configure a autenticação multifator (MFA) para os usuários. O Amazon Cognito aceita senha de uso único com marcação temporal (TOTP) e MFA de mensagens SMS.
- Proteja-se contra o acesso de contas de usuários mal-intencionados que estão sob controle.
- Crie seus próprios fluxos personalizados de autenticação em várias etapas.
- Procure usuários em outro diretório e migre-os para o Amazon Cognito.

Um grupo de usuários do Amazon Cognito também pode desempenhar uma função dupla como provedor de serviços (SP) para o seu IdPs e um IdP para o seu aplicativo. Os grupos de usuários do Amazon Cognito podem se conectar ao consumidor, IdPs como o Facebook e o Google, ou à força de trabalho, IdPs como o Okta e o Active Directory Federation Services (ADFS).

Com os tokens OAuth 2.0 e OpenID Connect (OIDC) emitidos por um grupo de usuários do Amazon Cognito, você pode

- Aceitar um token de ID em sua aplicação que autentique um usuário e forneça as informações necessárias para configurar o perfil do usuário.
- Aceitar um token de acesso em sua API com os escopos do OIDC que autorizam chamadas de API dos usuários.
- Recupere AWS credenciais de um pool de identidade do Amazon Cognito.

Recursos de grupos de usuários do Amazon Cognito

Recurso	Description
Provedor de identidade OIDC	Emita tokens de ID para autenticar usuários
Servidor de autorização	Emita tokens de acesso para autorizar o acesso do usuário a APIs
Provedor de serviços SAML 2.0	Transforme declarações SAML em tokens de ID e acesso

Parte confiável do OIDC	Transforme tokens OIDC em tokens de ID e acesso
Parte confiável do provedor social	Transforme tokens de ID da Apple, Facebook, Amazon ou Google em seu próprio ID e tokens de acesso
Serviço de front-end de autenticação	Cadastre, gerencie e autentique usuários com login gerenciado
Suporte de API para sua própria interface	Crie, gerencie e autentique usuários por meio de solicitações de API de autenticação no suporte [1] AWS SDKs
Autenticação multifator	Use mensagens SMS ou o TOTPs dispositivo do seu usuário como um fator de autenticação adicional[1]
Monitoramento e resposta de segurança	Proteja-se contra atividades maliciosas e senhas inseguras[1]
Personalize fluxos de autenticação	Crie seu próprio mecanismo de autenticação ou adicione etapas personalizadas aos fluxos existentes[2]
Groups (Grupos)	Crie agrupamentos lógicos de usuários e uma hierarquia de declarações de função do IAM ao passar tokens para grupos de identidades
Personalize tokens	Personalize seu ID e tokens de acesso com reivindicações novas, modificadas e suprimidas
Personalize os atributos do usuário	Atribua valores aos atributos do usuário e adicione seus próprios atributos personalizados

[1] O recurso não está disponível para usuários federados.

[2] O recurso não está disponível para usuários federados e de login gerenciado.

Para mais informações sobre grupos de usuários, consulte Conceitos básicos dos grupos de usuários e a Referências da API de grupos de usuários do Amazon Cognito Sync.

Bancos de identidades

Um grupo de identidades é uma coleção de identificadores exclusivos, ou identidades, que você atribui aos seus usuários ou convidados e autoriza a receber credenciais temporárias. AWS Quando você apresenta a prova de autenticação para um grupo de identidades na forma de declarações confiáveis de um SAML 2.0, OpenID Connect (OIDC) ou provedor de identidade social (IdP) 2.0 OAuth , você associa seu usuário a uma identidade no grupo de identidades. O token que seu grupo de identidades cria para a identidade pode recuperar credenciais de sessão temporárias de AWS Security Token Service ()AWS STS.

Para complementar as identidades autenticadas, você também pode configurar um grupo de identidades para autorizar o acesso AWS sem a autenticação do IdP. Você pode oferecer uma prova de autenticação personalizada com Identidades autenticadas pelo desenvolvedor. Também pode conceder credenciais temporárias da AWS a usuários convidados, com identidades não autenticadas.

Com os bancos de identidades, você tem duas maneiras de se integrar às políticas do IAM em sua Conta da AWS. Você pode usar esses dois recursos juntos ou individualmente.

Controle de acesso com base em função

Quando o usuário transmite declarações ao banco de identidades, o Amazon Cognito escolhe o perfil do IAM que ele solicita. Para personalizar as permissões do perfil de acordo com suas necessidades, aplique as políticas do IAM a cada perfil. Por exemplo, se o usuário demonstrar que está no departamento de marketing, ele receberá credenciais para um perfil com políticas adaptadas às necessidades de acesso do departamento de marketing. O Amazon Cognito pode solicitar um perfil padrão, um perfil baseado em regras que consultam as declarações do usuário ou um perfil baseado na associação do usuário a um grupo de usuários. Você também pode configurar a política de confiança do perfil para que o IAM confie somente em seu banco de identidades para gerar sessões temporárias.

Atributos para controle de acesso

Seu banco de identidades lê os atributos das declarações do usuário e os correlaciona às tags de entidade principal na sessão temporária do usuário. Depois, você pode configurar as políticas baseadas em recursos do IAM para permitir ou negar acesso a recursos com base em entidades principais do IAM que carregam as tags de sessão do banco de identidades. Por exemplo,

se seu usuário demonstrar que está no departamento de marketing, marque AWS STS sua sessão`Department: marketing`. Seu bucket do Amazon S3 permite operações de leitura com base em uma PrincipalTag condição aws: que exige um valor de `marketing` para a tag. `Department`

Recursos de bancos de identidades do Amazon Cognito

Recurso	Description
Parte confiável do grupo de usuários do Amazon Cognito	Troque um token de ID do seu grupo de usuários por credenciais de identidade da web de AWS STS
Provedor de serviços SAML 2.0	Declarações SAML do Exchange para credenciais de identidade na web de AWS STS
Parte confiável do OIDC	Troque tokens OIDC por credenciais de identidade na web de AWS STS
Parte confiável do provedor social	Troque OAuth tokens da Amazon, Facebook, Google, Apple e Twitter por credenciais de identidade na web de AWS STS
Festa de confiança personalizada	Com AWS credenciais, troque reivindicações em qualquer formato por credenciais de identidade na web de AWS STS
Acesso não autenticado	Emita credenciais de identidade na web de acesso limitado sem autenticação AWS STS
Controle de acesso com base em função	Escolha uma função do IAM para seu usuário autenticado com base em suas reivindic ações e configure suas funções para serem assumidas somente no contexto do seu grupo de identidades
Controle de acesso por atributo	Converta declarações em tags principais para sua sessão AWS STS temporária e use políticas do IAM para filtrar o acesso a recursos com base nas tags principais

Para mais informações sobre grupos de identidades, consulte Conceitos básicos dos bancos de identidades do Amazon Cognito e a Referências da API de grupos de identidades do Amazon Cognito Sync.

Comparação entre grupos de usuários e bancos de identidades do Amazon Cognito

Recurso	Description	Grupos de usuários	Bancos de identidades
Provedor de identidade OIDC	Emita tokens de ID OIDC para autenticar usuários do aplicativo	✓	
Diretório de usuários	Armazene perfis de usuário para autenticação	✓	
Autorizar o acesso à API	Emita tokens de acesso para autorizar o acesso do usuário a APIs (incluindo operações de API de autoatendimento de perfil de usuário), bancos de dados e outros recursos que aceitam escopos OAuth	✓	
Autorização de identidade na web do IAM	Gere tokens que você pode trocar AWS STS por AWS credenciais temporárias		✓

Provedor de serviços SAML 2.0 e provedor de identidade OIDC	Emita tokens OIDC personalizados com base em declaraçõ es de um provedor de identidade SAML 2.0	✓	
Parte confiável do OIDC e provedor de identidade do OIDC	Emita tokens OIDC personalizados com base em declaraçõ es de um provedor de identidade OIDC	✓	
OAuth 2.0 parte confiável e provedor de identidade OIDC	Emita tokens OIDC personalizados com base nos escopos de provedores sociais OAuth 2.0, como Apple e Google	✓	
Provedor de serviços SAML 2.0 e corretor de credenciais	Emitir AWS credencia is temporárias com base em declaraçõ es de um provedor de identidade SAML 2.0		✓
Parte confiável e corretor de credencia is do OIDC	Emitir AWS credencia is temporárias com base em declaraçõ es de um provedor de identidade do OIDC		✓

Parte confiável do provedor social e corretor de credencia is	Emita AWS credencia is temporárias com base em tokens web JSON de aplicativos de desenvolvedores com provedores sociais como Apple e Google		✓
Parte confiável e agente de credenciais do grupo de usuários do Amazon Cognito	Emita AWS credencia is temporárias com base em tokens web JSON dos grupos de usuários do Amazon Cognito		✓
Parceiro confiável e corretor de credencia is personalizados	Emita AWS credencia is temporárias para identidades arbitrári as, autorizadas pelas credenciais do IAM do desenvolvedor		✓
Serviço de front-end de autenticação	Cadastre, gerencie e autentique usuários com login gerenciado	✓	
Suporte de API para sua própria interface de autenticação	Crie, gerencie e autentique usuários por meio de solicitaç ões de API no Supported [1] AWS SDKs	✓	

MFA	Use mensagens SMS ou o TOTPs dispositivo do seu usuário como um fator de autenticação adicional [1]	✓
Monitoramento e resposta de segurança	Proteja-se contra atividades maliciosas e senhas inseguras[1]	✓
Personalize fluxos de autenticação	Crie seu próprio mecanismo de autenticação ou adicione etapas personalizadas aos fluxos existentes[1]	✓
User groups (Grupos de usuários)	Crie agrupamentos lógicos de usuários e uma hierarquia de declarações de função do IAM ao passar tokens para grupos de identidades	✓
Personalize tokens	Personalize seu ID e tokens de acesso com reivindicações e escopos novos, modificados e suprimidos	✓
AWS WAF web ACLs	Monitore e controle as solicitações para seu front-end de autenticação com AWS WAF	✓

Personalize os atributos do usuário	Atribua valores aos atributos do usuário e adicione seus próprios atributos personalizados	✓	
Acesso não autenticado	Emita credenciais de identidade na web de acesso limitado sem autenticação AWS STS		✓
Controle de acesso com base em função	Escolha uma função do IAM para seu usuário autenticado com base em suas reivindicações e configure a confiança de sua função para limitar o acesso aos usuários de identidade da web		✓
Controle de acesso por atributo	Transforme as declarações do usuário em tags principais para sua sessão AWS STS temporária e use as políticas do IAM para filtrar o acesso aos recursos com base nas tags principais		✓

[1] O recurso não está disponível para usuários federados.

Conceitos básicos do Amazon Cognito

Veja exemplos de aplicações de grupos de usuários em Conceitos básicos dos grupos de usuários.

Para obter uma introdução aos bancos de identidades, consulte Conceitos básicos dos bancos de identidades do Amazon Cognito.

Para obter links para experiências de configuração guiada com grupos de usuários e bancos de identidades, consulte Opções de configuração guiada para o Amazon Cognito.

Para começar a usar um AWS SDK, consulte Ferramentas para AWS desenvolvedores. Para obter recursos para desenvolvedores específicos do Amazon Cognito, consulte os Recursos para desenvolvedores do Amazon Cognito.

Para usar o Amazon Cognito, você precisa de uma Conta da AWS. Para obter mais informações, consulte Começando com AWS.

Disponibilidade regional

O Amazon Cognito está disponível em várias AWS regiões em todo o mundo. Em cada região, o Amazon Cognito é distribuído em várias zonas de disponibilidade. Essas zonas de disponibilidade são fisicamente isoladas umas das outras, mas são unidas por conexões de rede privadas, de baixa latência, de alta taxa de transferência e altamente redundantes. Essas zonas de disponibilidade permitem AWS fornecer serviços, incluindo o Amazon Cognito, com níveis muito altos de disponibilidade e redundância, além de minimizar a latência.

Para ver se o Amazon Cognito está disponível atualmente em algum Região da AWS, consulte AWS Serviços por região.

Para saber mais sobre endpoints de serviços de API regionais, consulte Endpoints de serviço da AWS no Referência geral da Amazon Web Services.

Para saber mais sobre quantas zonas de disponibilidade estão disponíveis em cada região, consulte Infraestrutura global da AWS.

Preços do Amazon Cognito

Para informações sobre preços do Amazon Cognito, consulte preços do Amazon Cognito.

Termos e conceitos comuns do Amazon Cognito

O Amazon Cognito fornece credenciais para aplicações web e aplicativos móveis. Ele se baseia em termos que são comuns no gerenciamento de identidade e acesso. Há muitos guias disponíveis sobre termos universais de identidade e acesso. Alguns exemplos são:

- Terminologia no IDPro Body of Knowledge
- Serviços de identidade da AWS
- Glossário do NIST CSRC

As listas a seguir descrevem termos que são exclusivos do Amazon Cognito ou que têm um contexto específico no Amazon Cognito.

Tópicos

- Geral
- Grupos de usuários
- Bancos de identidades

Geral

Os termos desta lista não são específicos do Amazon Cognito. Eles são amplamente reconhecidos entre os profissionais de gerenciamento de identidade e acesso. A lista de termos a seguir não é completa, mas serve como guia para o contexto específico do Amazon Cognito aqui.

Token de acesso

: Um token web JSON (JWT) que contém informações sobre a autorização de uma entidade para acessar sistemas de informação.

App, aplicação

: Normalmente, um aplicativo móvel. Neste guia, o aplicativo geralmente é uma abreviação de um aplicativo web ou aplicativo móvel que se conecta ao Amazon Cognito.

Controle de acesso baseado em atributos (ABAC)

: Modelo em que um aplicativo determina o acesso aos recursos com base nas propriedades de um usuário, como seu cargo ou departamento. As ferramentas do Amazon Cognito para aplicar o

ABAC incluem tokens de ID em grupos de usuários e tags de entidades principais em bancos de identidades.

Autenticação

O processo de estabelecer uma identidade autêntica para fins de acesso a um sistema de informação. O Amazon Cognito aceita provas de autenticação de provedores de identidades de terceiros e também serve como provedor de autenticação para aplicações de software.

Autorização

Um processo de concessão de permissões para um recurso. Os tokens de acesso ao grupo de usuários contêm informações que as aplicações podem usar para permitir que usuários e sistemas acessem recursos.

Servidor de autorização

Um sistema OAuth ou OpenID Connect (OIDC) que gera tokens web JSON. O servidor de autorização gerenciado de grupos de usuários do Amazon Cognito é o componente do servidor de autorização dos dois métodos de autenticação e autorização nos grupos de usuários. Os grupos de usuários também são compatíveis com fluxos de resposta a desafios da API na autenticação do SDK.

Aplicação confidencial, aplicação do lado do servidor

Uma aplicação à qual os usuários se conectam remotamente, com código em um servidor de aplicações e acesso a segredos. Normalmente, uma aplicação web.

Identity provider (IdP) (Provedor de identidade (IdP))

Serviço que armazena e verifica as identidades dos usuários. O Amazon Cognito pode solicitar autenticação de fornecedores externos e ser um IdP para aplicações.

JSON web token (JWT)

Um documento formatado em JSON que contém declarações sobre um usuário autenticado. Os tokens de ID autenticam usuários, os tokens de acesso autorizam os usuários e os tokens de atualização atualizam as credenciais. O Amazon Cognito recebe tokens de fornecedores externos e emite tokens para aplicações ou AWS STS.

Autorização de máquina a máquina (M2M)

O processo de autorização de solicitações para endpoints de API para entidades de máquina que não interagem com o usuário, como um nível de aplicação de servidor web. Os grupos de

usuários fornecem autorização de M2M em concessões de credenciais de cliente com escopos do OAuth 2.0 em tokens de acesso.

Autenticação multifator (MFA)

A exigência de que os usuários forneçam autenticação adicional após informarem nome de usuário e senha. Os grupos de usuários do Amazon Cognito têm recursos de MFA para usuários locais.

Provedor OAuth 2.0 (social)

Um IdP para um grupo de usuários ou banco de identidades que fornece acesso ao JWT e aos tokens de atualização. Os grupos de usuários do Amazon Cognito automatizam as interações com provedores sociais após a autenticação dos usuários.

Provedor OpenID Connect (OIDC)

Um IdP para um grupo de usuários ou banco de identidades que estende à especificação OAuth para fornecer tokens de ID. Os grupos de usuários do Amazon Cognito automatizam as interações com provedores OIDC após a autenticação dos usuários.

Chave de acesso, WebAuthn

Uma forma de autenticação na qual as chaves criptográficas, ou chaves de acesso, no dispositivo de um usuário fornecem sua prova de autenticação. Os usuários verificam sua presença com mecanismos biométricos ou de código PIN em um autenticador de hardware ou software. As chaves de acesso são resistentes ao phishing e estão vinculadas a sites/aplicações específicos, oferecendo uma experiência segura sem senha. Os grupos de usuários do Amazon Cognito oferecem suporte ao login com chaves de acesso.

Sem senha

Uma forma de autenticação na qual o usuário não precisa digitar uma senha. Os métodos de login sem senha incluem senhas de uso único (OTPs) enviadas para endereços de e-mail, números de telefone e chaves de acesso. Os grupos de usuários do Amazon Cognito oferecem suporte ao login com OTPs e chaves de acesso.

Aplicativo público

Um aplicativo independente em um dispositivo, com código armazenado localmente e sem acesso a segredos. Normalmente, um aplicativo móvel.

Servidor de recursos

Uma API com controle de acesso. Os grupos de usuários do Amazon Cognito também usam o servidor de recursos para descrever o componente que define a configuração para interagir com uma API.

Regras de controle de acesso com base em função (RBAC)

Modelo que concede acesso com base na designação funcional do usuário. Os bancos de identidades do Amazon Cognito implementam o RBAC com diferenciação entre os perfis do IAM.

Provedor de serviço (SP), parte confiável (RP)

Aplicação que depende de um IdP para atestar que os usuários são confiáveis. O Amazon Cognito atua como SP para IdPs externos e como IdP para SPs baseados em aplicações.

Provedor SAML

IdP para um grupo de usuários ou banco de identidades que gera documentos de declaração assinados digitalmente que seu usuário passa para o Amazon Cognito.

Identificador exclusivo universal (UUID)

Um rótulo de 128 bits aplicado a um objeto. Os UUIDs do Amazon Cognito são exclusivos por grupo de usuários ou banco de identidades, mas não seguem um formato específico de UUID.

Diretório de usuários

Conjunto de usuários e seus atributos que fornece essas informações para outros sistemas. Os grupos de usuários do Amazon Cognito são diretórios e também ferramentas para consolidar usuários a partir de diretórios de usuários externos.

Grupos de usuários

Quando você encontrar os termos na lista a seguir deste guia, saiba que eles se referem a um recurso ou configuração específica dos grupos de usuários.

Autenticação adaptável

Um recurso de segurança avançada que detecta possíveis atividades mal-intencionadas e aplica segurança adicional aos perfis de usuário.

Cliente da aplicação

Componente que define as configurações de um grupo de usuários como um IdP para uma aplicação.

URL de retorno de chamada, URI de redirecionamento, URL de retorno

Uma configuração em um cliente de aplicação e um parâmetro nas solicitações ao servidor de autorização do grupo de usuários. O URL de retorno de chamada é o destino inicial dos usuários autenticados na aplicação.

Autenticação baseada em opções

Uma forma de autenticação de API com grupos de usuários em que cada usuário tem um conjunto de opções de login disponíveis. Suas opções podem incluir nome de usuário e senha com ou sem MFA, login com chave de acesso ou login sem senha com senhas de uso único enviadas por e-mail ou SMS. Sua aplicação pode moldar o processo de escolha dos usuários solicitando uma lista de opções de autenticação ou declarando uma opção preferencial.

Compare com a autenticação baseada em clientes.

Autenticação baseada em clientes

Uma forma de autenticação com a API de grupos de usuários e os backends de aplicações criados com os SDKs da AWS. Na autenticação declarativa, sua aplicação determina de forma independente o tipo de login que um usuário deve realizar e solicita esse tipo antecipadamente.

Compare com a autenticação baseada em opções.

Credenciais comprometidas

Um recurso de segurança avançada que detecta senhas de usuários que os invasores possam conhecer e aplica segurança adicional aos perfis de usuário.

Confirmação

Processo que determina que os pré-requisitos foram atendidos para permitir que um novo usuário faça login. A confirmação geralmente é feita por meio da verificação do endereço de e-mail ou número de telefone.

Autenticação personalizada

Uma extensão dos processos de autenticação com acionadores do Lambda que definem desafios e respostas adicionais do usuário.

Autenticação do dispositivo

Processo de autenticação que substitui o MFA por um login que usa o ID de um dispositivo confiável.

Domínio, domínio do grupo de usuários

Um domínio da web que hospeda suas páginas de login gerenciado na AWS. Você pode configurar o DNS em um domínio que possui ou usar um prefixo de subdomínio de identificação em um domínio que pertence à AWS.

Plano Essentials

O plano de recursos com os últimos desenvolvimentos em grupos de usuários. O plano Essentials não inclui os recursos de segurança de aprendizado automatizado presentes no plano Plus.

Fornecedor externo, fornecedor de terceiros

IdP que tem uma relação de confiança com um grupo de usuários. Os grupos de usuários servem como uma entidade intermediária entre provedores externos e sua aplicação, gerenciando processos de autenticação com SAML 2.0, OIDC e provedores sociais. Os grupos de usuários consolidam os resultados de autenticação de provedores externos em um único IdP para que suas aplicações possam processar muitos usuários usando uma única biblioteca de cliente OIDC.

Plano de recursos

O grupo de recursos que você pode selecionar para um grupo de usuários. Os planos de recursos têm custos diferentes em sua fatura da AWS. Novos grupos de usuários usam como padrão o plano Essentials.

Planos atuais

- Plano Lite
- Plano Essentials
- Plano Plus

Usuário federado, usuário externo

Usuário em um grupo de usuários que foi autenticado por um provedor externo.

IU hospedada (clássica), páginas de IU hospedada

A versão inicial dos serviços de frontend de autenticação, parte confiável e provedor de identidades no domínio do grupo de usuários. A IU hospedada tem um conjunto básico

de recursos e uma aparência simplificada. Você pode aplicar a identidade visual de IU hospedada com o upload de um arquivo de imagem de logotipo e um arquivo com um conjunto predeterminado de estilos CSS. Compare com o login gerenciado.

Gatilho do Lambda

Função no AWS Lambda que um grupo de usuários pode invocar automaticamente em pontos-chave nos processos de autenticação de usuários. Você pode usar os acionadores do Lambda para personalizar os resultados da autenticação.

Usuário local

Um perfil de usuário no diretório de usuários do grupo de usuários que não foi criado pela autenticação com um provedor externo.

Usuário vinculado

Usuário de um provedor externo cuja identidade é mesclada com a de um usuário local.

Plano Lite

O plano de recursos com os recursos lançados originalmente com grupos de usuários. O plano Lite não inclui os novos recursos do plano Essentials nem os recursos de segurança de aprendizado automatizado no plano Plus.

Servidor de autorização gerenciado, servidor de autorização de IU hospedada, servidor de autorização

Um componente do login gerenciado que hospeda serviços para interação com IdPs e aplicações no domínio do grupo de usuários. A IU hospedada difere do login gerenciado nos recursos interativos com o usuário que oferece, mas tem os mesmos recursos do servidor de autorização.

Login gerenciado, páginas de login gerenciado

Um conjunto de páginas da web no domínio do grupo de usuários que hospeda serviços para autenticação do usuário. Esses serviços incluem funções para operar como um IdP, uma parte confiável para IdPs de terceiros e um servidor de uma IU de autenticação interativa com o usuário. Quando você configura um domínio para o grupo de usuários, o Amazon Cognito coloca todas as páginas de login gerenciado online.

Sua aplicação importa bibliotecas do OIDC que invocam os navegadores dos usuários e os direcionam para a IU de login gerenciado para cadastro, login, gerenciamento de senhas e outras operações de autenticação. Após a autenticação, as bibliotecas do OIDC podem processar o resultado da solicitação de autenticação.

Autenticação de login gerenciado

O login com os serviços no domínio do grupo de usuários é feito por meio de páginas do navegador interativas com o usuário ou solicitações de API HTTPS. As aplicações lidam com a autenticação de login gerenciado com bibliotecas do OpenID Connect (OIDC). Esse processo inclui login com provedores externos, login de usuários locais com páginas de login gerenciado interativas e autorização M2M. A autenticação com a IU hospedada clássica também se enquadra nesse termo.

Compare com a autenticação do SDK da AWS.

Plano Plus

O plano de recursos com os últimos desenvolvimentos e recursos avançados de segurança em grupos de usuários.

Autenticação do SDK, autenticação do SDK da AWS

Um conjunto de operações de API de autenticação e autorização que você pode adicionar ao backend da sua aplicação com um SDK da AWS. Esse modelo de autenticação requer um mecanismo de login personalizado. A API pode cadastrar usuários locais usuários vinculados.

Compare com a autenticação de login gerenciado.

Proteção contra ameaças, recursos avançados de segurança

Nos grupos de usuários, a proteção contra ameaças se refere às tecnologias projetadas para mitigar ameaças aos seus mecanismos de autenticação e autorização. Autenticação adaptativa, detecção de credenciais comprometidas e listas de bloqueio de endereços IP são exemplos de proteção contra ameaças.

Personalização do token

O resultado de um acionador do Lambda que antecede a geração do token e que modifica o ID do usuário ou o token de acesso em tempo de execução.

Grupo de usuários, provedor de identidades do Amazon Cognito`cognito-idp`, grupos de usuários do Amazon Cognito

Um recurso da AWS com serviços de autenticação e autorização para aplicações que funcionam com IdPs do OIDC.

Verificação

Processo de confirmar que um usuário tem um endereço de e-mail ou número de telefone. Um grupo de usuários envia um código a um usuário que inseriu um novo endereço de e-mail ou número de telefone. Quando ele envia o código para o Amazon Cognito, verifica a propriedade do destino da mensagem e pode receber mensagens adicionais do grupo de usuários. Veja também confirmação.

Perfil de usuário, conta de usuário

Uma entrada para um usuário no diretório de usuários. Todos os usuários, incluindo aqueles de IdPs de terceiros, têm um perfil no grupo de usuários.

Bancos de identidades

Quando você encontrar os termos na lista a seguir deste guia, saiba que eles se referem a um recurso ou configuração específica dos bancos de identidades.

Atributos para controle de acesso

Uma implementação de controle de acesso por atributo em bancos de identidades. Os bancos de identidades aplicam atributos do usuário como tags às credenciais do usuário.

Autenticação básica (clássica)

Processo de autenticação em que você pode personalizar a solicitação de credenciais do usuário.

Identidades autenticadas pelo desenvolvedor

Processo de autenticação que autoriza as credenciais do usuário do banco de identidades com as credenciais do desenvolvedor.

Credenciais do desenvolvedor

As chaves de API do IAM de um administrador do banco de identidades.

Autenticação aprimorada

Um fluxo de autenticação que seleciona um perfil do IAM e aplica as tags de entidade principal de acordo com a lógica que você define no banco de identidades.

Identidade

Um UUID que vincula um usuário da aplicação e suas credenciais de usuário ao perfil em um diretório de usuários externo que tem uma relação de confiança com um banco de identidades.

Banco de identidades, identidades federadas do Amazon Cognito, identidade do Amazon Cognito, **`cognito-identity`**

Um recurso da AWS com serviços de autenticação e autorização para aplicações que usam credenciais temporárias da AWS.

Identidade não autenticada do

Um usuário que não fez login com um IdP do banco de identidades. Você pode permitir que os usuários gerem credenciais de usuário limitadas para um único perfil do IAM antes da autenticação.

Credenciais do usuário

Chaves de API temporárias da AWS que os usuários recebem após a autenticação no banco de identidades.

Começando com AWS

Antes de começar a trabalhar com o Amazon Cognito, prepare-se com alguns recursos necessários AWS . Se você já consegue fazer login em um Conta da AWS, você pode pular esta seção. Continue lendo se estiver procurando informações sobre como se inscrever e fazer login com AWS credenciais. Depois de ter credenciais com permissões suficientes AWS Identity and Access Management (IAM), você pode começar a usar grupos de usuários e grupos de identidades.

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra a https://portal.aws.amazon.com/billing/inscrição.
2. Siga as instruções online.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo teclado do telefone.

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

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, você pode visualizar a atividade atual da sua conta e gerenciar sua conta acessando https://aws.amazon.com/e escolhendo Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS Centro de Identidade do AWS IAM, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login Console de gerenciamento da AWScomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, insira a senha.

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

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

Criar um usuário com acesso administrativo

1. Habilita o Centro de Identidade do IAM.

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

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

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com o seu usuário do Centro de Identidade do IAM, use o URL de login enviado ao seu endereço de e-mail quando o usuário do Centro de Identidade do IAM foi criado.

Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a 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 Centro de Identidade do AWS IAM .
2. Atribua usuários a um grupo e, em seguida, atribua o acesso de logon único ao grupo.

   Para obter instruções, consulte Adicionar grupos no Guia do usuário do Centro de Identidade do AWS IAM .

Conceitos básicos dos grupos de usuários

Você tem uma aplicação que requer autenticação e controle de acesso. Você pode trabalhar na estrutura OpenID Connect (OIDC) para autenticação única (SSO). O Amazon Cognito tem ferramentas para lidar com a lógica de autenticação no back-end do aplicativo com um AWS SDK e para invocar um navegador em seu cliente para acessar um servidor de autorização gerenciado.

O console do Amazon Cognito orienta você na criação de um grupo de usuários com base na visualização da sua estrutura de aplicação preferencial. A partir daí, você pode continuar adicionando recursos como login federado com redes sociais externas ou provedores de identidade SAML 2.0 (). IdPs Os modelos de aplicações no console do Amazon Cognito se baseiam na adição de bibliotecas do OIDC ao seu projeto e na invocação de um navegador.

À medida que você expande o conjunto de recursos e incorpora mais componentes do Amazon Cognito, leia o capítulo Grupos de usuários do Amazon Cognito para obter descrições completas de tudo o que você pode fazer com os grupos de usuários.

Os exemplos neste capítulo e no console do Amazon Cognito demonstram uma integração básica dos recursos de aplicação com os grupos de usuários do Amazon Cognito. Posteriormente, você pode ajustar o grupo de usuários para usar mais opções disponíveis. Em seguida, você pode atualizar seu aplicativo para adotar novos recursos e interagir com IdPs.

Se você não quiser usar as páginas de login gerenciadas, poderá criar um aplicativo com interfaces de autenticação personalizadas usando um AWS SDK ou. AWS Amplify As aplicações que você cria dessa forma interagem com a API de grupos de usuários e são adequadas somente para autenticar usuários locais. Continue aprendendo sobre esse modelo de autenticação em Outras opções de aplicações.

Tópicos

- Criar uma aplicação no console do Amazon Cognito
- Outras opções de aplicações
- Adicione mais recursos e opções de segurança ao grupo de usuários

Criar uma aplicação no console do Amazon Cognito

Os grupos de usuários adicionam opções de autenticação às aplicações de software. Para uma experiência inicial mais fácil, acesse o console do Amazon Cognito e siga as instruções. O processo

de criação orienta você não somente na configuração dos recursos do grupo de usuários, mas também na configuração das partes iniciais da sua aplicação.

Quando estiver tudo pronto para começar, navegue até o console do Amazon Cognito e clique no botão para criar um grupo de usuários. O processo de configuração guiará você pelas opções de configuração e linguagem de programação.

Recursos adicionais para conceitos de autenticação

- Autenticação com grupos de usuários do Amazon Cognito
- Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado
- Como funciona a autenticação com o Amazon Cognito
- Integração da autenticação e autorização do Amazon Cognito com aplicações móveis e da web

Como criar recursos do Amazon Cognito para sua aplicação

1. Acesse o console do Amazon Cognito. Para atribuir permissões à sua entidade principal do IAM para que ela possa criar e gerenciar recursos do Amazon Cognito, consulte AWS políticas gerenciadas para o Amazon Cognito. A política `AmazonCognitoPowerUser` é suficiente para a criação de grupos de usuários.
2. Selecione Criar grupo de usuários no menu Grupos de usuários ou selecione Começar gratuitamente em menos de 5 minutos.
3. Em Definir sua aplicação, escolha o tipo de aplicação que melhor se adequa ao cenário para o qual você deseja criar os serviços de autenticação e autorização.
4. Em Dê um nome para sua aplicação, insira um nome descritivo ou mantenha o nome padrão.
5. Você deve fazer algumas escolhas básicas em Configurar opções compatíveis com configurações que não podem ser alteradas após a criação do grupo de usuários.

    a. Em Opções para identificadores de login, informe como você deseja identificar os usuários quando eles fizerem login. Você pode preferir nomes de usuário gerados pelo usuário, endereços de e-mail ou números de telefone. Você também pode permitir uma combinação de várias opções. O Amazon Cognito aceita as opções que você configurar no campo de nome de usuário dos formulários de login gerenciado.

    b. Em Atributos obrigatórios para a inscrição, informe quais informações do usuário você deseja coletar quando os usuários se registram para uma nova conta. Nas páginas de login gerenciado, o Amazon Cognito apresenta solicitações para todos os atributos obrigatórios.

As Opções para identificadores de login influenciam seus atributos obrigatórios. O Nome de usuário requer atributos de e-mail ou telefone para que cada usuário possa receber um código de redefinição de senha em um e-mail ou SMS. O E-mail exige o atributo de e-mail e o Número de telefone requer o atributo de número de telefone.

6. Em Adicionar um URL de retorno, insira um caminho de redirecionamento para sua aplicação após a conclusão da autenticação dos usuários. Esse local deve ser uma rota em sua aplicação que usa bibliotecas do OpenID Connect (OIDC) para processar os resultados da autenticação do usuário. Um exemplo de URL de retorno para uma aplicação de teste é `https://localhost:3000/callback`. Na aplicação NodeJS de exemplo no console do Amazon Cognito, essa rota utiliza openid-client para coletar o token de acesso e resgatá-lo para obter informações do usuário. Você poderá navegar pelos exemplos para sua plataforma de desenvolvimento após criar seus recursos.
7. Selecione Criar sua aplicação. O Amazon Cognito cria um grupo de usuários e um cliente da aplicação com configurações padrão para seu tipo de aplicação. Você pode configurar opções adicionais, como provedores de identidades externos e autenticação multifator (MFA), após criar seus recursos iniciais.
8. Na página Configurar sua aplicação, você pode obter imediatamente exemplos de código para sua aplicação. Para explorar o novo grupo de usuários, role para baixo e selecione Ir para visão geral.
9. Para adicionar mais aplicações no mesmo grupo de usuários, navegue até o menu Clientes da aplicação e adicione um novo cliente da aplicação. Isso repetirá o processo de criação com foco na aplicação, mas adicionará somente um novo cliente da aplicação ao grupo de usuários existente.

Após criar um grupo de usuários e um ou mais clientes da aplicação com esse processo, você pode começar a testar as operações de autenticação com o login gerenciado. Essas opções de início rápido estão disponíveis para cadastro público. Recomendamos que você crie um ambiente de teste com o processo do console e, em seguida, mova o design finalizado para a produção. Dedique um tempo para se familiarizar com os recursos do Amazon Cognito. Em seguida, para migrar para cargas de trabalho de produção, crie configurações personalizadas e implante-as com ferramentas de automação como AWS CloudFormation e a. AWS Cloud Development Kit (AWS CDK)

O Amazon Cognito faz algumas configurações padrão nesse processo que você não pode reverter. Para obter mais informações sobre as configurações do grupo de usuários que você não pode alterar

e as opções que você pode escolher no console, consulte Como atualizar a configuração do grupo de usuários e do cliente da aplicação.

Configuração	Efeito	Como alterar	Mais informações
Segredo do cliente	Requer um hash do segredo do cliente nas solicitações de autenticação.	Crie um novo cliente de aplicativo com um aplicativo web tradicional ou um perfil de Machine-to-machine aplicativo.	Configurações específicas da aplicação com clientes de aplicação
Nome de usuário preferido	O grupo de usuários não aceita o atributo `preferred_username` como um alias.	Crie um grupo de usuários programaticamente com um AWS SDK.	Personalização dos atributos de login
Diferenciação de letras maiúsculas e minúsculas	Os nomes de usuário do grupo de usuários não diferenciam maiúsculas de minúsculas, por exemplo, `JohnD` é considerado o mesmo usuário que `johnd`.	Crie um grupo de usuários programaticamente com um AWS SDK.	Sensibilidade entre maiúsculas e minúsculas do grupo de usuários

Outras opções de aplicações

Você pode ter uma IU de aplicação existente que deseja integrar à autenticação do Amazon Cognito. Você pode até ter suas próprias páginas de autenticação existentes com uma configuração de diretório menos funcional do que os grupos de usuários do Amazon Cognito. Você pode adicionar ou substituir um componente de autenticação em um aplicativo desse tipo com integrações do Amazon Cognito AWS SDKs para uma variedade de linguagens de programação. Estes são alguns exemplos.

Se você criar um grupo de usuários para essa finalidade no console do Amazon Cognito, poderá não ser necessário ter um domínio do grupo de usuários que hospede páginas de login interativas e serviços OpenID Connect (OIDC). O processo de criação do grupo de usuários no console gera automaticamente um domínio para você. É possível excluir esse domínio na guia Domínio do grupo de usuários. Outras opções incluem a criação programática de recursos do Amazon Cognito para seu aplicativo com solicitações de API e com as opções de configuração automatizada AWS SDKs na CLI. AWS Amplify Para obter mais informações, consulte Integração da autenticação e autorização do Amazon Cognito com aplicações móveis e da web.

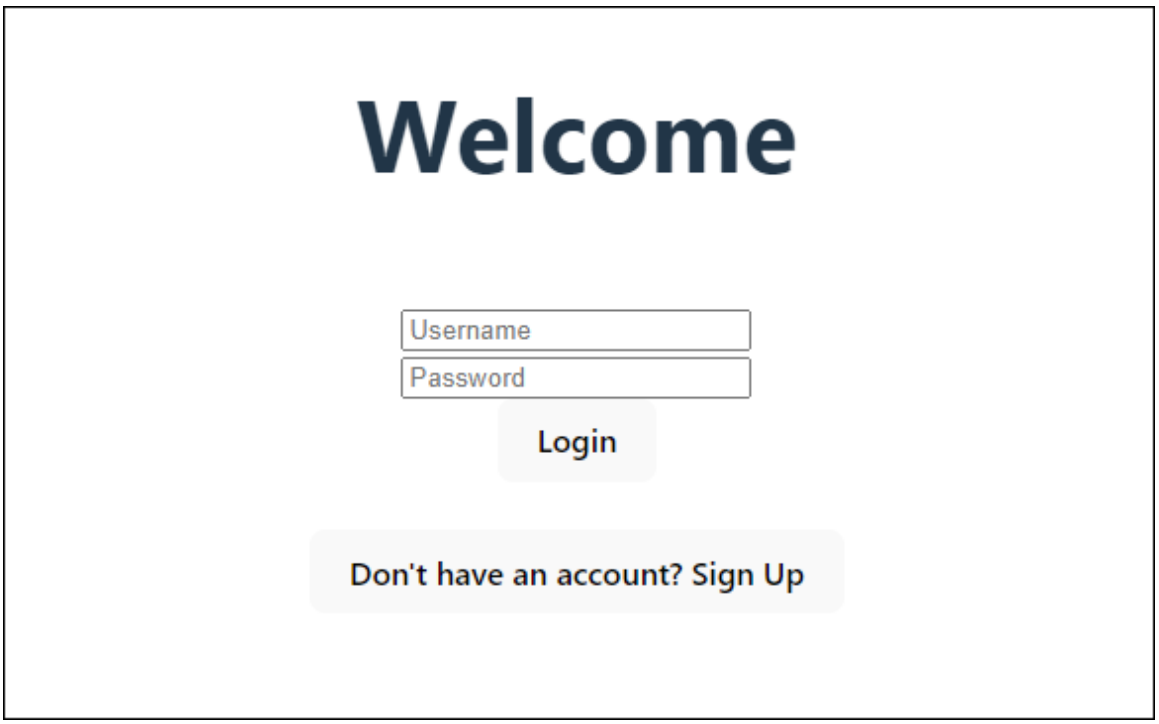
Tópicos

- Configurar um exemplo de aplicação de página única do React
- Configurar uma aplicação Android de exemplo com o Flutter

Configurar um exemplo de aplicação de página única do React

Neste tutorial, você criará uma aplicação de página única do React para testar a inscrição, a confirmação e o login do usuário. O React é uma biblioteca JavaScript baseada em aplicativos web e móveis, com foco na interface do usuário (UI). Este exemplo de aplicação demonstra algumas funções básicas dos grupos de usuários do Amazon Cognito. Se você já tem experiência em desenvolvimento de aplicativos web com o React, baixe o aplicativo de exemplo em GitHub.

A captura de tela a seguir é da página de autenticação inicial na aplicação que você vai criar.



Para configurar essa aplicação, o grupo de usuários deve atender aos seguintes requisitos:

- Os usuários podem fazer login com o endereço de e-mail. Opções de login do grupo de usuários do Cognito: E-mail.
- Os nomes de usuário não diferenciam maiúsculas de minúsculas. Requisitos de nome de usuário: a opção Tornar o nome de usuário sensível a maiúsculas e minúsculas não está selecionada.
- A autenticação multifator (MFA) não é necessária. Aplicação da MFA: a MFA é opcional.
- Seu grupo de usuários verifica os atributos para confirmação do perfil de usuário com uma mensagem de e-mail. Atributos a serem verificados: enviar mensagem de e-mail, verificar endereço de e-mail.
- E-mail é o único atributo obrigatório. Atributos obrigatórios: e-mail.
- Os usuários podem se cadastrar no seu grupo de usuários. Autorregistro: a opção Habilitar autorregistro está selecionada.
- Seu cliente de aplicação inicial é público e permite o login com nome de usuário e senha. Tipo de aplicação: Cliente público, Fluxos de autenticação: `ALLOW_USER_PASSWORD_AUTH`.

Criar uma aplicação do

Para criar essa aplicação, configure um ambiente de desenvolvedor. Os requisitos do ambiente do desenvolvedor são os seguintes:

1. O Node.js está instalado e atualizado.
2. O gerenciador de pacotes Node (npm) está instalado e atualizado pelo menos para a versão 10.2.3.
3. O ambiente pode ser acessado pela porta TCP 5173 em um navegador da Web.

Para criar uma aplicação web do React

1. Faça login em seu ambiente de desenvolvedor e procure o diretório principal da aplicação.

```
cd ~/path/to/project/folder/
```

2. Crie um serviço do React.

```
npm create vite@latest frontend-client -- --template react-ts
```

3. Clone a pasta do `cognito-developer-guide-react-example` projeto a partir do repositório de exemplos de AWS código em. GitHub

```
cd ~/some/other/path
```

```
git clone https://github.com/awsdocs/aws-doc-sdk-examples.git
```

```
cp -r ./aws-doc-sdk-examples/javascriptv3/example_code/cognito-identity-provider/
scenarios/cognito-developer-guide-react-example/frontend-client ~/path/to/project/
folder/
```

4. Procure o diretório `src` do seu projeto.

```
cd ~/path/to/project/folder/frontend-client/src
```

5. Edite `config.json` e substitua os valores a seguir:

   a. `YOUR_AWS_REGION`Substitua por um Região da AWS código. Por exemplo: `us-east-1`.

   b. Substitua `YOUR_COGNITO_USER_POOL_ID` pelo ID do grupo de usuários que você designou para teste. Por exemplo: `us-east-1_EXAMPLE`. O grupo de usuários deve estar no Região da AWS que você inseriu na etapa anterior.

c. Substitua `YOUR_COGNITO_APP_CLIENT_ID` pelo ID do cliente de aplicação que você designou para teste. Por exemplo: `1example23456789`. O cliente da aplicação deve estar no grupo de usuários da etapa anterior.

6. Se quiser acessar a aplicação de exemplo a partir de um IP diferente de `localhost`, edite `package.json` e altere a linha `"dev": "vite",` para `"dev": "vite --host 0.0.0.0",`.
7. Instale sua aplicação.

```
npm install
```

8. Inicie a aplicação.

```
npm run dev
```

9. Acesse a aplicação usando um navegador da Web em `http://localhost:5173` ou `http://[IP address]:5173`.
10. Cadastre um novo usuário com endereço de e-mail válido.
11. Recupere o código de confirmação da sua mensagem de e-mail. Insira o código de confirmação na aplicação.
12. Faça login com seu nome de usuário e senha.

Criação de um ambiente de desenvolvedor React com o Amazon Lightsail

Uma maneira rápida de começar a usar esse aplicativo é criar um servidor virtual na nuvem com o Amazon Lightsail.

Com o Lightsail, você pode criar rapidamente uma pequena instância de servidor que vem pré-configurada com os pré-requisitos para esse aplicativo de exemplo. Você pode usar SSH para sua instância com um cliente baseado em navegador e se conectar ao servidor web em um endereço IP público ou privado.

Para criar uma instância do Lightsail para esse aplicativo de exemplo

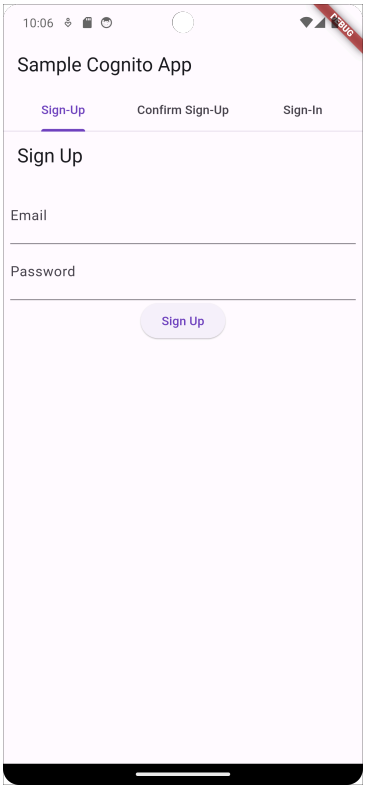
1. Acesse o console do Lightsail. Se solicitado, insira suas AWS credenciais.
2. Selecione Criar instância.
3. Em Selecionar uma plataforma, escolha Linux/Unix.
4. Em Selecionar esquema, selecione Node.js.

5. Em Identificar sua instância, dê um nome amigável ao seu ambiente de desenvolvimento.
6. Selecione Criar instância.
7. Depois que o Lightsail criar sua instância, selecione-a e, na guia Connect, escolha Connect using SSH.
8. Uma sessão SSH é aberta em uma janela do navegador. Execute `node -v` e `npm -v` para confirmar se a instância foi provisionada com Node.js e a versão mínima de npm 10.2.3.
9. Vá para configuração da aplicação React.

Configurar uma aplicação Android de exemplo com o Flutter

Neste tutorial, você criará uma aplicação móvel no Android Studio para emular um dispositivo e testar o cadastro, a confirmação e o login do usuário. Esta aplicação de exemplo cria um cliente móvel básico de grupos de usuários do Amazon Cognito para Android no Flutter. Se você já tem experiência em desenvolvimento de aplicativos móveis com o Flutter, baixe o aplicativo de exemplo em. GitHub

A captura de tela a seguir mostra a aplicação em execução em um dispositivo Android virtual.



Para configurar essa aplicação, o grupo de usuários deve atender aos seguintes requisitos:

- Os usuários podem fazer login com o endereço de e-mail. Opções de login do grupo de usuários do Cognito: E-mail.
- Os nomes de usuário não diferenciam maiúsculas de minúsculas. Requisitos de nome de usuário: a opção Tornar o nome de usuário sensível a maiúsculas e minúsculas não está selecionada.
- A autenticação multifator (MFA) não é necessária. Aplicação da MFA: a MFA é opcional.
- Seu grupo de usuários verifica os atributos para confirmação do perfil de usuário com uma mensagem de e-mail. Atributos a serem verificados: enviar mensagem de e-mail, verificar endereço de e-mail.
- E-mail é o único atributo obrigatório. Atributos obrigatórios: e-mail.
- Os usuários podem se cadastrar no seu grupo de usuários. Autorregistro: a opção Habilitar autorregistro está selecionada.
- Seu cliente de aplicação inicial é público e permite o login com nome de usuário e senha. Tipo de aplicação: Cliente público, Fluxos de autenticação: `ALLOW_USER_PASSWORD_AUTH`.

Criar uma aplicação do

Para criar uma aplicação Android de exemplo

1. Instale o Android Studio e as ferramentas de linha de comando.
2. No Android Studio, instale o plug-in do Flutter.
3. Crie um projeto do Android Studio a partir do conteúdo do diretório `cognito_flutter_mobile_app` nesta aplicação de exemplo.
   - Edite `assets/config.json` e substitua `<<YOUR USER POOL ID>>` e `<< YOUR CLIENT ID>>` com o IDs de seu grupo de usuários e cliente de aplicativo.
4. Instale o Flutter.
   a. Adicione o Flutter à sua variável PATH.
   b. Aceite as licenças com o comando a seguir.

      ```
      flutter doctor --android-licenses
      ```

   c. Verifique seu ambiente do Flutter e instale os componentes que faltam.

      ```
      flutter doctor
      ```

- Se algum componente estiver faltando, execute `flutter doctor -v` para saber como corrigir o problema.

d. Vá para o diretório do seu novo projeto do Flutter e instale as dependências.

- Executar `flutter pub add amazon_cognito_identity_dart_2`.

e. Executar `flutter pub add flutter_secure_storage`.

5. Crie um dispositivo virtual do Android.

   1. Na interface do usuário do Android Studio, crie um dispositivo com o gerenciador de dispositivos.
   2. Na CLI, execute `flutter emulators --create --name android-device`.

6. Inicie seu dispositivo Android virtual.

   1. Na interface do usuário do Android Studio, selecione o ícone

      de início ao lado do seu dispositivo virtual.
   2. Na CLI, execute `flutter emulators --launch android-device`.

7. Inicie a aplicação no dispositivo virtual.

   1. Na interface do usuário do Android Studio, selecione o ícone

      de implantação.
   2. Na CLI, execute `flutter run`.

8. Navegue até seu dispositivo virtual em execução no Android Studio.
9. Cadastre um novo usuário com endereço de e-mail válido.
10. Recupere o código de confirmação da sua mensagem de e-mail. Insira o código de confirmação na aplicação.
11. Faça login com seu nome de usuário e senha.

Adicione mais recursos e opções de segurança ao grupo de usuários

Depois de seguir os tutoriais para criar exemplos de aplicações, você pode ampliar o escopo da implementação do seu grupo de usuários. Ou, se você não criou uma aplicação de teste, crie um grupo de usuários de acordo com suas preferências. Você pode personalizar os recursos do grupo de usuários para outras aplicações ou adicionar provedores de identidades externos. Ao planejar a migração para colocar grupos de usuários do Amazon Cognito em aplicações de produção, você pode avaliar exemplos e tutoriais adicionais.

Se sua próxima prioridade for examinar e aplicar opções de segurança de aplicações nos grupos de usuários, consulte Praticas recomendadas de segurança para grupos de usuários do Amazon Cognito.

O Amazon Cognito tem planos de recursos que adicionam opções funcionais e de segurança quando você opta por níveis mais altos. Você pode começar com o plano Lite, adicionar opções avançadas de autenticação e autorização com o plano Essentials e adicionar barreiras de proteção de raciocínio automatizado com o plano Plus. Para obter mais informações, consulte Planos de recursos de grupos de usuários.

A seguir estão alguns recursos adicionais de grupos de usuários do Amazon Cognito:

- Aplicar a identidade visual às páginas de login gerenciado
- Adicionar MFA a um grupo de usuários
- Segurança avançada com proteção contra ameaças
- Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda
- Como usar o Amazon Pinpoint para análise de grupos de usuários

Para ter uma visão geral dos modelos de autenticação e autorização do Amazon Cognito, consulte Como funciona a autenticação com o Amazon Cognito.

Para acessar outro Serviços da AWS após uma autenticação bem-sucedida do grupo de usuários, consulteAcessando Serviços da AWS usando um pool de identidades após o login.

Além de usar o Console de gerenciamento da AWS e o grupo de usuários SDKs, você também pode gerenciar seus grupos de usuários usando AWS Command Line Interfaceo.

Tópicos

Adicionar login social ao seu grupo de usuários

Possibilitar que os usuários façam login na aplicação por meio de seus provedores de identidade públicos ou sociais existentes pode melhorar sua experiência de autenticação. Os grupos de usuários do Amazon Cognito se integram a provedores de identidade social (IdPs) populares, como Facebook, Google, Amazon e Apple, oferecendo aos usuários opções de login convenientes com as quais eles já estão familiarizados.

Ao configurar o login social, você oferece aos usuários uma alternativa à criação de uma conta dedicada apenas para sua aplicação. Isso pode melhorar as taxas de conversão e simplificar o processo de cadastro. Do ponto de vista do usuário, ele pode aplicar suas credenciais sociais existentes para se autenticar rapidamente, sem o atrito de lembrar outro nome de usuário e senha.

Configurar um IdP social em seu grupo de usuários envolve algumas etapas importantes. Você deve registrar a aplicação no provedor social para obter um ID de cliente e um segredo. Em seguida, você pode adicionar a configuração do IdP social ao seu grupo de usuários, especificando os escopos que deseja solicitar e os atributos do grupo de usuários que deseja mapear a partir dos atributos do IdP. No tempo de execução, o Amazon Cognito gerencia a troca de tokens com o provedor, mapeia os atributos do usuário e emite tokens para sua aplicação no formato de grupo de usuários compartilhado.

Inscrever-se com um IdP social

Antes de criar um IdP social com o Amazon Cognito, é necessário registrar sua aplicação no IdP social para receber um ID do cliente e a chave secreta do cliente.

Para registrar um aplicativo com o Facebook

1. Crie uma conta de desenvolvedor com o Facebook.
2. Faça login com as credenciais do Facebook.
3. No menu My Apps (Meus aplicativos), escolha Create New App (Criar novo aplicativo).

   Se você não tiver uma aplicação do Facebook, verá uma opção diferente. Escolha Criar aplicativo.

4. Na página Criar uma aplicação, selecione um caso de uso para a aplicação e escolha Próximo.
5. Forneça um nome para a aplicação do Facebook e escolha Criar ID da aplicação.
6. Na barra de navegação à esquerda, selecione Configurações da aplicação e, depois, selecione Básico.
7. Anote o App ID (ID do aplicativo) e a App Secret (Chave secreta do aplicativo). Você poderá usá-los na próxima seção.
8. Escolha + Adicionar plataforma na parte inferior da página.
9. Na tela Selecionar plataforma, selecione as plataformas e escolha Próximo.
10. Escolha Salvar alterações.
11. Para App Domains (Domínios da aplicação), insira o domínio do grupo de usuários.

    ```
    https://your_user_pool_domain
    ```
12. Escolha Salvar alterações.
13. Na barra de navegação, selecione Produtos e, depois, Configurar em Login com Facebook.
14. No menu Configurar de Login com Facebook, selecione Configurações.

    Insira seu URL de redirecionamento em OAuth URIsRedirecionamento válido. O URL de redirecionamento consiste no domínio do grupo de usuários com o endpoint `/oauth2/idpresponse`.

    ```
    https://your_user_pool_domain/oauth2/idpresponse
    ```
15. Escolha Salvar alterações.

Para registrar um aplicativo com a Amazon

1. Crie uma conta de desenvolvedor com a Amazon.
2. Faça login com as credenciais da Amazon.
3. Você precisa criar um perfil de segurança da Amazon para receber o ID do cliente e a chave secreta do cliente da Amazon.

   Selecione Aplicativos e serviços, na barra de navegação na parte superior da página, depois, selecione Login with Amazon.
4. Escolha Create a Security Profile (Criar um perfil de segurança).

5. Insira o Security Profile Name (Nome do perfil de segurança), Security Profile Description (Descrição do perfil de segurança) e um Consent Privacy Notice URL (URL de notificação de consentimento de privacidade).
6. Escolha Save (Salvar).
7. Selecione Client ID (ID de cliente) e Client Secret (Segredo de cliente) para mostrar o ID e o segredo do cliente. Você poderá usá-los na próxima seção.
8. Passe o cursor sobre o ícone de engrenagem e escolha Web Settings (Configurações da Web) e, em seguida, escolha Edit (Editar).
9. Insira o domínio do grupo de usuários em Allowed Origins (Origens permitidas).

```
https://<your-user-pool-domain>
```

10. Insira seu domínio do grupo de usuários com o `/oauth2/idpresponse` endpoint em Allowed Return URLs.

```
https://<your-user-pool-domain>/oauth2/idpresponse
```

11. Escolha Salvar.

Para registrar um aplicativo com o Google

Para mais informações sobre OAuth 2.0 na plataforma Google Cloud, consulte Saiba mais sobre autenticação e autorização na documentação do Google Workspace for Developers.

1. Crie uma conta de desenvolvedor com o Google.
2. Faça login no Console do Google Cloud Platform.
3. Na barra de navegação superior, escolha Select a project (Selecionar um projeto). Se você já tiver um projeto na plataforma do Google, esse menu exibirá seu projeto padrão.
4. Selecione NEW PROJECT (Novo projeto).
5. Insira um nome para o produto e, depois, escolha CREATE (Criar).
6. Na barra de navegação esquerda, escolha APIs e Serviços e, em seguida, escolha Tela de consentimento do Oauth.
7. Insira as informações da aplicação, um Domínio da aplicação, Domínios autorizados e Informações de contato do desenvolvedor. Seus Domínios autorizados devem incluir `amazoncognito.com` e a raiz de seu domínio personalizado. Por exemplo: `example.com`. Escolha SAVE AND CONTINUE (Salvar e continuar).

8. 1. Em Escopos, escolha Adicionar ou remover escopos e escolha, no mínimo, os seguintes OAuth escopos.

   1. `.../auth/userinfo.email`
   2. `.../auth/userinfo.profile`
   3. OpenID

9. Em Test users (Testar usuários), escolha Add Users (Adicionar usuários). Insira seu e-mail e todos os outros usuários de teste autorizados e escolha Salvar e continuar.
10. Expanda a barra de navegação esquerda novamente, escolha Serviços APIs e, em seguida, escolha Credenciais.
11. Escolha CRIAR CREDENCIAIS e, em seguida, escolha ID OAuth do cliente.
12. Escolha um Application type (Tipo de aplicação) e forneça ao seu cliente um Name (Nome).
13. Em JavaScript Origens autorizadas, escolha ADICIONAR URI. Insira o domínio de seu grupo de usuários.

    ```
    https://<your-user-pool-domain>
    ```

14. Em Redirecionamento autorizado URIs, escolha ADICIONAR URI. Insira o caminho para o endpoint `/oauth2/idpresponse` do domínio de seu grupo de usuários.

    ```
    https://<your-user-pool-domain>/oauth2/idpresponse
    ```

15. Selecione CRIAR.
16. Armazene com segurança os valores que o Google exibe em Seu ID de cliente e Seu segredo do cliente. Forneça esses valores ao Amazon Cognito quando você adicionar um IdP do Google.

Para registrar uma aplicação na Apple

Para obter mais informações sobre como configurar o login com a Apple, consulte Configuring Your Environment for Sign in with Apple (Configurar o ambiente para login com a Apple) na documentação do Apple Developer.

1. Crie uma conta de desenvolvedor com a Apple.
2. Faça login com as credenciais da Apple.
3. Na barra de navegação à esquerda, escolha Certificates, Identifiers & Profiles (Certificados, identificadores e perfis).

4. Na barra de navegação à esquerda, escolha Identifiers (Identificadores).
5. Na página Identifiers (Identificadores), escolha o ícone +.
6. Na página Registrar um novo identificador, escolha Aplicativo e IDs, em seguida, escolha Continuar.
7. Na página Selecionar um tipo, escolha Aplicação e, depois, Continuar.
8. Na página Register an App ID (Registrar ID de uma aplicação), faça o seguinte:
    1. Em Description (Descrição), insira uma descrição.
    2. Em App ID Prefix (Prefixo do ID da aplicação), insira um Bundle ID (ID do pacote). Anote o valor em App ID Prefix (Prefixo do ID da aplicação). Você usará esse valor após escolher a Apple como seu provedor de identidade em Configurar o grupo de usuários com um IdP social.
    3. Em Capabilities (Recursos), escolha Sign In with Apple (Fazer login com a Apple) e, depois, selecione Edit (Editar).
    4. Na página Entrar com a Apple: Configuração do ID do aplicativo, escolha configurar o aplicativo como principal ou agrupado com outro aplicativo IDs e escolha Salvar.
    5. Escolha Continue (Continuar).
9. Na página Confirm your App ID (Confirmar ID do seu app), escolha Register (Registrar).
10. Na página Identifiers (Identificadores), escolha o ícone +.
11. Na página Registrar um novo identificador, escolha Serviços e IDs, em seguida, escolha Continuar.
12. Na página Register a Services ID (Registrar um ID de serviços), faça o seguinte:
    1. Em Description (Descrição), insira uma descrição.
    2. Em Identifier (Identificador), insira um identificador. Anote esse ID de serviços, pois você precisará desse valor depois de escolher a Apple como provedor de identidades em Configurar o grupo de usuários com um IdP social.
    3. Escolha Continuar e, depois, Registrar.
13. Escolha o ID de serviços que você acabou de criar na página Identificadores.
    1. Selecione Sign In with Apple (Fazer login com a Apple) e escolha Configure (Configurar).
    2. Na página Web Authentication Configuration (Configuração da autenticação web), selecione o ID da aplicação que você criou anteriormente como o Primary App ID (ID da aplicação principal).

3. Escolha o ícone + ao lado do site URLs.
4. Em Domains and subdomains (Domínios e subdomínios), insira o domínio do grupo de usuários sem um prefixo `https://`.

   ```
   <your-user-pool-domain>
   ```

5. Em Return URLs, insira o caminho para o `/oauth2/idpresponse` endpoint do seu domínio do grupo de usuários.

   ```
   https://<your-user-pool-domain>/oauth2/idpresponse
   ```

6. Selecione Próximo e, em seguida, selecione Concluído. Não é necessário verificar o domínio.
7. Escolha Continue (Continuar) e, depois, Save (Salvar).

14. No painel de navegação à esquerda, selecione Keys (Chaves).
15. Na página Keys (Chaves), escolha o ícone +.
16. Na página Register a New Key (Registrar uma chave nova), faça o seguinte:

    1. Em Key Name (Nome da chave), insira um nome de chave.
    2. Escolha Sign In with Apple (Fazer login com a Apple) e escolha Configure (Configurar).
    3. Na página Configurar chave, selecione o ID da aplicação que você criou anteriormente como o ID da aplicação principal. Escolha Salvar.
    4. Escolha Continue (Continuar) e, depois, Register (Registrar).

17. Na página Baixe sua chave, escolha Download para baixar a chave privada e anote a ID da chave. Em seguida, escolha Concluído. Você precisará dessa chave privada e do valor de Key ID (ID da chave) mostrado nesta página depois de escolher a Apple como provedor de identidade no Configurar o grupo de usuários com um IdP social.

Adicionar um IdP social ao seu grupo de usuários

Nesta seção, você configura um IdP social no grupo de usuários usando o ID e a chave secreta do cliente da seção anterior.

Para configurar um provedor de identidade social do grupo de usuários com o Console de gerenciamento da AWS

1. Acesse o console do Amazon Cognito. Você pode ser solicitado a fornecer suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Provedores sociais e externos. Localize Federated sign-in (Acesso federado) e selecione Add an identity provider (Adicionar um provedor de identidade).
5. Selecione um provedor de identidade social: Facebook, Google, Login with Amazon ou Sign in with Apple.
6. Escolha entre as seguintes etapas, com base em sua opção de provedor de identidade social:

   - Google e Login with Amazon: insira o ID do cliente da aplicação e o segredo do cliente da aplicação gerado na seção anterior.
   - Facebook: insira o ID do cliente da aplicação e o segredo do cliente da aplicação gerado na seção anterior e, em seguida, escolha uma versão da API (por exemplo, versão 2.12). Recomendamos escolher a versão mais recente possível, já que cada API do Facebook tem um ciclo de vida e uma data de desativação. Os escopos e atributos do Facebook podem variar entre as versões da API. Recomendamos testar seu login de identidade social com o Facebook para garantir que a federação funciona como previsto.
   - Fazer login com a Apple: insira o ID de serviços, o ID de equipe, o ID da chave e a chave privada gerados na seção anterior.
7. Insira os nomes dos Escopos autorizados que deseja utilizar. Os escopos definem quais atributos do usuário (como `name` e `email`) você deseja acessar com a aplicação. Para o Facebook, eles devem estar separados por vírgulas. Para o Google e o Login with Amazon, eles devem estar separados por espaços. Para Sign in with Apple, marque a caixa de seleção dos escopos que deseja acessar.

Provedor de identidade social	Escopos de exemplo
Facebook	`public_profile, email`
Google	`profile email openid`
Login da Amazon	`profile postal_code`

Provedor de identidade social	Escopos de exemplo
Fazer login com a Apple	`email name`

O consentimento do usuário da aplicação é solicitado para o fornecimento desses atributos à sua aplicação. Para mais informações sobre os escopos de provedores sociais, consulte a documentação do Google, Facebook, Login with Amazon ou do Sign in with Apple.

Em caso de acesso com Sign in with Apple, a seguir apresentamos os cenários de usuário cujos escopos talvez não sejam retornados:

- Um usuário final encontra falhas depois de sair da página de login com a Apple (elas podem ter origem de falhas internas dentro do Amazon Cognito ou de qualquer elemento escrito pelo desenvolvedor).
- O identificador de ID do serviço é usado em grupos de usuários e and/or outros serviços de autenticação.
- Um desenvolvedor adiciona outros escopos depois que o usuário faz login. Os usuários só recuperam novas informações quando se autenticam e atualizam seus tokens.
- Um desenvolvedor exclui o usuário e, a seguir, o usuário faz login novamente sem remover a aplicação de seu perfil de ID da Apple.

8. Mapeie atributos do provedor de identidade para o grupo de usuários. Para obter mais informações, consulte Coisas a saber sobre mapeamentos.
9. Escolha Criar.
10. No menu Clientes da aplicação, escolha um dos clientes da aplicação na lista e selecione Editar configurações de interface do usuário hospedada. Adicione o novo provedor de identidade social ao cliente da aplicação em Identity providers (Provedores de identidade).
11. Escolha Salvar alterações.

Testar a configuração do IdP social

Você pode criar um URL de login usando os elementos das duas seções anteriores. Use-o para testar a configuração do IdP social.

```
https://mydomain.auth.us-east-1.amazoncognito.com/login?
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Você pode encontrar o domínio na página do console Domain name (Nome do domínio) do grupo de usuários. O client_id está na página App client settings (Configurações de cliente de aplicação). Use o URL de retorno de chamada para o parâmetro redirect_uri. Esse é o URL da página para a qual o usuário será redirecionado após uma autenticação bem-sucedida.

> **Note**
>
> O Amazon Cognito cancela solicitações de autenticação que não são concluídas em 5 minutos e redireciona o usuário para o login gerenciado. A página exibe a mensagem de erro `Something went wrong` (Ocorreu algum problema).

Adicionar um provedor de identidades (IdP) SAML 2.0

Os usuários da aplicação podem fazer login por meio de um provedor de identidades (IdP) SAML 2.0. Você pode escolher o SAML 2.0 IdPs em vez das redes sociais IdPs quando seus clientes são clientes internos ou empresas vinculadas à sua organização. Quando um IdP social permite que todos os usuários se registrem em uma conta, é mais provável que um IdP SAML se associe a um diretório de usuários controlado por sua organização. Quer os usuários façam login diretamente ou por meio de terceiros, todos têm um perfil no grupo de usuários. Pule esta etapa se você não quiser adicionar login por meio de um provedor de identidade SAML.

Para obter mais informações, consulte Como usar provedores de identidade SAML com um grupo de usuários.

Você deve atualizar o provedor de identidades SAML e configurar o grupo de usuários. Consulte a documentação do seu provedor de identidades SAML para obter informações sobre como adicionar o grupo de usuários como uma aplicação ou parte dependente para o provedor de identidades SAML 2.0.

Você também precisa fornecer um endpoint do serviço da declaração (ACS) para o provedor de identidades SAML. Configure o endpoint a seguir no domínio do grupo de usuários para a vinculação POST SAML 2.0 no provedor de identidades SAML. Consulte Como configurar um domínio de grupo de usuários para obter mais informações sobre domínios do grupo de usuários.

```
https://Your user pool domain/saml2/idpresponse
```

```
With an Amazon Cognito domain:
https://<yourDomainPrefix>.auth.<region>.amazoncognito.com/saml2/idpresponse
With a custom domain:
https://Your custom domain/saml2/idpresponse
```

Você pode encontrar o prefixo do domínio e o valor da região para o grupo de usuários no menu Domínio no console do Amazon Cognito.

Para alguns provedores de identidades SAML, você também precisa fornecer o `urn` do provedor de serviços (SP), também chamado de URI de público ou ID da entidade SP no formato:

```
urn:amazon:cognito:sp:<yourUserPoolID>
```

Você pode encontrar o ID do grupo de usuários no painel Visão geral do grupo de usuários no console do Amazon Cognito.

Você também deve configurar o provedor de identidade SAML para fornecer valores de atributo para todos os atributos necessários no seu grupo de usuários. Normalmente, `email` é um atributo obrigatório para grupos de usuários. Nesse caso, o provedor de identidade SAML deve fornecer um valor `email` (solicitação) na declaração do SAML.

Os grupos de usuários do Amazon Cognito são compatíveis com a federação SAML 2.0 com endpoints de pós-vinculação. Isso elimina a necessidade de a aplicação recuperar ou analisar as respostas de declaração do SAML, pois o grupo de usuários recebe diretamente a resposta do SAML do seu provedor de identidades, por meio de um agente de usuário.

Para configurar um provedor de identidade SAML 2.0 no seu grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Provedores sociais e externos. Localize Federated sign-in (Acesso federado) e selecione Add an identity provider (Adicionar um provedor de identidade).
5. Selecione um provedor de identidade social SAML.
6. Insira Identifiers (Identificadores) separados por vírgulas. Um identificador diz ao Amazon Cognito que ele deve conferir o endereço de e-mail que um usuário insere quando faz o login. Em seguida, ele o direciona para o provedor que corresponde ao seu domínio.

7. Escolha Add sign-out flow (Adicionar fluxo de desconexão) se quiser que o Amazon Cognito envie solicitações de desconexão assinadas ao seu provedor quando um usuário se desconectar. Você deve configurar o provedor de identidades SAML 2.0 para enviar respostas de logout para o endpoint `https://<your Amazon Cognito domain>/saml2/logout` que é criado quando você configura o login gerenciado. O endpoint `saml2/logout` usa a associação POST.

   > **Note**
   >
   > Se essa opção for selecionada e seu provedor de identidades SAML esperar uma solicitação de logout assinada, você também precisará configurar o certificado de assinatura fornecido pelo Amazon Cognito com seu IdP SAML.
   > O IdP SAML processará a solicitação de logout assinada e fará logout do seu usuário da sessão do Amazon Cognito.

8. Escolha uma Metadata document source (Fonte de documento de metadados). Se seu provedor de identidade oferecer metadados SAML em um URL público, você pode escolher Metadata document URL (URL do documento de metadados) e inserir esse URL público. Do contrário, escolha Upload metadata document (Carregar documento de metadados) e, em seguida, um arquivo de metadados que você tenha baixado de seu provedor anteriormente.

   > **Note**
   >
   > Se seu provedor tiver um endpoint público, recomendamos que você insira um URL do documento de metadados em vez de carregar um arquivo. Isso permite que o Amazon Cognito atualize os metadados automaticamente. Normalmente, a atualização de metadados ocorre a cada seis horas ou antes de os metadados expirarem, o que ocorrer primeiro.

9. Selecione Map attributes between your SAML provider and your app (Mapear atributos entre seu provedor SAML e sua aplicação) para mapear atributos do provedor SAML ao perfil de usuário em seu grupo de usuários. Inclua os atributos obrigatórios do grupo de usuários no mapa de atributos.

   Por exemplo, quando você escolher o User pool attribute (Atributo do grupo de usuários) `email`, insira o nome de atributo SAML conforme ele aparece na afirmação SAML do seu provedor de identidade. Seu provedor de identidade pode oferecer exemplos de afirmações SAML como

referência. Alguns provedores de identidade usam nomes simples, como `email`, enquanto outros usam nomes de atributos formatados por URL, como o exemplo a seguir:

```
http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
```

10. Escolha Criar.

Conceitos básicos dos bancos de identidades do Amazon Cognito

Com os grupos de identidades do Amazon Cognito, você pode criar identidades exclusivas e atribuir permissões aos usuários. Seu banco de identidades pode trazer identidades dos seguintes tipos de serviços de autenticação:

- Usuários em um grupo de usuários do Amazon Cognito
- Usuários que realizam a autenticação por meio de provedores de identidades externos como Facebook, Google, Apple ou um provedor de identidades OIDC ou SAML
- Usuários autenticados por meio de seu próprio processo de autenticação existente

Depois que os usuários se autenticam com seu provedor e apresentam autorização a um grupo de identidades, eles recebem AWS credenciais temporárias. As credenciais dos usuários têm permissões que você define para acessar outros Serviços da AWS.

Tópicos

Criar um grupo de identidades no Amazon Cognito

Você pode criar um grupo de identidades por meio do console do Amazon Cognito ou usar o (AWS Command Line Interface CLI) ou o Amazon Cognito. APIs O procedimento a seguir é um guia geral para criar um banco de identidades no console. Você também pode ir direto para o console e seguir a experiência guiada e o conteúdo de ajuda em linha.

Para criar um novo grupo de identidades no console

1. Faça login no console do Amazon Cognito e selecione Bancos de identidades. Para atribuir permissões à sua entidade principal do IAM para que ela possa criar e gerenciar recursos

do Amazon Cognito, consulte AWS políticas gerenciadas para o Amazon Cognito. A política `AmazonCognitoPowerUser` é suficiente para a criação de bancos de identidades.

2. Selecione Criar banco de identidades.
3. Em Configurar confiança do banco de identidades, opte por configurar seu banco de identidades para Acesso autenticado, Acesso de convidado ou ambos.

   - Se você selecionou Acesso autenticado, escolha um ou mais Tipos de identidade que você deseja definir como origem de identidades autenticadas no banco de identidades. Se você configurar um Provedor de desenvolvedor personalizado, não poderá modificá-lo nem o excluir depois de criar o banco de identidades.

4. Em Configurar permissões, selecione um perfil padrão do IAM para usuários autenticados ou convidados em seu banco de identidades.

   a. Selecione Criar um perfil do IAM se quiser que o Amazon Cognito crie um perfil para você com permissões básicas e uma relação de confiança com seu banco de identidades. Insira um Nome de perfil do IAM para identificar seu novo perfil; por exemplo, `myidentitypool_authenticatedrole`. Selecione Visualizar documento de política para examinar as permissões que o Amazon Cognito atribuirá ao novo perfil do IAM.

   b. Você pode optar por usar uma função do IAM existente se já tiver uma função na sua Conta da AWS que queira usar. Você deve configurar sua política de confiança de perfis do IAM para incluir `cognito-identity.amazonaws.com`. Configure sua política de confiança de perfil para permitir que o Amazon Cognito assuma o perfil somente quando apresentar evidências de que a solicitação se originou de um usuário autenticado em seu banco de identidades específico. Para obter mais informações, consulte Permissões e confiança de função.

5. Em Connect identity providers, insira os detalhes dos provedores de identidade (IdPs) que você escolheu em Configurar a confiança do grupo de identidades. Você pode ser solicitado a fornecer informações do cliente do OAuth aplicativo, escolher um grupo de usuários do Amazon Cognito, escolher um IdP do IAM ou inserir um identificador personalizado para um provedor de desenvolvedores.

   a. Selecione Configurações de perfil para cada IdP. Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras. Com um IdP de grupo de usuários do Amazon Cognito, você também pode Selecionar um perfil com preferred_role em tokens. Para ter mais informações sobre a

declaração `cognito:preferred_role`, consulte Como atribuir valores de precedência a grupos.

i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.

b. Configure Atributos para controle de acesso para cada IdP. Os atributos para controle de acesso correlacionam as declarações do usuário com as tags de entidade principal que o Amazon Cognito aplica à sua sessão temporária. Você pode criar políticas do IAM para filtrar o acesso do usuário com base nas tags aplicadas à sessão.

i. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

ii. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

iii. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.

6. Em Configurar propriedades, insira um Nome em Nome do banco de identidades.
7. Em Autenticação básica (clássica), escolha se você deseja Ativar fluxo básico. Com o fluxo básico ativo, você pode ignorar as seleções de função que você fez para você IdPs e ligar diretamente. AssumeRoleWithWebIdentity Para obter mais informações, consulte Fluxo de autenticação dos bancos de identidades.
8. Em Tags, selecione Adicionar tag se quiser aplicar tags ao banco de identidades.
9. Em Revisar e criar, confirme as seleções que você fez para o novo banco de identidades. Selecione Editar para retornar ao assistente e alterar as configurações. Quando terminar, selecione Criar banco de identidades.

Configurar um SDK

Para usar os grupos de identidade do Amazon Cognito AWS Amplify, configure o AWS SDK para Java ou o. SDK para .NET Para obter mais informações, consulte os tópicos a seguir.

- Configurando o SDK para o JavaScript Guia do AWS SDK para JavaScript Desenvolvedor
- Documentação do Amplify no Amplify Dev Center
- Provedor de credenciais do Amazon Cognito no Guia do desenvolvedor do SDK para .NET

Integrar os provedores de identidade

Os bancos de identidades (identidades federadas) do Amazon Cognito são compatíveis com a autenticação de usuários por meio de grupos de usuários do Amazon Cognito, provedores de identidades federadas, incluindo Amazon, Facebook, Google, Apple e provedores de identidades SAML, além de identidades não autenticadas. Esse recurso também é compatível com Identidades autenticadas pelo desenvolvedor, que permite registrar e autenticar os usuários por meio de seu próprio processo de autenticação de backend.

Para saber mais sobre como usar um grupo de usuários do Amazon Cognito para criar seu próprio diretório, consulte Grupos de usuários do Amazon Cognito e Acessando Serviços da AWS usando um pool de identidades após o login.

Para saber mais sobre como usar provedores de identidade externos, consulte Bancos de identidades: provedores de identidade de terceiros.

Para saber mais sobre a integração do seu próprio processo de autenticação de backend, consulte Identidades autenticadas pelo desenvolvedor.

Obter credenciais

Os grupos de identidade do Amazon Cognito fornecem AWS credenciais temporárias para usuários convidados (não autenticados) e para usuários que se autenticaram e receberam um token. Com essas AWS credenciais, seu aplicativo pode acessar com segurança um back-end interno AWS ou externo por meio do Amazon API AWS Gateway. Consulte Como obter credenciais.

Aplicação de exemplo para bancos de identidades

O caso de uso mais comum dos grupos de identidade do Amazon Cognito é federar usuários de vários sistemas de login e entregar credenciais temporárias de acesso limitado AWS diretamente ao cliente. Isso elimina a necessidade de criar um agente de credenciais para obter permissões para acessar seus AWS recursos. Por exemplo, talvez seja necessário permitir que os usuários façam login com suas contas de rede social e acessem os ativos da aplicação do Amazon S3 para seu aplicativo móvel. Os bancos de identidades também fornecem credenciais aos usuários que fazem login com grupos de usuários.

Neste tutorial, você criará um aplicativo web em que poderá obter credenciais temporárias autenticadas e de convidado nos fluxos de autenticação avançados e básicos com provedores de identidade compatíveis (IdPs) em grupos de identidades. Se você já tem experiência em desenvolvimento web, baixe o aplicativo de exemplo em GitHub.

Baixe o aplicativo de exemplo em GitHub

Esta aplicação de exemplo demonstra os seguintes recursos dos bancos de identidades do Amazon Cognito:

Fluxos de autenticação em bancos de identidades

- Fluxo de autenticação aprimorado com detalhamento das solicitações de API
- Fluxo de autenticação básico com detalhamento das solicitações de API

Implementação do acesso de convidado (não autenticado)

- Forneça AWS service (Serviço da AWS) acesso limitado sem exigir login

Integração com provedores de identidades compatíveis

- Social IdPs (Facebook, Amazon, Twitter, Apple e Google) para acesso do consumidor
- Enterprise IdPs (por meio do OpenID Connect ou SAML) para usuários corporativos
- Grupos de usuários do Amazon Cognito

AWS gerenciamento de credenciais

- Troca de tokens do provedor de identidades por credenciais temporárias da AWS
- Usando credenciais temporárias para acessar AWS serviços com segurança

Após configurar a aplicação no servidor web de desenvolvimento e acessá-la em um navegador, você verá as opções a seguir.

Tópicos

Pré-requisitos

Antes de começar, você precisará configurar os recursos a seguir.

- Uma AWS conta com acesso ao Amazon Cognito. Se você não tiver uma AWS conta, siga as instruções emComeçando com AWS.
- Python 3.8 ou posterior instalado na sua máquina de desenvolvimento.
- GitHub acesso.
- AWS credenciais configuradas com permissões para fazer solicitações autenticadas ao Amazon Cognito. APIs Essas credenciais são obrigatórias para a autenticação do desenvolvedor.

Para obter mais informações sobre a implementação de AWS credenciais e federação de grupos de identidades em seu SDK específico, consulte. the section called “Como obter credenciais”

Configuração do provedor de autenticação

Para obter melhores resultados com esse aplicativo, configure e integre um ou mais provedores de identidade terceirizados (IdPs) ou grupos de usuários do Amazon Cognito ao seu pool de identidade do Amazon Cognito. Após concluir os pré-requisitos e antes de executar essa aplicação de demonstração, escolha quais provedores de identidades configurar. O console do Amazon Cognito orienta você no processo de configuração de bancos de identidades e provedores.

Grupos de usuários do Amazon Cognito

- the section called “Autenticação”
- the section called “Clientes de aplicativo”

Provedores de identidade social

- Google: the section called “Google”
- Facebook: the section called “Facebook”
- Amazon: the section called “Login da Amazon”

Provedores OpenID Connect (OIDC)

- the section called “Provedores Open ID Connect”

Provedores de SAML

- the section called “Provedores de identidade SAML”

> **Note**
>
> Para essa aplicação de demonstração, não é necessário configurar todos os provedores de identidades compatíveis. Você pode começar com um que corresponda ao seu caso de uso. Cada link fornece instruções detalhadas de configuração.

Implantar a aplicação de demonstração

Clone o repositório

1. Abra uma janela do terminal.
2. Clone o repositório `aws-doc-sdk-examples` ou recupere esta pasta no repositório.

```
git clone https://github.com/awsdocs/aws-doc-sdk-examples.git
```

3. Navegue até o diretório de projeto do .

```
cd python/example_code/cognito/scenarios/identity_pools_example_demo/web
```

Criar um banco de identidades do

Para criar um banco de identidades no Amazon Cognito para sua aplicação, siga as instruções em the section called “Como configurar bancos de identidades”.

Como configurar um banco de identidades para a aplicação de demonstração

1. Abra o console do Amazon Cognito.
2. No menu de navegação à esquerda, selecione Grupos de identidades. Escolha um banco de identidades existente ou crie um.
3. Em Acesso de usuário, habilite Acesso autenticado e Acesso de convidado. Configure um perfil do IAM novo ou existente e atribua a ele as permissões que você deseja conceder a cada tipo de usuário.
4. Em Acesso de usuário, configure todos os provedores de identidades que deseja configurar.
5. Em Propriedades do grupo de identidades, habilite Autenticação básica (clássica).
6. Mantenha o navegador aberto no console do banco de identidades. Você usará o ID do banco de identidades e outras informações de configuração na configuração da sua aplicação.

Configurar e executar a aplicação

As etapas a seguir guiarão você na configuração inicial da sua aplicação de demonstração.

Como configurar a aplicação de demonstração

1. Abra uma linha de comando em `python/example_code/cognito/scenarios/identity_pools_example_demo/web` em seu clone `aws-doc-sdk-examples`.
2. Crie um arquivo `.env` copiando o arquivo de ambiente de exemplo.

```
cp .env.example .env
```

3. Abra o arquivo `.env` em um editor de textos. Substitua os valores de exemplo no arquivo pelos seus próprios valores de configuração.

4. Instale as dependências de backend.

```
pip install -r requirements.txt
```

5. Inicie o servidor de backend:

```
cd backend
python oauth_server.py
```

6. Abra uma nova janela de terminal, navegue até o diretório do projeto e inicie o servidor de frontend:

```
cd frontend
python -m http.server 8001
```

7. Abra seu navegador e acesse a aplicação em http://localhost:8001. Seu navegador exibirá a interface da aplicação de demonstração, pronta para testar a autenticação de bancos de identidades.

Explorar os métodos de autenticação no banco de identidades

Esta seção orienta você pelos fluxos de autenticação básica e aprimorada usando a aplicação de demonstração de bancos de identidades do Amazon Cognito. Com esta demonstração, você aprenderá como os grupos de identidades funcionam com vários provedores de identidade para fornecer AWS credenciais temporárias aos usuários do seu aplicativo.

Na seção Demonstração interativa da aplicação de exemplo, você primeiro escolherá entre dois tipos de acesso compatíveis com bancos de identidades.

Acesso não autenticado (convidado)

Forneça AWS credenciais aos usuários que ainda não se autenticaram.

Acesso autenticado

Troque tokens do provedor de identidade por AWS credenciais com um escopo completo de permissões disponíveis. Escolha um provedor de identidades dentre aqueles que você configurou no arquivo `.env`.

Acesso não autenticado (convidado)

Esta etapa demonstra como obter AWS credenciais temporárias para usuários não autenticados (convidados) por meio do recurso de acesso de convidados do seu grupo de identidades. Na aplicação de demonstração, você testará os fluxos avançado e básico para ver como os bancos de identidades emitem credenciais sem exigir o login do usuário. O acesso de convidado usa a mesma sequência de API do acesso autenticado, mas sem fornecer tokens de provedor de identidade (como OAuth tokens do Google, Facebook ou declarações SAML de provedores corporativos).

Continue a leitura se quiser informações sobre como fornecer acesso limitado à AWS para usuários sem exigir autenticação. Depois de implementar o acesso de convidado, você aprenderá a fornecer AWS credenciais com segurança a usuários anônimos e a entender as diferenças entre os dois fluxos de autenticação.

> **Important**
>
> O acesso não autenticado pode emitir credenciais para qualquer pessoa com acesso à Internet, por isso é melhor usado para AWS recursos que exigem segurança mínima, como ativos públicos APIs e gráficos. Antes de prosseguir com essa etapa, verifique se você configurou o banco de identidades com o acesso de convidado habilitado e garanta que as políticas do IAM adequadas estejam em vigor para limitar as permissões.

Guest access with enhanced flow

O fluxo aprimorado é uma abordagem simplificada para obter credenciais da AWS para usuários não autenticados com duas solicitações de API.

Como testar o acesso de convidado com o fluxo aprimorado

1. Na aplicação de demonstração, navegue até a seção Demonstração interativa.
2. Selecione a guia Acesso de convidado.
3. Selecione a guia Fluxo aprimorado.
4. Clique em Testar acesso de convidado.
5. O aplicativo obtém AWS credenciais temporárias de seus grupos de identidades sem solicitações adicionais de autenticação.
6. Após a autenticação bem-sucedida, você verá a interface da web exibindo o painel Resultados e terá duas opções para explorá-lo:

a. Botão Exibir somente credenciais: escolha esse botão se quiser ver diretamente AWS as credenciais temporárias geradas sem os detalhes do fluxo da API.

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
    "Expiration": "2025-08-07T00:58:21-07:00"
  }
}
```

b. Exibir botão detalhado do fluxo da API: escolha esse botão se quiser ver as solicitações da step-by-step API.

- Solicitação de API `GetId()` com seu `identityPoolId`. Não são necessários tokens de autenticação para acesso de convidado.

  ```
  {
    "IdentityPoolId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
  ```

  Se for válido, ele encontrará ou criará e retornará o `IdentityID` do usuário. Um exemplo de resposta é semelhante a:

  ```
  {
    "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
  ```

- `GetCredentialsForIdentity()` com o `identityPoolId` retornado.

  ```
  POST GetCredentialsForIdentity
  {
    "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
  }
  ```

  Cognito valida o acesso de convidados, assume a função não autenticada internamente e retorna uma credencial temporária da AWS. AWS STS(Não há autenticação do

IAM nessa chamada; a confiança de função deve permitir `cognito-identity-amazonzaws.com`.)

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
    "Expiration": "2025-08-07T00:58:21-07:00"
  }
}
```

Guest access with basic flow

O fluxo básico fornece controle granular sobre o processo de autenticação com solicitações de API separadas para recuperação de identidade e geração de credenciais.

Como testar o acesso de convidado com o fluxo básico

1. Na aplicação de demonstração, navegue até a seção Demonstração interativa.
2. Selecione a guia Acesso de convidado.
3. Selecione a guia Fluxo básico.
4. Clique em Testar acesso de convidado.
5. O aplicativo obtém AWS credenciais temporárias de seus grupos de identidades sem solicitações adicionais de autenticação.
6. Após a autenticação bem-sucedida, você verá a interface da web exibindo o painel Resultados e terá duas opções para explorá-lo.

   a. Botão Exibir somente credenciais: escolha esse botão se quiser ver diretamente AWS as credenciais temporárias geradas sem os detalhes do fluxo da API.

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
    "Expiration": "2025-08-12T13:36:17-07:00"
```

```
  }
}
```

b. Exibir botão detalhado do fluxo da API: escolha esse botão se quiser ver as solicitações da step-by-step API.

- Solicitação de API `GetId()` com o ID do banco de identidades. Não são necessários tokens de autenticação para acesso de convidado.

```
POST GetId
{
  "IdentityPoolId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

Se for válido, ele encontrará ou criará e retornará o `IdentityID` do usuário. Um exemplo de resposta é semelhante a:

```
{
   "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

- `GetOpenIdToken()` com o `IdentityID` retornado e o mesmo mapa `Logins`.

```
POST GetOpenIdToken
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Token": "eyJraWQiOiJFWAMPLE......"
}
```

O que acontece nessa etapa: o Amazon Cognito emite um token de identidade web do OpenID Connect de curta duração, proveniente de cognito-identity.amazonaws.com, que representa esse `IdentityId`. O token inclui declarações do OIDC que AWS STS avaliam, incluindo aud (seu ID de grupo de identidade) e amr (autenticado ou não autenticado). A política de confiança do seu perfil do IAM deve exigir essas declarações.

- `AssumeRoleWithWebIdentity()`- Seu aplicativo liga AWS STS diretamente para trocar o token OpenID do Amazon Cognito por credenciais temporárias AWS

```
POST sts:AssumeRoleWithWebIdentity
{
  "RoleArn": "arn:aws:iam::111122223333:role/
Cognito_IdentityPoolUnauth_Role",
  "WebIdentityToken": "eyJraWQiOiJFWAMPLE......"
}
```

Resposta:

```
{
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "FwoGZXIvYXdzEEXAMPLE......"
  }
}
```

O que acontece nessa etapa: após a validação: retorna as credenciais temporárias da AWS

Usar as credenciais temporárias

Essas credenciais temporárias funcionam como AWS credenciais padrão, mas com permissões limitadas definidas pela função não autenticada do IAM do seu grupo de identidades. Você pode usá-los com qualquer AWS SDK ou AWS CLI. Para obter mais informações sobre a configuração AWS SDKs com credenciais, consulte Provedores de credenciais padronizados no Guia de referência de ferramentas AWS SDKs e ferramentas.

Os exemplos abaixo não constituem uma lista completa, mas mostram maneiras comuns pelas quais o recurso de convidado de um banco de identidades pode melhorar a experiência do usuário.

Conteúdo público somente para leitura

Os exemplos a seguir configuram provedores de credenciais para acesso limitado ao Amazon S3 como usuário convidado.

Python

```
# Example: Using credentials with boto3
import boto3

# Configure client with temporary credentials
s3_client = boto3.client(
    's3',
    aws_access_key_id='AKIAIOSFODNN7EXAMPLE',
    aws_secret_access_key='wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY',
    aws_session_token='IQoJb3JpZ2luX2VjEEXAMPLE......'
)

# Make API requests within IAM role permissions
response = s3_client.list_objects_v2(Bucket='my-public-bucket')

# Access public content
for obj in response.get('Contents', []):
    print(f"File: {obj['Key']}, Size: {obj['Size']} bytes")
```

JavaScript

```
// Example: Accessing public content
import { S3Client, GetObjectCommand } from "@aws-sdk/client-s3";

const s3Client = new S3Client({
    region: "us-east-1",
    credentials: {
        accessKeyId: 'AKIAIOSFODNN7EXAMPLE',
        secretAccessKey: 'wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY',
        sessionToken: 'IQoJb3JpZ2luX2VjEEXAMPLE......'
    }
});

// Access public images or documents
const response = await s3Client.send(new GetObjectCommand({
    Bucket: 'my-public-content',
    Key: 'product-catalog.pdf'
}));
```

Características “Try-before-login”

Os exemplos a seguir usam o acesso somente leitura ao Amazon DynamoDB como usuário convidado.

Python

```
# Example: Limited app functionality for trial users
import boto3

dynamodb = boto3.client(
    'dynamodb',
    aws_access_key_id='AKIAIOSFODNN7EXAMPLE',
    aws_secret_access_key='wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY',
    aws_session_token='IQoJb3JpZ2luX2VjEEXAMPLE......'
)

# Allow guest users to view sample data (limited to 5 items)
response = dynamodb.scan(TableName='SampleProducts', Limit=5)
```

JavaScript

```
// Example: Limited app functionality for trial users
import { DynamoDBClient, ScanCommand } from "@aws-sdk/client-dynamodb";

const dynamodbClient = new DynamoDBClient({
    region: "us-east-1",
    credentials: {
        accessKeyId: 'AKIAIOSFODNN7EXAMPLE',
        secretAccessKey: 'wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY',
        sessionToken: 'IQoJb3JpZ2luX2VjEEXAMPLE......'
    }
});

// Allow guest users to view sample data (limited to 5 items)
const response = await dynamodbClient.send(new ScanCommand({
    TableName: 'SampleProducts',
    Limit: 5
}));
```

Autenticação do provedor de identidades social

Esta etapa explora o fluxo geral de uso de provedores de identidades sociais com bancos de identidades do Amazon Cognito. A autenticação social fornece uma experiência de login familiar e, ao mesmo tempo, mantém a segurança por meio do gerenciamento de identidades federadas. Você pode fazer login a partir de um provedor de identidade social (IdP), como Google, Facebook e Amazon, e depois trocar esse token de IdP por credenciais temporárias. AWS A integração do Twitter e da Apple também é compatível com bancos de identidades, mas não é compatível na aplicação de exemplo.

O banco de identidades em si não é um diretório de usuários. Ele não armazena senhas nem campos de perfil. Em vez disso, ela confia no externo IdPs para autenticar o usuário e se concentra em autorizar esse usuário já autenticado a ligar diretamente para os AWS serviços, vendendo credenciais para funções do IAM.

Social identity provider with enhanced flow

Esta seção mostra como você pode usar um provedor de identidades social para conectar um usuário e, usando o fluxo aprimorado, trocar o token do provedor em um banco de identidades do Amazon Cognito por credenciais temporárias para solicitar recursos da AWS .

Usar o login social com o fluxo aprimorado na aplicação de exemplo

1. Na aplicação de demonstração, navegue até a seção Demonstração interativa.
2. Selecione a guia Acesso autenticado.
3. Selecione a guia Fluxo aprimorado.
4. Escolha um provedor social compatível que você configurou, por exemplo, Fazer login com o Google, Fazer login com o Facebook ou Fazer login com a Amazon.
5. Faça login e concorde em compartilhar dados do usuário com a aplicação.
6. O provedor redireciona de volta para o URI de redirecionamento da aplicação.
7. O aplicativo envia o token do provedor para seu grupo de identidades e recupera credenciais temporárias AWS
8. A aplicação exibe o painel Resultados na interface da web.

   Após a autenticação bem-sucedida, você verá a interface da web exibindo o painel Resultados e terá duas opções para explorá-lo:

a. Botão Exibir somente credenciais: escolha esse botão se quiser ver diretamente AWS as credenciais temporárias geradas sem os detalhes do fluxo da API.

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
    "Expiration": "2025-08-12T13:36:17-07:00"
  }
}
```

b. Exibir botão detalhado do fluxo da API: escolha esse botão se quiser ver as solicitações da step-by-step API.

   • A aplicação conecta o usuário com um IdP social e obtém o token do provedor. Os bancos de identidades aceitam estes artefatos dos provedores sociais:

Provedor de identidades	Chave do provedor do Cognito	Finalidade
Google	`accounts.google.com`	OAuth 2.0 tokens do Google Sign-In
Facebook	`graph.facebook.com`	Tokens de acesso do login do Facebook
Amazon	`www.amazon.com`	OAuth tokens do Login with Amazon

   Após a autenticação bem-sucedida com o provedor social, seu aplicativo recebe uma OAuth resposta contendo o token de acesso e outros detalhes de autenticação:

```
{
   "access_token": "ya29.A0AS3H6NEXAMPLE......",
   "expires_in": 3599,
   "scope": "openid https://www.examplesocial....",
   "token_type": "Bearer",
```

```
   "id_token": "eyJhbGciOiJSUzI1NiIsEXAMPLE......"
}
```

- Solicitação de API `GetId()` com o ID do banco de identidades e um mapa `Logins` contendo o token do provedor social.

```
POST GetId
{
  "IdentityPoolId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Logins": {
    "accounts.google.com": "eyJhbGciOiJSUzI1NiIsEXAMPLE......"
  }
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

- `GetCredentialsForIdentity()` com o `IdentityID` retornado e o mesmo mapa `Logins`.

```
POST GetCredentialsForIdentity
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Logins": {
    "accounts.google.com": "eyJhbGciOiJSUzI1NiIsEXAMPLE......"
  }
}
```

Resposta:

```
{
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
    "Expiration": "2025-08-07T00:58:21-07:00"
  },
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
```

```
}
```

O que aconteceu: o Amazon Cognito validou o token em relação ao provedor configurado, escolheu uma função do IAM com base na configuração do seu provedor e ligou AWS STS em seu nome. Seu banco de identidades então retornou as credenciais temporárias.

Social identity provider with basic flow

Esta seção mostra como você pode usar um provedor de identidade social para cadastrar um usuário e, usando o fluxo básico, trocar o token do provedor em um pool de identidade do Amazon Cognito por credenciais temporárias para chamar serviços. AWS

Usar o login social com o fluxo básico na aplicação de exemplo

1. Na aplicação de demonstração, navegue até a seção Demonstração interativa.
2. Selecione a guia Acesso autenticado.
3. Selecione a guia Fluxo básico.
4. Escolha um provedor social compatível que você configurou, por exemplo, Fazer login com o Google, Fazer login com o Facebook ou Fazer login com a Amazon.
5. Faça login e concorde em compartilhar dados do usuário com a aplicação.
6. O provedor redireciona de volta para o URI de redirecionamento da aplicação.
7. O aplicativo envia o token do provedor para seu grupo de identidades e recupera credenciais temporárias AWS
8. A aplicação exibe o painel Resultados na interface da web.

   Após a autenticação bem-sucedida, você verá a interface da web exibindo o painel Resultados e terá duas opções para explorá-lo:

   a. Botão Exibir somente credenciais: escolha esse botão se quiser ver diretamente AWS as credenciais temporárias geradas sem os detalhes do fluxo da API.

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
    "Expiration": "2025-08-12T13:36:17-07:00"
```

```
  }
}
```

b. Exibir botão detalhado do fluxo da API: escolha esse botão se quiser ver as solicitações da step-by-step API.

- A aplicação conecta o usuário com um IdP social e obtém o token do provedor. Os bancos de identidades aceitam estes artefatos dos provedores sociais:

Provedor de identidades	Chave do provedor do Cognito	Finalidade
Google	`accounts.google.com`	OAuth 2.0 tokens do Google Sign-In
Facebook	`graph.facebook.com`	Tokens de acesso do login do Facebook
Amazon	`www.amazon.com`	OAuth tokens do Login with Amazon

Após a autenticação bem-sucedida com o provedor social, seu aplicativo recebe uma OAuth resposta contendo o token de acesso e outros detalhes de autenticação:

```
{
   "access_token": "ya29.A0AS3H6NEXAMPLE......",
   "expires_in": 3599,
   "scope": "openid https://www.examplesocial....",
   "token_type": "Bearer",
   "id_token": "eyJhbGciOiJSUzI1NiIsEXAMPLE......"
}
```

- Solicitação de API `GetId()` com o ID do banco de identidades e um mapa `Logins` contendo o token do provedor social.

```
POST GetId
{
  "IdentityPoolId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Logins": {
    "accounts.google.com": "token..."
```

```
  }
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

- `GetOpenIdToken()` com o `IdentityID` retornado e o mesmo mapa Logins.

```
POST GetOpenIdToken
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Logins": {
    "accounts.google.com": "token..."
  }
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Token": "eyJraWQiOiJFWAMPLE......"
}
```

- `AssumeRoleWithWebIdentity()` com o token OpenID

```
POST AssumeRoleWithWebIdentity
{
  "RoleArn": "arn:aws:iam::111122223333:role/Cognito_IdentityPoolAuth_Role",
  "WebIdentityToken": "eyJraWQiOiJFWAMPLE......"
}
```

Resposta:

```
{
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
```

```
    "Expiration": "2025-08-12T14:36:17-07:00"
  }
}
```

O que aconteceu: o Amazon Cognito validou o token em relação ao provedor configurado e emitiu um token OpenID. O aplicativo ligou AWS STS diretamente para assumir uma função do IAM e receber credenciais temporárias.

Noções básicas sobre o acesso social

- Os usuários sociais recebem AWS credenciais temporárias por meio dos grupos de identidade do Amazon Cognito após se autenticarem com seu provedor social.
- Cada usuário autenticado recebe um ID de identidade exclusivo que persiste em todas as sessões.
- Essas credenciais estão vinculadas a um perfil do IAM projetado especificamente para acesso autenticado, fornecendo permissões mais amplas do que o acesso de convidado.
- Os tokens do provedor social são trocados por AWS credenciais, mantendo a identidade e as permissões do usuário.

Autenticação de grupos de usuários do Amazon Cognito

Esta etapa explora a autenticação do Amazon Cognito com a integração de login gerenciado de grupos de usuários. Ao vincular um grupo de usuários como um IdP a um banco de identidades, os tokens do grupo de usuários autorizam o banco de identidades a emitir credenciais temporárias.

User pool authentication with enhanced flow

O fluxo aprimorado fornece uma abordagem simplificada para obter credenciais da AWS por meio de bancos de identidades do Amazon Cognito com uma única solicitação de API.

Usar a autenticação do grupo de usuários do Amazon Cognito com o fluxo aprimorado do banco de identidades

1. Na aplicação de demonstração, navegue até a seção Demonstração interativa.
2. Selecione a guia Acesso autenticado.
3. Selecione a guia Fluxo aprimorado.
4. Selecione Fazer login com grupos de usuários do Amazon Cognito.

5. Conclua o login com seu nome de usuário e senha no login gerenciado.
6. O grupo de usuários redireciona de volta ao URI de redirecionamento da aplicação com um código de autorização.
7. A aplicação troca o código de autorização com o grupo de usuários por tokens web JSON.
8. O aplicativo troca o token de ID com seu grupo de identidades por AWS credenciais temporárias
9. A aplicação exibe o painel Resultados na interface da web.

   Após a autenticação bem-sucedida, você verá a interface da web exibindo o painel Resultados e terá duas opções para explorá-lo:

   a. Botão Exibir somente credenciais: escolha esse botão se quiser ver diretamente AWS as credenciais temporárias geradas sem os detalhes do fluxo da API.

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
    "Expiration": "2025-08-12T13:36:17-07:00"
  }
}
```

   b. Exibir botão detalhado do fluxo da API: escolha esse botão se quiser ver as solicitações da step-by-step API.

   - A aplicação conecta o usuário com o Amazon Cognito. Após a autenticação bem-sucedida com o grupo de usuários, seu aplicativo recebe uma resposta OAuth 2.0 contendo o token de ID (JWT). Os bancos de identidades aceitam tokens de ID JWT dos grupos de usuários usando este formato de chave de provedor:

Provedor de identidades	Chave do provedor do Cognito	Finalidade
Grupo de usuários do Amazon Cognito	`cognito-idp.{region}.amazonaws.com/{user-pool-id}`	Tokens de ID JWT dos grupos de usuários do Amazon Cognito

Após a autenticação bem-sucedida com o grupo de usuários, seu aplicativo recebe uma resposta OAuth 2.0 contendo o token de ID (JWT):

```
{
   "id_token": "eyJraWQiOiJFWAMPLE......",
   "token_type": "Bearer",
   "expires_in": 3600
}
```

- Solicitação de API `GetId()` com seu `identityPoolId` e um mapa `Logins` que inclui a chave de provedor do grupo de usuários mapeada para `id_token`. O Amazon Cognito verificou que a assinatura, o emissor, a expiração e o público (`aud`) do token de ID do grupo de usuários correspondem a um dos clientes do aplicativo que IDs você registrou para esse IdP do grupo de usuários no grupo de identidades.

```
POST GetId
{
  "AccountId": "111122223333",
  "IdentityPoolId": "us-east-1:1ac4a76d-1fef-48aa-83af-4224799c0b5c",
  "Logins": {
    "cognito-idp.us-east-1.amazonaws.com/us-east-1_EXAMPLE123":
 "eyJraWQiOiJFWAMPLE......"
  }
}
```

Se for válido, ele encontrará ou criará e retornará o `IdentityID` do usuário. Um exemplo de resposta é semelhante a:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

- `GetCredentialsForIdentity()` com o `identityPoolId` retornado e o mesmo mapa `Logins` com o `id_token`. O Amazon Cognito revalida a assinatura, o emissor, a expiração e o público-alvo (`aud`) do token de ID do grupo de usuários que IDs você registrou para esse IdP do grupo de usuários no grupo de identidades.

```
POST GetCredentialsForIdentity
{
```

```
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Logins": {
    "cognito-idp.us-east-1.amazonaws.com/us-east-1_EXAMPLE123":
 "eyJraWQiOiJFWAMPLE......"
  }
}
```

Se válido, ele escolhe uma função do IAM (roles-in-token, regras ou padrão), liga AWS STS em seu nome e retorna AWS credenciais temporárias. Um exemplo de resposta é semelhante a:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Credentials": {
    "AccessKeyId": "ASIAW7TIP7EJEXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
    "Expiration": "2025-08-12T14:36:17-07:00"
  }
}
```

User pool authentication with basic flow

O fluxo básico fornece controle granular sobre o processo de autenticação com solicitações de API separadas para recuperação de identidade e geração de credenciais.

Usar a autenticação do grupo de usuários do Amazon Cognito com o fluxo básico do banco de identidades

1. Na aplicação de demonstração, navegue até a seção Demonstração interativa.
2. Selecione a guia Acesso autenticado.
3. Selecione a guia Fluxo básico.
4. Selecione Fazer login com grupos de usuários do Amazon Cognito.
5. Conclua o login com seu nome de usuário e senha no login gerenciado.
6. O grupo de usuários redireciona de volta ao URI de redirecionamento da aplicação com um código de autorização.
7. A aplicação troca o código de autorização com o grupo de usuários por tokens web JSON.

8. O aplicativo troca o token de ID com seu grupo de identidades por AWS credenciais temporárias
9. A aplicação exibe o painel Resultados na interface da web.

   Após a autenticação bem-sucedida, você verá a interface da web exibindo o painel Resultados e terá duas opções para explorá-lo:

   a. Botão Exibir somente credenciais: escolha esse botão se quiser ver diretamente AWS as credenciais temporárias geradas sem os detalhes do fluxo da API.

   ```
   {
     "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
     "Credentials": {
       "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
       "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
       "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
       "Expiration": "2025-08-12T13:36:17-07:00"
     }
   }
   ```

   b. Exibir botão detalhado do fluxo da API: escolha esse botão se quiser ver as solicitações da step-by-step API.

   - A aplicação conecta o usuário com um grupo de usuários do Amazon Cognito e obtém o token de ID (JWT) como artefato. Após a autenticação bem-sucedida com o grupo de usuários, seu aplicativo recebe uma OAuth resposta contendo o token de ID (JWT). Os bancos de identidades usam esse token para autenticação:

     ```
     {
        "id_token": "eyJraWQiOiJFWAMPLE......",
        "token_type": "Bearer",
        "expires_in": 3600
     }
     ```

   - Solicitação de API `GetId()` com o ID do banco de identidades e um mapa `Logins` que inclui a chave do provedor do grupo de usuários e o token de ID como valor. O Amazon Cognito verificou que a assinatura, a expiração e o público (aud) do token de ID do grupo de usuários correspondem a um dos clientes do aplicativo IDs que você registrou para esse IdP do grupo de usuários no grupo de identidades.

     ```
     POST GetId
     ```

```
{
  "AccountId": "111122223333",
  "IdentityPoolId": "us-east-1:1ac4a76d-1fef-48aa-83af-4224799c0b5c",
  "Logins": {
    "cognito-idp.us-east-1.amazonaws.com/us-east-1_EXAMPLE123":
 "eyJraWQiOiJFWAMPLE......"
  }
}
```

Se for válido, ele encontrará ou criará e retornará o `IdentityID` do usuário. Um exemplo de resposta é semelhante a:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

- `GetOpenIdToken()` com o `IdentityID` retornado e o mesmo mapa `Logins`.

```
POST GetOpenIdToken
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Logins": {
    "cognito-idp.us-east-1.amazonaws.com/us-east-1_EXAMPLE123":
 "eyJraWQiOiJFWAMPLE......"
  }
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Token": "eyJraWQiOiJFWAMPLE......"
}
```

O que acontece nessa etapa: o Amazon Cognito emite um token de identidade web do OpenID Connect de curta duração, proveniente de cognito-identity.amazonaws.com, que representa esse `IdentityId`. O token inclui declarações do OIDC que AWS STS avaliam, incluindo aud (seu ID de grupo de identidade) e amr (autenticado ou não autenticado). A política de confiança do seu perfil do IAM deve exigir essas declarações.

- `AssumeRoleWithWebIdentity()`- Seu aplicativo liga AWS STS diretamente para trocar o token OpenID do Amazon Cognito por credenciais temporárias AWS

```
POST sts:AssumeRoleWithWebIdentity
{
  "RoleArn": "arn:aws:iam::111122223333:role/Cognito_IdentityPoolAuth_Role",
  "WebIdentityToken": "eyJraWQiOiJFWAMPLE......",
  "RoleSessionName": "CognitoIdentityCredentials"
}
```

Resposta:

```
{
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "FwoGZXIvYXdzEEXAMPLE......",
    "Expiration": "2025-08-12T14:36:17-07:00"
  },
  "AssumedRoleUser": {
    "AssumedRoleId": "AROAW7TIP7EJYEXAMPLE:CognitoIdentityCredentials",
    "Arn": "arn:aws:sts::111122223333:assumed-role/
Cognito_IdentityPoolAuth_Role/CognitoIdentityCredentials"
  }
}
```

O que a aplicação de demonstração fez: sua aplicação enviou o token OpenID de `GetOpenIdToken()` para AWS STS, solicitando credenciais temporárias. O AWS STS realizou verificações de validação e emitiu credenciais:

Noções básicas sobre o acesso ao grupo de usuários

- Os usuários do grupo de usuários recebem AWS credenciais temporárias por meio dos grupos de identidade do Amazon Cognito.
- Essas credenciais estão vinculadas a um perfil do IAM especificada na configuração do banco de identidades.
- Os tokens de ID do grupo de usuários são trocados por AWS credenciais por meio do pool de identidades.

Autenticação SAML

Esta etapa explora a autenticação SAML. Os usuários podem entrar com provedores de identidade corporativa que oferecem suporte ao SAML para acessar AWS os serviços. O fluxo básico com SAML não é compatível na aplicação de exemplo.

SAML authentication with enhanced flow

Esta seção mostra como você pode usar um provedor de identidade SAML para cadastrar um usuário e, usando o fluxo aprimorado, trocar a declaração de SAML em um pool de identidade do Amazon Cognito por credenciais temporárias AWS para chamar serviços. AWS

Usar a autenticação SAML com o fluxo aprimorado do banco de identidades

1. Na aplicação de demonstração, navegue até a seção Demonstração interativa.
2. Selecione a guia Acesso autenticado.
3. Selecione a guia Fluxo aprimorado.
4. Selecione Fazer login com o provedor SAML.
5. Conclua o login com suas credenciais corporativas.
6. O grupo de usuários redireciona de volta para o URI de redirecionamento da aplicação com uma declaração SAML.
7. A aplicação troca o código de autorização com o grupo de usuários por tokens web JSON.
8. O aplicativo troca a resposta SAML com seu grupo de identidades por credenciais temporárias AWS
9. A aplicação exibe o painel Resultados na interface da web.

   Após a autenticação bem-sucedida, você verá a interface da web exibindo o painel Resultados e terá duas opções para explorá-lo:

   a. Botão Exibir somente credenciais: escolha esse botão se quiser ver diretamente AWS as credenciais temporárias geradas sem os detalhes do fluxo da API.

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
```

```
    "Expiration": "2025-08-12T13:36:17-07:00"
  }
}
```

b. Exibir botão detalhado do fluxo da API: escolha esse botão se quiser ver as solicitações da step-by-step API.

- A aplicação conecta o usuário com um IdP SAML e obtém a resposta SAML. Os bancos de identidades aceitam declarações SAML de provedores corporativos usando o ARN do provedor SAML como chave:

Provedor de identidades	Chave do provedor do Cognito	Finalidade
Provedor SAML	`arn:aws:iam::111122223333:saml-provider/EXAMPLE`	Declarações de SAML da empresa IdPs

Após a autenticação bem-sucedida com o provedor SAML, a aplicação recebe uma resposta SAML via HTTP POST para o URL de retorno de chamada:

```
{
  "saml_response": "PD94bWwgdmVyc2lvbj0iMS4wIiBFWAMPLE...",
  "provider_arn": "arn:aws:iam::111122223333:saml-provider/EXAMPLE",
  "status": "Authentication successful"
}
```

- Solicitação de API `GetId()` com o ID do banco de identidades e um mapa `Logins` contendo o ARN e a declaração do provedor SAML.

```
POST GetId
{
  "AccountId": "111122223333",
  "IdentityPoolId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Logins": {
    "arn:aws:iam::111122223333:saml-provider/EXAMPLE":
 "PD94bWwgdmVyc2lvbj0iMS4wIiBFWAMPLE..."
  }
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

- `GetCredentialsForIdentity()` com o `IdentityID` retornado e o mesmo mapa `Logins`.

```
POST GetCredentialsForIdentity
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Logins": {
    "arn:aws:iam::111122223333:saml-provider/EXAMPLE":
 "PD94bWwgdmVyc2lvbj0iMS4wIiBFWAMPLE..."
  }
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......"
  }
}
```

O que aconteceu: o Amazon Cognito validou a declaração do SAML em relação ao provedor configurado, escolheu uma função do IAM com base nos atributos ou regras do SAML e ligou em seu nome. AWS STS

Noções básicas sobre o acesso SAML

- Os usuários corporativos recebem AWS credenciais temporárias dos grupos de identidade do Amazon Cognito após se autenticarem com seu provedor de SAML.
- Cada usuário autenticado recebe um ID de identidade exclusivo que persiste em todas as sessões.

- Essas credenciais estão vinculadas a um perfil do IAM projetado especificamente para acesso autenticado, fornecendo permissões mais amplas do que o acesso de convidado.
- As asserções do SAML são trocadas por AWS credenciais, mantendo a identidade do usuário e os atributos da empresa.

Autenticação OpenID Connect (OIDC)

Esta etapa explora a autenticação OIDC com provedores de identidades corporativos. Os usuários podem fazer login por meio do provedor de identidade corporativa da organização (como Azure AD, Okta ou Google Workspace) para acessar AWS os serviços. Continue a leitura se quiser informações sobre como integrar a autenticação baseada em padrões aos recursos da AWS . Após implementar a autenticação OIDC, você aprenderá como aproveitar as declarações OIDC para um controle de acesso refinado.

OIDC authentication with enhanced flow

Esta seção mostra como você pode usar um provedor de identidade do OIDC para cadastrar um usuário e, usando o fluxo aprimorado, trocar o token do OIDC em um pool de identidade do Amazon Cognito por credenciais temporárias para chamar serviços. AWS AWS

Usar o login OIDC com o fluxo aprimorado do banco de identidades

1. Na aplicação de demonstração, navegue até a seção Demonstração interativa.
2. Selecione a guia Acesso autenticado.
3. Selecione a guia Fluxo aprimorado.
4. Selecione Fazer login com o provedor OIDC.
5. Conclua o login com suas credenciais corporativas.
6. O provedor OIDC redireciona de volta à aplicação com um código de autorização.
7. A aplicação troca o código de autorização com o grupo de usuários por tokens web JSON.
8. O aplicativo envia o token OIDC para seu grupo de identidades e recupera credenciais temporárias. AWS
9. A aplicação exibe o painel Resultados na interface da web.

   Após a autenticação bem-sucedida, você verá a interface da web exibindo o painel Resultados e terá duas opções para explorá-lo:

a. Botão Exibir somente credenciais: escolha esse botão se quiser ver diretamente AWS as credenciais temporárias geradas sem os detalhes do fluxo da API.

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
    "Expiration": "2025-08-12T13:36:17-07:00"
  }
}
```

b. Exibir botão detalhado do fluxo da API: escolha esse botão se quiser ver as solicitações da step-by-step API.

- A aplicação conecta o usuário com um IdP OIDC e obtém o token de ID. Os bancos de identidades aceitam tokens OIDC de provedores corporativos:

Provedor de identidades	Chave do provedor do Cognito	Finalidade
Provedor OIDC	`example-provider.com/oauth2/default`	Tokens de ID OIDC da empresa IdPs

  Após a autenticação bem-sucedida com o provedor do OIDC, seu aplicativo recebe uma resposta OAuth 2.0 contendo os tokens:

```
{
  "token_type": "Bearer",
  "expires_in": 3600,
  "access_token": "eyJraWQiOiJFWAMPLE......",
  "scope": "email openid profile",
  "id_token": "eyJraWQiOiJFWAMPLE......"
}
```

- Solicitação de API `GetId()` com o ID do banco de identidades e um mapa `Logins` contendo o token do provedor OIDC.

```
POST GetId
{
  "AccountId": "111122223333",
  "IdentityPoolId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Logins": {
    "example-provider.com/oauth2/default": "eyJraWQiOiJFWAMPLE......"
  }
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

- `GetCredentialsForIdentity()` com o `IdentityID` retornado e o mesmo mapa Logins.

```
POST GetCredentialsForIdentity
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Logins": {
    "example-provider.com/oauth2/default": "eyJraWQiOiJFWAMPLE......"
  }
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......"
  }
}
```

O que aconteceu: o Amazon Cognito validou o token OIDC em relação ao provedor configurado, escolheu uma função do IAM (padrão, baseada em declarações ou mapeada por regras) e ligou em seu nome. AWS STS

OIDC authentication with basic flow

Esta seção mostra como você pode usar um provedor de identidade do OIDC para cadastrar um usuário e, usando o fluxo básico, trocar o token do OIDC em um pool de identidade do Amazon Cognito por credenciais temporárias para chamar serviços. AWS AWS

Usar o login OIDC com o fluxo básico do banco de identidades

1. Na aplicação de demonstração, navegue até a seção Demonstração interativa.
2. Selecione a guia Acesso autenticado.
3. Selecione a guia Fluxo básico.
4. Selecione Fazer login com o provedor OIDC.
5. Conclua o login com suas credenciais corporativas.
6. O provedor OIDC redireciona de volta à aplicação com um código de autorização.
7. A aplicação troca o código de autorização com o grupo de usuários por tokens web JSON.
8. O aplicativo envia o token OIDC para seu grupo de identidades e recupera credenciais temporárias. AWS
9. A aplicação exibe o painel Resultados na interface da web.

   Após a autenticação bem-sucedida, você verá a interface da web exibindo o painel Resultados e terá duas opções para explorá-lo:

   a. Botão Exibir somente credenciais: escolha esse botão se quiser ver diretamente AWS as credenciais temporárias geradas sem os detalhes do fluxo da API.

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "IQoJb3JpZ2luX2VjEEXAMPLE......",
    "Expiration": "2025-08-12T13:36:17-07:00"
  }
}
```

b. Exibir botão detalhado do fluxo da API: escolha esse botão se quiser ver as solicitações da step-by-step API.

- A aplicação conecta o usuário com um IdP OIDC e obtém o token de ID. Os bancos de identidades aceitam tokens OIDC de provedores corporativos:

Provedor de identidades	Chave do provedor do Cognito	Finalidade
Provedor OIDC	`example-provider.com/oauth2/default`	Tokens de ID OIDC da empresa IdPs

Após a autenticação bem-sucedida com o provedor do OIDC, seu aplicativo recebe uma resposta OAuth 2.0 contendo os tokens:

```
{
  "token_type": "Bearer",
  "expires_in": 3600,
  "access_token": "eyJraWQiOiJFWAMPLE......",
  "scope": "openid email profile",
  "id_token": "eyJraWQiOiJFWAMPLE......"
}
```

- Solicitação de API `GetId()` com o ID do banco de identidades e um mapa `Logins` contendo o token do provedor OIDC.

```
POST GetId
{
  "IdentityPoolId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "Logins": {
    "example-provider.com/oauth2/default": "eyJraWQiOiJFWAMPLE......"
  }
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222"
}
```

- `GetOpenIdToken()` com o IdentityID retornado e o mesmo mapa `Logins`.

```
POST GetOpenIdToken
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Logins": {
    "example-provider.com/oauth2/default": "eyJraWQiOiJFWAMPLE......"
  }
}
```

Resposta:

```
{
  "IdentityId": "us-east-1:a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "Token": "eyJraWQiOiJFWAMPLE......"
}
```

- `AssumeRoleWithWebIdentity()` com o token OpenID

```
POST AssumeRoleWithWebIdentity
{
  "RoleArn": "arn:aws:iam::111122223333:role/Cognito_IdentityPoolAuth_Role",
  "WebIdentityToken": "eyJraWQiOiJFWAMPLE......"
}
```

Resposta:

```
{
  "Credentials": {
    "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
    "SessionToken": "FwoGZXIvYXdzEEXAMPLE......",
    "Expiration": "2025-08-12T14:36:17-07:00"
  }
}
```

O que aconteceu: o Amazon Cognito validou o token OIDC em relação ao provedor configurado e retornou um token OpenID. O aplicativo ligou AWS STS diretamente para assumir a função apropriada do IAM e recebeu credenciais de curta duração.

Noções básicas sobre a autenticação OIDC

- Baseado em padrões: o OIDC é baseado em OAuth 2.0 e fornece informações de identidade padronizadas.
- Validação de tokens: os tokens de ID podem ser validados quanto à autenticidade.
- Acesso baseado em declarações: as declarações OIDC podem ser usadas para mapeamento de perfis e controle de acesso.
- Integração corporativa: funciona com provedores de identidades corporativos populares.

Próximas etapas

Agora que você configurou e explorou a aplicação de demonstração, você pode:

- Configurar provedores de identidades adicionais que você ainda não testou.
- Experimentar a autenticação avançada e básica para entender suas diferenças.
- Personalizar a demonstração para seu próprio caso de uso.
- Integrar os bancos de identidades do Amazon Cognito em suas próprias aplicações.

Opções de configuração guiada para o Amazon Cognito

Você pode querer avaliar os recursos do Amazon Cognito em uma experiência estruturada e guiada. Aqui estão alguns recursos externos que fornecem experiências personalizadas com grupos de usuários e bancos de identidades.

Concluir um workshop

O AWS workshop studio hospeda um workshop que orienta você pela configuração da maioria dos recursos do Amazon Cognito. Esses recursos incluem a API de grupos de usuários, a interface de usuário hospedada dos grupos de usuários, os bancos de identidades e as configurações de segurança.

Adicionar o código da aplicação a partir de exemplos

O capítulo sobre exemplos de código deste guia tem um código de aplicação para você usar com os grupos de usuários e os bancos de identidades. A seção de grupos de usuários do capítulo de exemplos de código contém trechos que abrangem operações individuais e exemplos mais longos de aplicações completos em diversas linguagens de programação.

Criar uma aplicação full-stack com o AWS Amplify

O AWS Amplify é um AWS service (Serviço da AWS) para desenvolvedores que desejam desenvolver e hospedar uma aplicação e uma interface de usuário. O Amazon Cognito é o componente de autenticação do Amplify. Quando você adiciona autenticação à sua aplicação, o Amplify pode automatizar a implantação dos recursos do grupo de usuários e do banco de identidades do Amazon Cognito. Consulte também Integração da autenticação e autorização do Amazon Cognito com aplicações móveis e da web.

Mais recursos da aplicação Amazon Cognito no GitHub

- Exemplos de fluxo de autenticação com o .NET para Amazon Cognito
- Autenticação sem senha do Amazon Cognito
- Exemplo de PetStore com Amazon Verified Permissions
- Exemplo de aplicação React usando ABAC + bancos de identidades para acessar recursos do AWS
- Autorização máquina a máquina baseada no Amazon Cognito e no API Gateway usando o AWS CDK

- Criação de autorização refinada usando o Amazon Cognito, o API Gateway e o IAM
- CloudFront authorization@edge

Mais workshops

- Implementar a autenticação sem senha com o Amazon Cognito e o WebAuthn
- Workshop do Amazon Cognito
- Workshop de solução de problemas do Amazon Cognito
- Autenticação e autorização com o Amazon Cognito e o Verified Permissions
- Análise detalhada de JWT do Amazon Cognito

Postagens do blog

- Proteja clientes públicos do Amazon Cognito usando um proxy do Amazon CloudFront
- Como configurar o Amazon Cognito para autenticação federada usando o Azure AD
- Simplifique a autenticação de aplicações web: um guia para a federação do AD FS com grupos de usuários do Amazon Cognito

Integração da autenticação e autorização do Amazon Cognito com aplicações móveis e da web

A implementação do Amazon Cognito é uma combinação de ferramentas administrativas de Console de gerenciamento da AWS nosso AWS SDK e bibliotecas de SDK em aplicativos. O console do Amazon Cognito é a interface visual para configuração e gerenciamento dos grupos de usuários e bancos de identidades do Amazon Cognito.

A integração de menor esforço que você pode criar com grupos de usuários do Amazon Cognito é com o login gerenciado. O login gerenciado é um aplicativo de login ready-to-use baseado na web para testes e implantação rápidos de grupos de usuários do Amazon Cognito. A autenticação do grupo de usuários com login gerenciado requer bibliotecas do OpenID Connect (OIDC) que direcionam os usuários para páginas de login hospedado. Nessa série de endpoints web interativos e de redirecionamento, o Amazon Cognito gerencia o fluxo de autenticação, incluindo login de terceiros, autenticação multifator (MFA) e a escolha de um fluxo de autenticação. Sua aplicação só precisa processar o resultado da autenticação retornado pelo Amazon Cognito na resposta.

Você também pode adicionar um AWS SDK ao seu aplicativo, criar interfaces de autenticação personalizadas e invocar operações de API para autenticação e autorização de seus usuários. AWS Amplifyé uma ferramenta AWS service (Serviço da AWS) para criar aplicativos completos, com a autenticação do Amazon Cognito no back-end.

Por exemplo, a aplicação pode invocar o login gerenciado para o login do usuário e, depois, chamar o endpoint do token pelo código da aplicação a fim de trocar o código de autorização do usuário por tokens. Depois, a aplicação deve interpretar e armazenar os tokens do usuário e apresentá-los no contexto apropriado para autenticação e autorização. O Amplify adiciona ferramentas de integração guiada com funções integradas para esses processos.

Você também pode criar recursos do Amazon Cognito inteiramente em código. Os grupos de identidades não têm as mesmas opções de autenticação gerenciada dos grupos de usuários. Para acessar as AWS credenciais em seus aplicativos, implemente operações de grupos de identidades em módulos SDK importados. Para começar a usar seu próprio código de aplicativo personalizado, visite os exemplos de código do Amazon Cognito para. AWS SDKs Para integração com o Amazon Cognito como um provedor de identidades do OpenID Connect, use Ferramentas para desenvolvedores do OpenID Connect.

Antes de usar a autenticação e a autorização do Amazon Cognito, escolha uma plataforma de aplicações e prepare seu código para se integrar ao serviço. Para obter as plataformas disponíveis para AWS SDKs, consulteAutenticação com AWS SDKs. AWS CLI É um SDK de linha de comando para o Amazon Cognito e outros Serviços da AWS, e é um lugar valioso para começar a se familiarizar com as operações da API do Amazon Cognito e sua sintaxe.

> **Note**
>
> Alguns componentes do Amazon Cognito só podem ser configurados com a API. Por exemplo, você só pode definir um gatilho Lambda personalizado de SMS ou remetente de e-mail para um grupo de usuários com uma solicitação que atualize `LambdaConfig` a propriedade UserPoolda classe em CreateUserPooluma UpdateUserPoolsolicitação de API.

A API de grupos de usuários do Amazon Cognito compartilha seu namespace com várias classes de operações de API. Uma classe configura grupos de usuários e seus processos, provedores de identidades e usuários. Outra inclui operações não autenticadas para que os usuários em um cliente público façam login, saiam e gerenciem seus perfis. A classe final de operações de API executa operações de usuário que você autoriza com suas próprias AWS credenciais em um cliente confidencial do lado do servidor. Você deve conhecer a arquitetura da aplicação pretendida antes de começar a implementar o código dela. Para obter mais informações, consulte Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado.

Tópicos

- Autenticação com AWS Amplify
- Autenticação com AWS SDKs
- Como funciona a autenticação com o Amazon Cognito
- Usando esse serviço com um AWS SDK
- Autorização com o Amazon Verified Permissions

Autenticação com AWS Amplify

AWS Amplify é uma solução completa para criar aplicativos web e móveis. Com o Amplify, você pode se conectar aos recursos existentes com as bibliotecas do Amplify ou criar e configurar recursos com a interface da linha de comando (CLI) do Amplify. O Amplify também tem componentes de interface

de usuário conectados, como Autenticador para configuração e personalização da experiência de login e inscrição na aplicação.

Para usar os atributos de autenticação do Amplify na aplicação de front-end, consulte a documentação a seguir por plataforma.

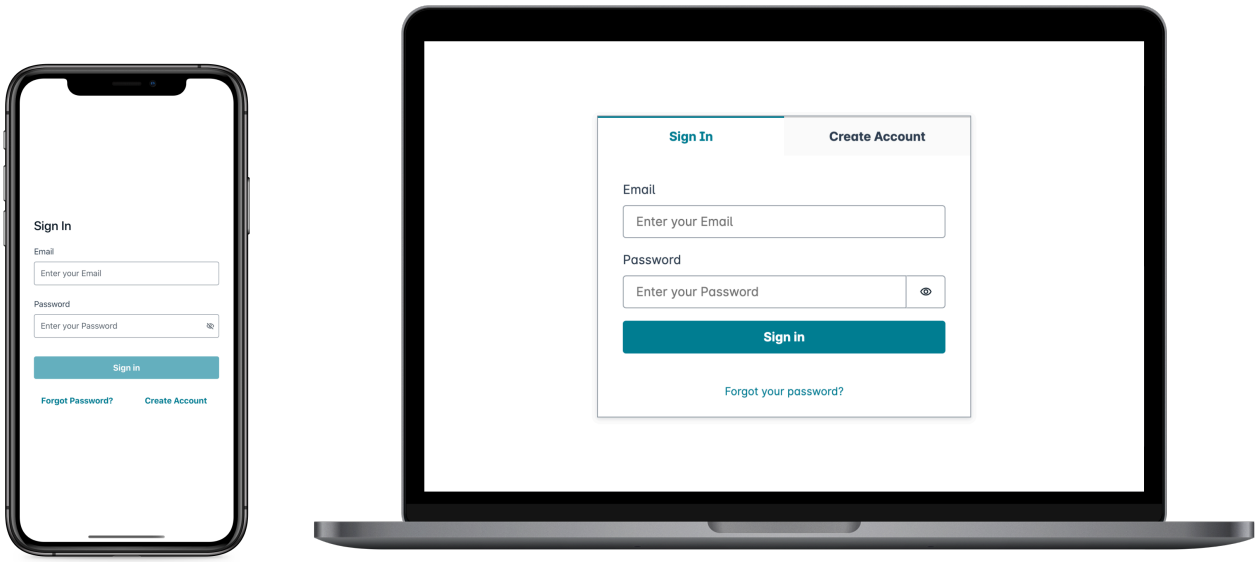
- Autenticação do Amplify para React
- Autenticação do Amplify para React Native
- Autenticação do Amplify para Swift (iOS)
- Autenticação do Amplify para Android
- Autenticação do Amplify para Flutter

As bibliotecas do Amplify são de código aberto e estão disponíveis em. GitHub Para saber mais sobre como o Amplify Auth implementa a autenticação do Amazon Cognito, acesse as seguintes bibliotecas:

- amplify-js
- amplify-swift
- amplify-flutter
- amplify-android

Criar uma interface de usuário (UI) com o Amplify

O Login gerenciado do grupo de usuários pode atender às necessidades essenciais de um frontend de autenticação para aplicações web ou aplicativos móveis. Para personalizar a interface de usuário (IU) além dos parâmetros que o login gerenciado acomoda, crie uma aplicação personalizada. Amplify UI é uma coleção personalizável de componentes de front-end em vários idiomas.



Para começar a usar o componente de autenticação personalizado, acesse a documentação a seguir para o componente Autenticador.

- Autenticador para Android
- Autenticador para Angular
- Autenticador para Flutter
- Autenticador para React
- Autenticador para React Native
- Autenticador para Swift
- Autenticador para Vue

Autenticação com AWS SDKs

Para usar um back-end seguro para criar seu próprio microsserviço de identidade que interage com o Amazon Cognito, conecte-se aos grupos de usuários do Amazon Cognito e à API de grupos de identidade do Amazon Cognito com AWS um SDK no idioma de sua escolha.

Para obter detalhes sobre cada operação de API, consulte a Referência da API de grupos de usuários do Amazon Cognito e a Referência da API do Amazon Cognito. Esses documentos contêm. Consulte também seções com recursos para usar uma variedade SDKs de plataformas suportadas.

- AWS Command Line Interface
- AWS SDK para .NET
- AWS SDK para C++
- AWS SDK para Go
- AWS SDK para Java V2
- AWS SDK para JavaScript
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

Como funciona a autenticação com o Amazon Cognito

Quando seu cliente faz login em um grupo de usuários do Amazon Cognito, seu aplicativo recebe tokens web JSON (). JWTs

Quando seu cliente faz login em um grupo de identidades, seja com um token de grupo de usuários ou outro provedor, seu aplicativo recebe AWS credenciais temporárias.

Com o login do grupo de usuários, você pode implementar a autenticação e a autorização inteiramente com um SDK da AWS . Se você não quiser criar seus próprios componentes de interface de usuário (IU), pode invocar uma interface web previamente criada (login gerenciado) ou a página de login do seu provedor de identidades (IdP) de terceiros.

Este tópico é uma visão geral de algumas das maneiras pelas quais seu aplicativo pode interagir com o Amazon Cognito para se autenticar com tokens de ID, autorizar com tokens de acesso e acessar com credenciais do grupo de identidades Serviços da AWS .

Tópicos

- Autenticação de grupo de usuários com login gerenciado
- Autenticação e autorização da API do grupo de usuários com um AWS SDK
- Autenticação do grupo de usuários com um provedor de identidades de terceiros

- Autenticação do banco de identidades

Autenticação de grupo de usuários com login gerenciado

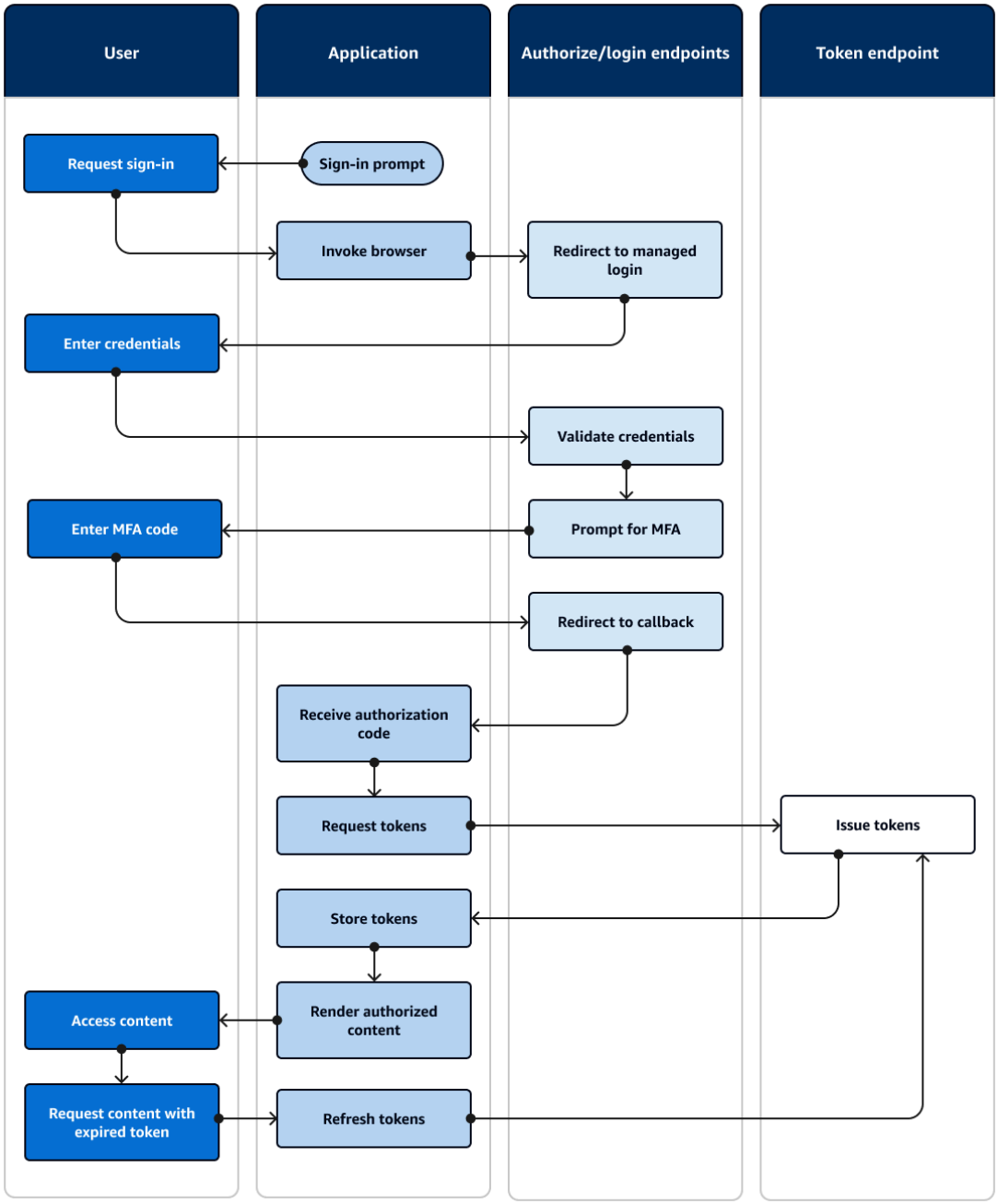
O login gerenciado é um site vinculado ao grupo de usuários e ao cliente da aplicação. Ele pode realizar operações de login, cadastro e redefinição de senha para seus usuários. Uma aplicação com componente de login gerenciado para autenticação pode exigir menos trabalho do desenvolvedor ao fazer a implementação. Uma aplicação pode ignorar a autenticação de componentes da IU e invocar páginas da web de login gerenciado no navegador do usuário.

Os aplicativos coletam os usuários JWTs com um local de redirecionamento da web ou do aplicativo. As aplicações que implementam o login gerenciado podem se conectar a grupos de usuários para autenticação como se fossem um IdP OpenID Connect (OIDC).

O login gerenciado se enquadra no modelo no qual as aplicações exigem os serviços de autenticação de um servidor de autorização OIDC, mas não necessitam imediatamente de recursos como autenticação personalizada, integração de bancos de identidades ou autoatendimento de recursos do usuário. Para usar algumas dessas opções avançadas, você pode implementá-las com um componente de grupos de usuários de um SDK.

Os modelos gerenciados de login e autenticação de IdP de terceiros, com uma dependência primária da implementação do OIDC, são os melhores para modelos de autorização avançados com escopos 2.0. OAuth

O diagrama a seguir ilustra uma sessão de login típica para autenticação de login gerenciado.



Fluxo de autenticação de login gerenciado

1. Um usuário acessa sua aplicação.
2. Ele seleciona um link “Fazer login”.
3. A aplicação direciona o usuário para uma solicitação de login nas páginas de login gerenciado do domínio do grupo de usuários.
4. Ele insere nome de usuário e senha.

5. O grupo de usuários valida as credenciais do usuário e determina que o usuário ativou autenticação multifator (MFA).
6. A página de login gerenciado solicita que o usuário insira um código de MFA.
7. O usuário insere seu código de MFA.
8. O grupo de usuários redireciona o usuário para o URL da aplicação.
9. A aplicação coleta o código de autorização do parâmetro de solicitação de URL que o login gerenciado anexou ao URL de retorno de chamada.
10. A aplicação solicita tokens com o código de autorização.
11. O endpoint do token retorna JWTs ao aplicativo.
12. O aplicativo decodifica, valida e armazena ou armazena em cache os dados do usuário. JWTs
13. A aplicação exibe o componente de controle de acesso solicitado.
14. O usuário visualiza seu conteúdo.
15. Depois, o token de acesso do usuário expira e ele solicita a visualização de um componente de acesso controlado.
16. A aplicação determina que a sessão do usuário deve persistir. Ele solicita novos tokens do endpoint do token usando o token de atualização.

Variantes e personalização

Você pode personalizar a aparência de suas páginas de login gerenciado com o editor de identidade visual para todo o grupo de usuários ou no nível de qualquer cliente da aplicação. Você também pode configurar aplicações clientes com seus próprios provedores de identidade, escopos, acesso aos atributos do usuário e configuração de segurança avançada.

Recursos relacionados

- Login gerenciado do grupo de usuários
- Escopos, M2M e servidores de recursos
- Referência de login gerenciado e endpoints do grupo de usuários

Autenticação e autorização da API do grupo de usuários com um AWS SDK

AWS desenvolveu componentes para grupos de usuários do Amazon Cognito, ou provedor de identidade do Amazon Cognito, em uma variedade de estruturas de desenvolvedores. Os métodos

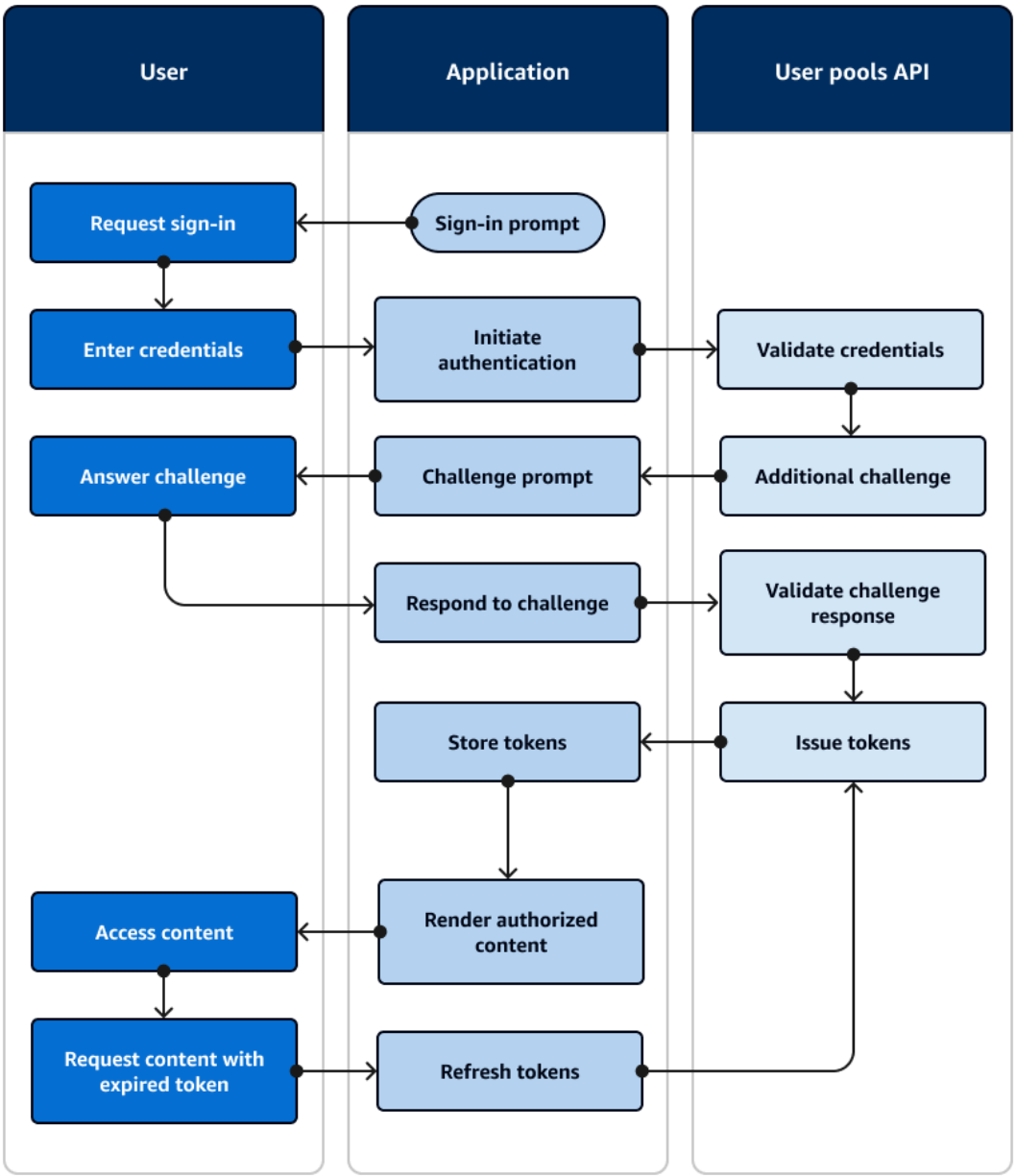
incorporados a eles SDKs chamam a API de grupos de usuários do Amazon Cognito. O mesmo namespace da API de grupos de usuários tem operações para configuração de grupos de usuários e para autenticação de usuários. Para ter uma visão geral mais completa, consulte Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado.

A autenticação de API se ajusta ao modelo em que suas aplicações têm componentes de interface de usuário existentes e dependem principalmente do grupo de usuários como um diretório de usuários. Esse design adiciona o Amazon Cognito como um componente dentro de uma aplicação maior. Ele exige lógica programática para lidar com conjuntos complexos de desafios e respostas.

Essa aplicação não precisa fazer uma implementação completa de terceiros que usam OpenID Connect (OIDC). Em vez disso, ele tem a capacidade de decodificar e usar JWTs. Quando você quiser acessar o conjunto completo de recursos do grupo de usuários para usuários locais, crie a autenticação com o SDK do Amazon Cognito em seu ambiente de desenvolvimento.

A autenticação de API com OAuth escopos personalizados é menos orientada para a autorização de API externa. Para adicionar escopos personalizados a um token de acesso a partir da autenticação da API, modifique o token em runtime com um Acionador do Lambda antes da geração do token.

O diagrama a seguir ilustra uma sessão de login típica para autenticação da API.



Fluxo de autenticação da API

1. Um usuário acessa sua aplicação.
2. Ele seleciona um link “Fazer login”.
3. Ele insere nome de usuário e senha.
4. O aplicativo invoca o método que faz uma solicitação de InitiateAuthAPI. A solicitação passa as credenciais do usuário para um grupo de usuários.

5. O grupo de usuários valida as credenciais do usuário e determina que o usuário ativou autenticação multifator (MFA).
6. O grupo de usuários responde com um desafio que solicita um código de MFA.
7. A aplicação gera um prompt que coleta o código MFA do usuário.
8. O aplicativo invoca o método que faz uma solicitação de RespondToAuthChallengeAPI. A solicitação passa o código MFA do usuário.
9. O grupo de usuários valida o código MFA do usuário.
10. O grupo de usuários responde com o do JWTs usuário.
11. O aplicativo decodifica, valida e armazena ou armazena em cache os dados do usuário. JWTs
12. A aplicação exibe o componente de controle de acesso solicitado.
13. O usuário visualiza seu conteúdo.
14. Depois, o token de acesso do usuário expira e ele solicita a visualização de um componente de acesso controlado.
15. A aplicação determina que a sessão do usuário deve persistir. Ele invoca o InitiateAuthmétodo novamente com o token de atualização e recupera novos tokens.

Variantes e personalização

Você pode ampliar esse fluxo com desafios adicionais, por exemplo, seus próprios desafios de autenticação personalizados. Você pode restringir automaticamente o acesso de usuários cujas senhas foram comprometidas ou cujas características de login inesperadas indiquem uma tentativa de login mal-intencionada. Esse fluxo é praticamente o mesmo para operações de cadastro, atualização de atributos de usuário e redefinição de senhas. A maioria desses fluxos tem operações de API públicas (do lado do cliente) e confidenciais (do lado do servidor) duplicadas.

Recursos relacionados

- API de grupos de usuários do Amazon Cognito
- Conceitos básicos dos grupos de usuários
- Integração da autenticação e autorização do Amazon Cognito com aplicações móveis e da web
- Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado

Autenticação do grupo de usuários com um provedor de identidades de terceiros

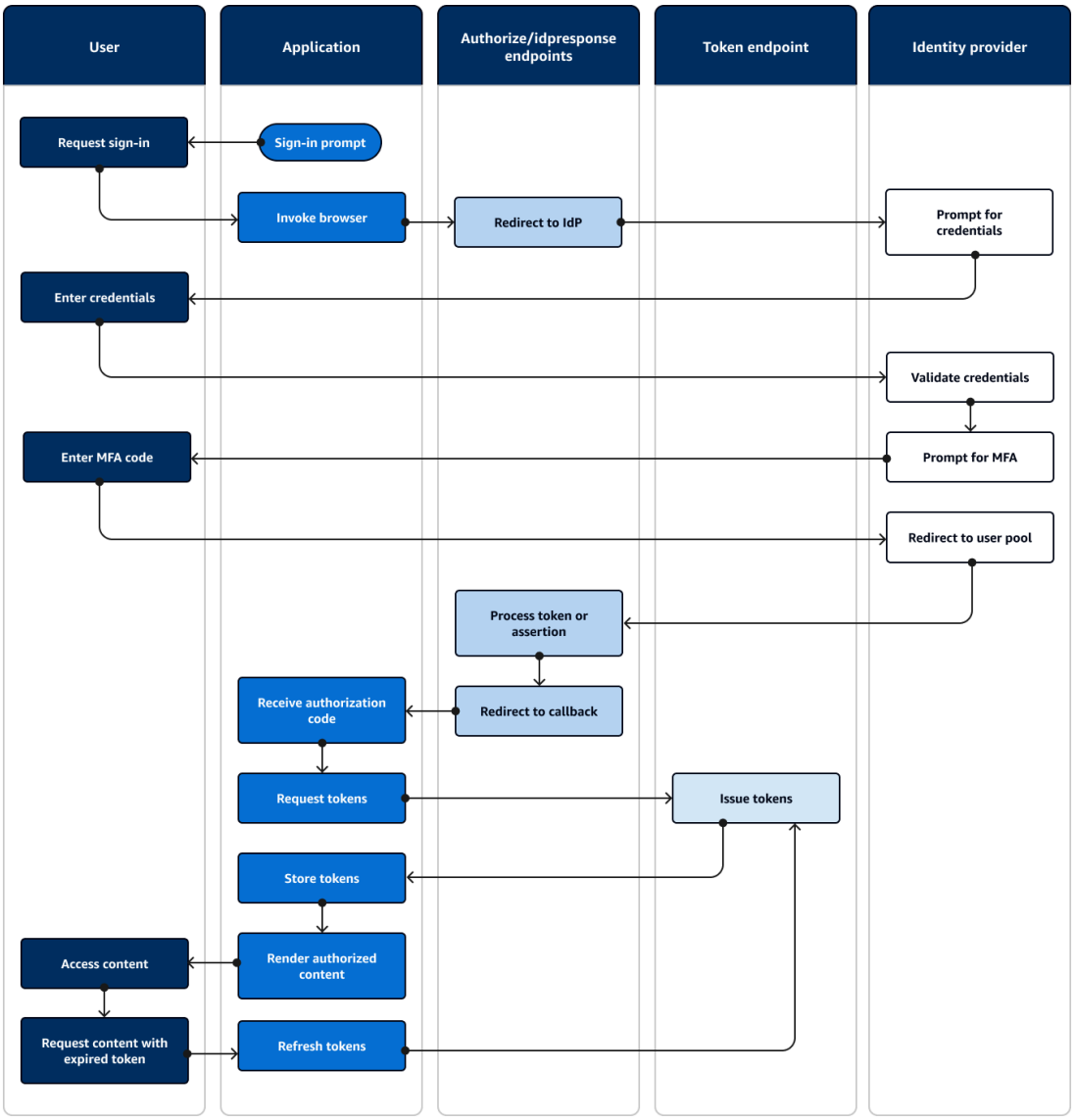
O login com um provedor de identidades (IdP) externo, ou autenticação federada, é um modelo semelhante ao login gerenciado. Sua aplicação é uma parte dependente do OIDC em seu grupo de usuários, enquanto o grupo de usuários serve de passagem para um IdP. O IdP pode ser um diretório de usuários consumidores, como Facebook ou Google, ou pode ser um diretório corporativo SAML 2.0 ou OIDC, como o Azure.

Em vez de login gerenciado no navegador do usuário, sua aplicação invoca um endpoint de redirecionamento no servidor de autorização do grupo de usuários. Do ponto de vista do usuário, ele escolhe o botão de login na aplicação. Em seguida, o IdP solicita que ele faça login. Assim como na autenticação de login gerenciada, um aplicativo coleta JWTs em um local de redirecionamento no aplicativo.

A autenticação com um IdP de terceiros se enquadra em um modelo em que os usuários podem não querer criar uma senha ao se inscreverem na sua aplicação. É possível adicionar facilmente a autenticação de terceiros a uma aplicação que implementou a autenticação de login gerenciado. Na verdade, o login gerenciado e o login de terceiros IdPs produzem um resultado de autenticação consistente a partir de pequenas variações no que você invoca nos navegadores dos usuários.

Assim como a autenticação de login gerenciada, a autenticação federada é melhor para modelos de autorização avançados com OAuth escopos 2.0.

O diagrama a seguir ilustra uma sessão de login típica para autenticação federada.



Fluxo de autenticação federada

1. Um usuário acessa sua aplicação.
2. Ele seleciona um link “Fazer login”.
3. A aplicação direciona o usuário para um prompt de login usando seu IdP.
4. Ele insere nome de usuário e senha.
5. O IdP valida as credenciais do usuário e determina que o usuário ativou autenticação multifator (MFA).
6. O IdP solicita que o usuário insira um código de MFA.
7. O usuário insere seu código de MFA.

8. O IdP redireciona o usuário para o grupo de usuários com uma resposta SAML ou um código de autorização.
9. Se o usuário tiver passado um código de autorização, o grupo de usuários fará uma troca silenciosa do código por tokens IdP. O grupo de usuários valida os tokens IdP e redireciona o usuário para a aplicação com um novo código de autorização.
10. A aplicação coleta o código de autorização do parâmetro de solicitação de URL que o grupo de usuários anexou ao URL de retorno de chamada.
11. A aplicação solicita tokens com o código de autorização.
12. O endpoint do token retorna JWTs ao aplicativo.
13. O aplicativo decodifica, valida e armazena ou armazena em cache os dados do usuário. JWTs
14. A aplicação exibe o componente de controle de acesso solicitado.
15. O usuário visualiza seu conteúdo.
16. Depois, o token de acesso do usuário expira e ele solicita a visualização de um componente de acesso controlado.
17. A aplicação determina que a sessão do usuário deve persistir. Ele solicita novos tokens do endpoint do token usando o token de atualização.

Variantes e personalização

Você pode iniciar a autenticação federada no login gerenciado, onde os usuários podem escolher em uma lista das IdPs que você atribuiu ao seu cliente de aplicativo. O login gerenciado também pode solicitar um endereço de e-mail e encaminhar automaticamente a solicitação de um usuário para o IdP SAML correspondente. A autenticação com um provedor de identidades de terceiros não exige interação do usuário com o login gerenciado. Sua aplicação pode adicionar um parâmetro de solicitação à solicitação do servidor de autorização do usuário e fazer com que o usuário seja redirecionado silenciosamente para a página de login do IdP.

Recursos relacionados

- Login do grupo de usuários com provedores de identidades de terceiros
- Escopos, M2M e servidores de recursos
- Referência de login gerenciado e endpoints do grupo de usuários

Autenticação do banco de identidades

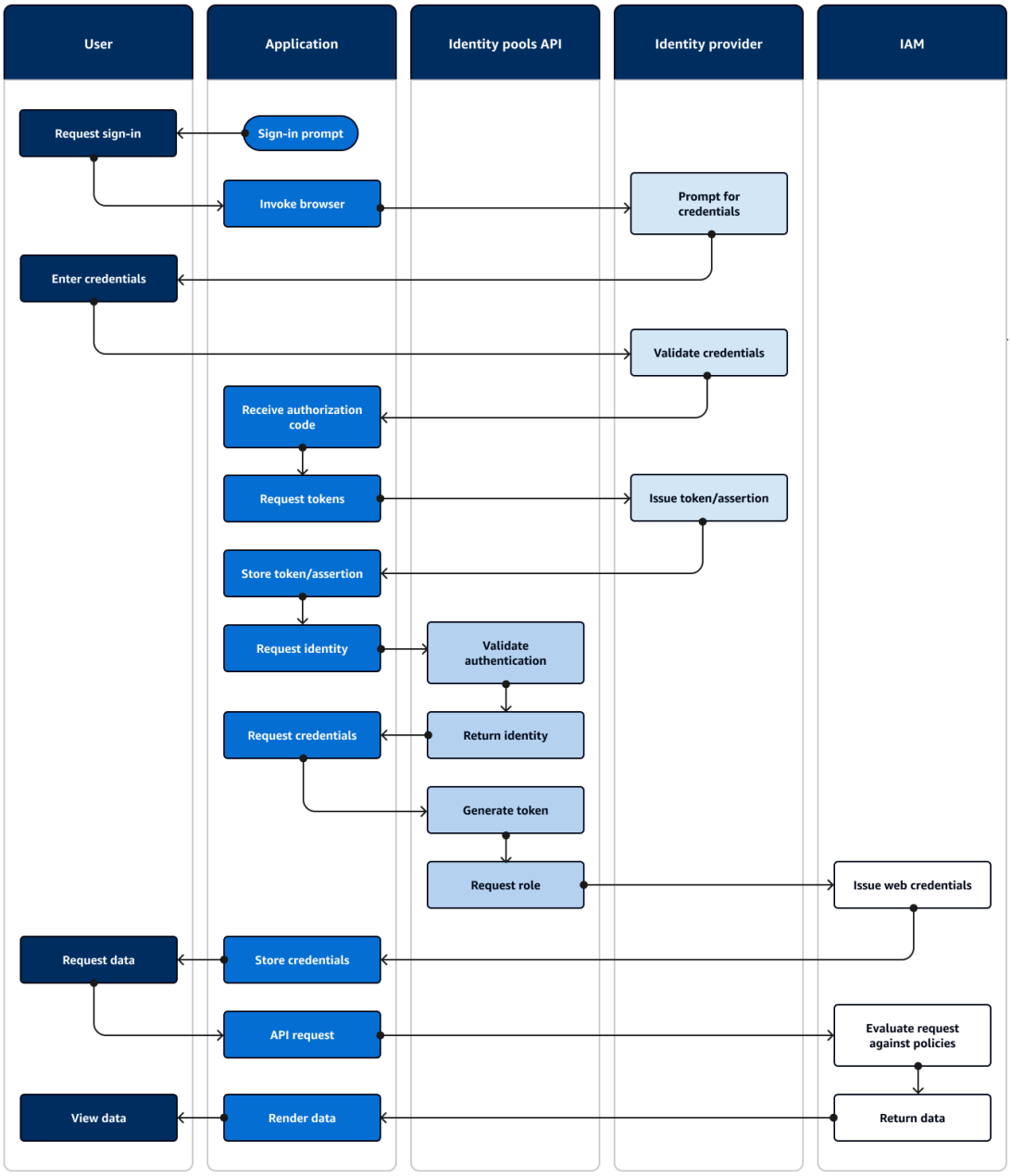
Um banco de identidades é um componente da aplicação que é diferente de um grupo de usuários em função, namespace de API e modelo de SDK. Onde os grupos de usuários oferecem autenticação e autorização baseadas em tokens, os grupos de identidades oferecem autorização para AWS Identity and Access Management (IAM).

Você pode atribuir um conjunto de grupos IdPs de identidades e fazer login de usuários com eles. Os grupos de usuários são estreitamente integrados como grupos de identidades IdPs e oferecem aos grupos de identidades o máximo de opções para controle de acesso. Ao mesmo tempo, há uma ampla variedade de opções de autenticação para bancos de identidades. Os grupos de usuários unem fontes de identidade SAML, OIDC, sociais, de desenvolvedores e convidados como rotas para AWS credenciais temporárias de grupos de identidades.

A autenticação com um banco de identidades é externa. Ela segue um dos fluxos do grupo de usuários ilustrados anteriormente ou um fluxo que você desenvolve de forma independente com outro IdP. Depois que a aplicação ativa faz a autenticação inicial, ela passa a prova para um banco de identidades e recebe uma sessão temporária em troca.

A autenticação com um grupo de identidades se encaixa em um modelo em que você impõe o controle de acesso aos ativos e dados do aplicativo Serviços da AWS com a autorização do IAM. Assim como acontece com a autenticação de API em grupos de usuários, um aplicativo bem-sucedido inclui AWS SDKs cada um dos serviços que você deseja acessar para o benefício de seus usuários. AWS SDKs aplique as credenciais da autenticação do grupo de identidades como assinaturas às solicitações de API.

O diagrama a seguir ilustra uma sessão de login típica para autenticação do banco de identidades com o uso de um IdP.



Fluxo de autenticação do banco de identidades

1. Um usuário acessa sua aplicação.
2. Ele seleciona um link “Fazer login”.
3. A aplicação direciona o usuário para um prompt de login usando seu IdP.
4. Ele insere nome de usuário e senha.
5. O IdP valida as credenciais do usuário.

6. O IdP redireciona o usuário para a aplicação com uma resposta SAML ou um código de autorização.
7. Se o usuário tiver passado um código de autorização, a aplicação fará uma troca do código por tokens IdP.
8. O aplicativo decodifica, valida e armazena ou armazena em cache a declaração do JWTs usuário.
9. O aplicativo invoca o método que faz uma solicitação de GetIdAPI. Ele passa o token ou a declaração do usuário e solicita um ID da identidade.
10. O banco de identidades valida o token ou a declaração de acordo com os provedores de identidade configurados.
11. O banco de identidades retorna um ID de identidade.
12. O aplicativo invoca o método que faz uma solicitação de GetCredentialsForIdentityAPI. Ele passa o token ou as declarações do usuário e solicita um perfil do IAM.
13. O banco de identidades gera um novo JWT. O novo JWT contém declarações que solicitam um perfil do IAM. O banco de identidades determina o perfil com base na solicitação do usuário e nos critérios de seleção de função na configuração do banco de identidades para o IdP.
14. AWS Security Token Service (AWS STS) responde à AssumeRoleWithWebIdentitysolicitação do grupo de identidades. A resposta contém credenciais de API para uma sessão temporária com um perfil do IAM.
15. A aplicação armazena as credenciais da sessão.
16. O usuário realiza uma ação na aplicação que requer recursos com acesso protegido na AWS.
17. O aplicativo aplica as credenciais temporárias como assinaturas às solicitações de API para o necessário. Serviços da AWS
18. O IAM avalia as políticas associadas ao perfil nas credenciais. Ele as compara com a solicitação.
19. AWS service (Serviço da AWS) Retorna os dados solicitados.
20. A aplicação renderiza os dados na interface do usuário.
21. O usuário visualiza os dados.

Variantes e personalização

Para visualizar a autenticação com um grupo de usuários, insira uma das visões gerais anteriores do grupo de usuários após a etapa de Emitir token/declaração. A autenticação do desenvolvedor substitui todas as etapas anteriores a Solicitar identidade por uma solicitação assinada pelas

credenciais do desenvolvedor. A autenticação de convidados também vai direto para Solicitar identidade, não valida a autenticação e retorna as credenciais para um perfil do IAM de acesso limitado.

Recursos relacionados

- Banco de identidades do Amazon Cognito
- Funções do IAM do usuário
- Fluxo de autenticação dos bancos de identidades

Usando esse serviço com um AWS SDK

AWS kits de desenvolvimento de software (SDKs) estão disponíveis para muitas linguagens de programação populares. Cada SDK fornece uma API, exemplos de código e documentação que permitem que os desenvolvedores criem facilmente aplicações em seu idioma de preferência.

Documentação do SDK	Exemplos de código
AWS SDK para C++	AWS SDK para C++ exemplos de código
AWS CLI	AWS CLI exemplos de código
AWS SDK para Go	AWS SDK para Go exemplos de código
AWS SDK para Java	AWS SDK para Java exemplos de código
AWS SDK para JavaScript	AWS SDK para JavaScript exemplos de código
AWS SDK para Kotlin	AWS SDK para Kotlin exemplos de código
AWS SDK para .NET	AWS SDK para .NET exemplos de código
AWS SDK para PHP	AWS SDK para PHP exemplos de código
Ferramentas da AWS para PowerShell	Ferramentas da AWS para PowerShell exemplos de código
AWS SDK para Python (Boto3)	AWS SDK para Python (Boto3) exemplos de código

Documentação do SDK	Exemplos de código
AWS SDK para Ruby	AWS SDK para Ruby exemplos de código
AWS SDK para Rust	AWS SDK para Rust exemplos de código
AWS SDK para SAP ABAP	AWS SDK para SAP ABAP exemplos de código
AWS SDK para Swift	AWS SDK para Swift exemplos de código

> **Exemplo de disponibilidade**
>
> Não consegue encontrar o que precisa? Solicite um exemplo de código usando o link Fornecer feedback na parte inferior desta página.

Autorização com o Amazon Verified Permissions

O Amazon Verified Permissions é um serviço de autorização para as aplicações que você cria. Quando você adiciona um grupo de usuários do Amazon Cognito como uma fonte de identidade, a aplicação pode passar tokens de acesso ou identidade (ID) do grupo de usuários para o Verified Permissions tomar uma decisão de permissão ou negação. O Verified Permissions consideram as propriedades do usuário e o contexto da solicitação com base nas políticas que você escreve na linguagem de política Cedar. O contexto da solicitação pode incluir um identificador para o documento, a imagem ou outro recurso solicitado e a ação que o usuário deseja realizar no recurso.

Seu aplicativo pode fornecer a identidade do usuário ou os tokens de acesso às permissões verificadas IsAuthorizedWithTokenou às solicitações de BatchIsAuthorizedWithTokenAPI. Essas operações de API aceitam seus usuários como `Principal` e tomam decisões de autorização de `Action` sobre o `Resource` que eles desejam acessar. A personalização adicional `Context` pode contribuir para uma decisão de acesso detalhada.

Quando a aplicação apresenta um token em uma solicitação de API `IsAuthorizedWithToken`, o Verified Permissions realiza as validações a seguir.

1. O grupo de usuários é uma fonte de identidade do Verified Permissions configurada para o repositório de políticas solicitado.

2. A reivindicação `client_id` ou `aud`, no token de acesso ou identidade, respectivamente, corresponde a um ID de cliente da aplicação do grupo de usuários que você forneceu ao Verified Permissions. Para verificar essa reivindicação, é necessário configurar a validação do ID do cliente na fonte de identidade do Verified Permissions.
3. O token não expirou.
4. O valor da declaração `token_use` no token corresponde aos parâmetros que você passou para `IsAuthorizedWithToken`. A declaração `token_use` deverá ser `access` se você a tiver passado para o parâmetro `accessToken` e `id` se você a tiver passado para o parâmetro `identityToken`.
5. A assinatura em seu token vem das chaves web JSON publicadas (JWKs) do seu grupo de usuários. Você pode ver seu JWKs em`https://cognito-idp.`*`Region`*`.amazonaws.com/`*`your user pool ID`*`/.well-known/jwks.json`.

Tokens revogados e usuários excluídos

O Verified Permissions valida somente as informações que ele conhece da fonte de identidade e do prazo de expiração do token do usuário. O Verified Permissions não verifica a revogação do token ou a existência do usuário. Se você revogou o token do usuário ou excluiu o perfil do usuário do grupo de usuários, o Verified Permissions considerará o token válido até que ele expire.

Avaliação de políticas

Configure o grupo de usuários como uma fonte de identidade para o repositório de políticas. Configure a aplicação para enviar os tokens de usuários em solicitações ao Verified Permissions. Para cada solicitação, o Verified Permissions compara as reivindicações no token com uma política. Uma política do Verified Permissions é como uma política do IAM na AWS. Ela declara uma entidade principal, um recurso e uma ação. O Verified Permissions responderá à sua solicitação `Allow` se ela corresponder a uma ação permitida e não corresponder a uma ação `Deny` explícita; caso contrário, ele responde com `Deny`. Para obter mais informações, consulte Políticas do Amazon Verified Permissions no Guia do usuário do Amazon Verified Permissions.

Personalização de tokens

Para alterar, adicionar e remover as reivindicações do usuário que você deseja apresentar ao Verified Permissions, personalize o conteúdo em seu acesso e nos tokens de identidade com um Acionador do Lambda antes da geração do token. Com um gatilho de geração de pré-token, é possível adicionar e modificar reivindicações nos tokens. Por exemplo, é possível consultar um banco de dados para obter atributos adicionais do usuário e codificá-los no token de ID.

> **Note**
>
> Devido à forma como o Verified Permissions processa as solicitações, não adicione reivindicações com o nome `cognito`, `dev` ou `custom` na função de geração de pré-token. Quando você apresenta esses prefixos de solicitação reservados não no formato delimitado por dois pontos como `cognito:username`, como nomes completos de reivindicações, as solicitações de autorização falham.

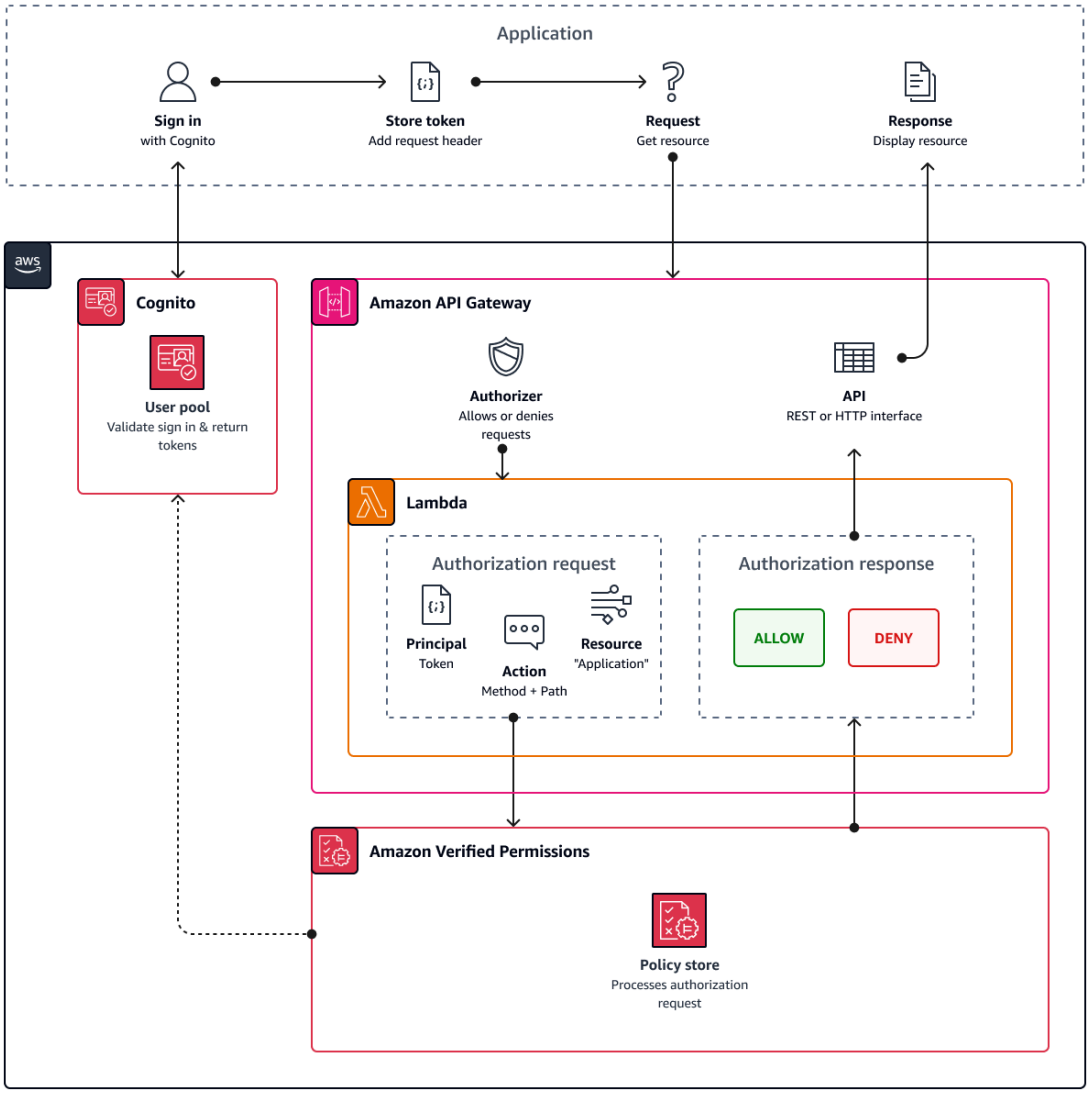
Recursos adicionais do

- Mapeamento de tokens do Amazon Cognito para o esquema do Verified Permissions
- Autorize o API Gateway APIs usando Amazon Verified Permissions e Amazon Cognito
- Workshop: Autenticação e autorização com o Amazon Cognito e o Verified Permissions

Autorização da API com o Verified Permissions

Seu ID ou tokens de acesso podem autorizar solicitações para back-end do Amazon API Gateway REST APIs com permissões verificadas. Você pode criar um repositório de políticas com links imediatos para o grupo de usuários e a API. Com a opção inicial Configurar com o API Gateway e uma fonte de identidade, o Verified Permissions adiciona uma fonte de identidade do grupo de usuários ao repositório de políticas e um autorizador Lambda à API. Quando seu aplicativo passa um token portador do grupo de usuários para a API, o autorizador Lambda invoca o Verified Permissions. O autorizador passa o token como uma entidade principal e o caminho e o método da solicitação como uma ação.

O diagrama a seguir ilustra o fluxo de autorização para uma API do API Gateway com o Verified Permissions. Para ver uma análise detalhada, consulte repositórios de políticas vinculados à API, no Guia do usuário do Amazon Verified Permissions.



O Verified Permissions estrutura a autorização da API em torno de grupos de usuários. Como os tokens de ID e acesso incluem uma `cognito:groups` declaração, seu repositório de políticas pode gerenciar o controle de acesso baseado em função (RBAC) para você APIs em diversos contextos de aplicação.

Escolher configurações do repositório de políticas

Ao configurar uma fonte de identidade em um repositório de políticas, você deve informar se deseja processar tokens de acesso ou ID. Essa decisão é importante para a forma como seu mecanismo de políticas opera. Os tokens de ID contêm atributos do usuário. Os tokens de acesso contêm informações de controle de acesso do usuário: OAuth escopos. Embora os dois tipos de token tenham informações de associação ao grupo, geralmente recomendamos o token de acesso para

o RBAC com um repositório de políticas do Verified Permissions. O token de acesso aumenta a associação ao grupo com escopos que podem contribuir para a decisão de autorização. As declarações em um token de acesso se tornam contextuais na solicitação de autorização.

Você também deve configurar os tipos de entidades de usuário e grupo ao configurar um grupo de usuários como fonte de identidade. Os tipos de entidade são identificadores de entidade principal, de ação e de recursos que você pode consultar nas políticas do Verified Permissions. As entidades nos repositórios de políticas podem ter uma relação de associação, em que uma entidade pode ser membro de uma entidade principal. Com a associação, você pode referenciar grupos de entidade principal, grupos de ação e grupos de recursos. No caso de grupos de usuários, o tipo de entidade do usuário que você especificar deve ser membro do tipo de entidade do grupo. Quando você configura um repositório de políticas vinculado à API ou segue a Configuração guiada no console do Verified Permissions, seu repositório de políticas tem automaticamente essa relação principal-membro.

O token de ID pode combinar o RBAC com o controle de acesso por atributo (ABAC). Depois de criar um repositório de políticas vinculado à API, você pode aprimorar suas políticas com atributos de usuário e associação a grupos. As declarações de atributo em um token de ID se tornam atributos de entidade principal na solicitação de autorização. Suas políticas podem tomar decisões de autorização com base nos atributos da entidade principal.

Você também pode configurar um repositório de políticas para aceitar tokens com uma declaração `aud` ou `client_id` que corresponda a uma lista de clientes de aplicações aceitáveis que você fornece.

Exemplo de política para autorização de API baseada em perfil

O exemplo de política a seguir foi criado pela configuração de um repositório de políticas de permissões verificadas para um PetStoreexemplo de API REST.

```
permit(
  principal in PetStore::UserGroup::"us-east-1_EXAMPLE|MyGroup",
  action in [ PetStore::Action::"get /pets", PetStore::Action::"get /pets/{petId}" ],
  resource
  );
```

O Verified Permissions retorna uma decisão `Allow` à solicitação de autorização da aplicação quando:

1. A aplicação passou um ID ou token de acesso em um cabeçalho `Authorization` como um token portador.
2. A aplicação passou um token com uma declaração `cognito:groups` que contém a string`MyGroup`.
3. A aplicação fez uma solicitação `HTTP GET` para, por exemplo, `https://myapi.example.com/pets` ou `https://myapi.example.com/pets/scrappy`.

Exemplo de política para um usuário do Amazon Cognito

Seu grupo de usuários também pode gerar solicitações de autorização para o Verified Permissions em condições diferentes das solicitações de API. Você pode enviar qualquer decisão de controle de acesso na aplicação ao seu repositório de políticas. Por exemplo, você pode complementar a segurança do Amazon DynamoDB ou do Amazon S3 com controle de acesso baseado em atributos antes que qualquer solicitação transite pela rede, reduzindo o uso da cota.

O exemplo a seguir usa a linguagem de política Cedar para permitir que usuários do setor financeiro que se autenticam com um cliente de aplicação do grupo de usuários leiam e escrevam `example_image.png`. John, um usuário da aplicação, recebe um token de ID do cliente da aplicação e o passa em uma solicitação GET para um URL que exige autorização, `https://example.com/images/example_image.png`. O token de ID de John tem uma reivindicação `aud` do ID de cliente da aplicação do grupo de usuários `1234567890example`. A função do Lambda de geração de pré-token também inseriu uma nova reivindicação `costCenter` com um valor, para John, de `Finance1234`.

```
permit (
   principal,
   actions in [ExampleCorp::Action::"readFile", "writeFile"],
   resource == ExampleCorp::Photo::"example_image.png"
)
when {
   principal.aud == "1234567890example" &&
   principal.custom.costCenter like "Finance*"
};
```

O corpo da solicitação a seguir resulta em uma resposta `Allow`.

```
{
   "accesstoken": "[John's ID token]",
```

```
   "action": {
      "actionId": "readFile",
      "actionType": "Action"
   },
   "resource": {
      "entityId": "example_image.png",
      "entityType": "Photo"
   }
}
```

Quando você quiser especificar uma entidade principal em uma política do Verified Permissions, use o seguinte formato:

```
permit (
   principal == [Namespace]::[Entity]::"[user pool ID]|[user sub]",
   action,
   resource
);
```

Veja a seguir um exemplo de entidade principal para um usuário em um grupo de usuários com ID `us-east-1_Example` com sub, ou ID de usuário `973db890-092c-49e4-a9d0-912a4c0a20c7`.

```
principal == ExampleCorp::User::"us-east-1_Example|973db890-092c-49e4-
a9d0-912a4c0a20c7",
```

Quando você quiser especificar um grupo de usuários em uma política do Verified Permissions, use o seguinte formato:

```
permit (
   principal in [Namespace]::[Group Entity]::"[Group name]",
   action,
   resource
);
```

Controle de acesso por atributo

A autorização com permissões verificadas para seus aplicativos e o recurso de atributos para controle de acesso dos grupos de identidade do Amazon Cognito para AWS credenciais são formas de controle de acesso baseado em atributos (ABAC). Veja a seguir uma comparação dos recursos do ABAC do Verified Permissions e do Amazon Cognito. No ABAC, um sistema examina os atributos de uma entidade e toma uma decisão de autorização com base nas condições que você define.

Serviço	Processo	Resultado
Amazon Verified Permissions	Retorna uma `Deny` decisão `Allow` or da análise de um grupo de usuários JWT.	O acesso aos recursos do aplicativo é bem-sucedido ou não, com base na avaliação da política da Cedar.
Grupos de identidade do Amazon Cognito (atributos para controle de acesso)	Atribui tags de sessão ao seu usuário com base em seus atributos . As condições da política do IAM podem verificar as tags `Allow` ou `Deny` o acesso do usuário Serviços da AWS a.	Uma sessão marcada com AWS credenciais temporárias para uma função do IAM.

Exemplos de código do Amazon Cognito usando AWS SDKs

Os exemplos de código a seguir mostram como usar o Amazon Cognito com um Kit de desenvolvimento de software (SDK) da AWS.

Para ver uma lista completa dos Guias do desenvolvedor e exemplos de código do SDK da AWS, consulte Usando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código

- Exemplos de código para o Amazon Cognito Identity usando AWS SDKs
  - Exemplos básicos do Amazon Cognito Identity usando AWS SDKs
    - Ações para o Amazon Cognito Identity usando AWS SDKs
      - Use CreateIdentityPool com um AWS SDK ou CLI
      - Use DeleteIdentityPool com um AWS SDK ou CLI
      - Usar DescribeIdentityPool com uma CLI
      - Use GetCredentialsForIdentity com um AWS SDK
      - Usar GetIdentityPoolRoles com uma CLI
      - Use ListIdentityPools com um AWS SDK ou CLI
      - Usar SetIdentityPoolRoles com uma CLI
      - Usar UpdateIdentityPool com uma CLI
  - Cenários para o Amazon Cognito Identity usando AWS SDKs
    - Criar uma aplicação de exploração do Amazon Textract
- Exemplos de código para o Amazon Cognito Identity Provider usando AWS SDKs
  - Exemplos básicos para o Amazon Cognito Identity Provider usando AWS SDKs
    - Olá, Amazon Cognito
    - Ações para o Amazon Cognito Identity Provider usando AWS SDKs
      - Use AdminCreateUser com um AWS SDK ou CLI
      - Use AdminGetUser com um AWS SDK ou CLI
      - Use AdminInitiateAuth com um AWS SDK ou CLI
      - Use AdminRespondToAuthChallenge com um AWS SDK ou CLI
      - Use AdminSetUserPassword com um AWS SDK ou CLI

Exemplos de código para o Amazon Cognito Identity usando AWS SDKs

Os exemplos de código a seguir mostram como usar o Amazon Cognito Identity com um kit de desenvolvimento AWS de software (SDK).

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar perfis de serviço individuais, você pode ver as ações no contexto em seus cenários relacionados.

Cenários são exemplos de código que mostram como realizar tarefas específicas chamando várias funções dentro de um serviço ou combinadas com outros Serviços da AWS.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Identidade do Amazon Cognito

Exemplos básicos do Amazon Cognito Identity usando AWS SDKs

Os exemplos de código a seguir mostram como usar os conceitos básicos do Amazon Cognito Identity com. AWS SDKs

Exemplos

- Ações para o Amazon Cognito Identity usando AWS SDKs
  - Use CreateIdentityPool com um AWS SDK ou CLI
  - Use DeleteIdentityPool com um AWS SDK ou CLI
  - Usar DescribeIdentityPool com uma CLI
  - Use GetCredentialsForIdentity com um AWS SDK
  - Usar GetIdentityPoolRoles com uma CLI
  - Use ListIdentityPools com um AWS SDK ou CLI
  - Usar SetIdentityPoolRoles com uma CLI
  - Usar UpdateIdentityPool com uma CLI

Ações para o Amazon Cognito Identity usando AWS SDKs

Os exemplos de código a seguir demonstram como realizar ações individuais do Amazon Cognito Identity com. AWS SDKs Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções para configurar e executar o código.

Esses trechos chamam a API de identidade do Amazon Cognito e são trechos de código de programas maiores que devem ser executados no contexto. É possível ver as ações em contexto em Cenários para o Amazon Cognito Identity usando AWS SDKs.

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para ter uma lista completa, consulte a Referência de API do Amazon Cognito Identity.

Exemplos

- Use CreateIdentityPool com um AWS SDK ou CLI
- Use DeleteIdentityPool com um AWS SDK ou CLI
- Usar DescribeIdentityPool com uma CLI
- Use GetCredentialsForIdentity com um AWS SDK
- Usar GetIdentityPoolRoles com uma CLI
- Use ListIdentityPools com um AWS SDK ou CLI
- Usar SetIdentityPoolRoles com uma CLI
- Usar UpdateIdentityPool com uma CLI

Use **`CreateIdentityPool`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `CreateIdentityPool`.

CLI

AWS CLI

Como criar um banco de identidades com o provedor de banco de identidades Cognito

Este exemplo cria um grupo de identidades chamado MyIdentityPool. Ele tem um provedor de banco de identidades Cognito. Identidades não autenticadas não são permitidas.

Comando:

```
aws cognito-identity create-identity-pool --identity-pool-
name MyIdentityPool --no-allow-unauthenticated-identities --cognito-
identity-providers ProviderName="cognito-idp.us-west-2.amazonaws.com/us-
west-2_aaaaaaaaa",ClientId="3n4b5urk1ft4fl3mg5e62d9ado",ServerSideTokenCheck=false
```

Saída:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
      {
          "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-
west-2_111111111",
          "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
          "ServerSideTokenCheck": false
      }
  ]
}
```

- Para obter detalhes da API, consulte CreateIdentityPoolem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
 software.amazon.awssdk.services.cognitoidentity.model.CreateIdentityPoolRequest;
import
 software.amazon.awssdk.services.cognitoidentity.model.CreateIdentityPoolResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc

/**
 * 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 CreateIdentityPool {
    public static void main(String[] args) {
        final String usage = """
                Usage:
                    <identityPoolName>\s

                Where:
                    identityPoolName - The name to give your identity pool.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }
```

```
        String identityPoolName = args[0];
        CognitoIdentityClient cognitoClient = CognitoIdentityClient.builder()
                .region(Region.US_EAST_1)
                .build();

        String identityPoolId = createIdPool(cognitoClient, identityPoolName);
        System.out.println("Unity pool ID " + identityPoolId);
        cognitoClient.close();
    }

    public static String createIdPool(CognitoIdentityClient cognitoClient, String
 identityPoolName) {
        try {
            CreateIdentityPoolRequest poolRequest =
 CreateIdentityPoolRequest.builder()
                    .allowUnauthenticatedIdentities(false)
                    .identityPoolName(identityPoolName)
                    .build();

            CreateIdentityPoolResponse response =
 cognitoClient.createIdentityPool(poolRequest);
            return response.identityPoolId();

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Para obter detalhes da API, consulte CreateIdentityPoola Referência AWS SDK for Java 2.x da API.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: Cria um novo banco de identidades que permite identidades não autenticadas.

```
New-CGIIdentityPool -AllowUnauthenticatedIdentities $true -IdentityPoolName
 CommonTests13
```

Saída:

```
LoggedAt                       : 8/12/2015 4:56:07 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName          :
IdentityPoolId                 : us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3
IdentityPoolName               : CommonTests13
OpenIdConnectProviderARNs      : {}
SupportedLoginProviders        : {}
ResponseMetadata               : Amazon.Runtime.ResponseMetadata
ContentLength                  : 136
HttpStatusCode                 : OK
```

- Para obter detalhes da API, consulte CreateIdentityPoolem Referência de Ferramentas da AWS para PowerShell cmdlet (V4).

Ferramentas para PowerShell V5

Exemplo 1: Cria um novo banco de identidades que permite identidades não autenticadas.

```
New-CGIIdentityPool -AllowUnauthenticatedIdentities $true -IdentityPoolName
 CommonTests13
```

Saída:

```
LoggedAt                       : 8/12/2015 4:56:07 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName          :
IdentityPoolId                 : us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3
IdentityPoolName               : CommonTests13
OpenIdConnectProviderARNs      : {}
SupportedLoginProviders        : {}
ResponseMetadata               : Amazon.Runtime.ResponseMetadata
ContentLength                  : 136
HttpStatusCode                 : OK
```

- Para obter detalhes da API, consulte CreateIdentityPoolem Referência de Ferramentas da AWS para PowerShell cmdlet (V5).

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSCognitoIdentity


    /// Create a new identity pool and return its ID.
    ///
    /// - Parameters:
    ///     - name: The name to give the new identity pool.
    ///
    /// - Returns: A string containing the newly created pool's ID, or `nil`
    ///   if an error occurred.
    ///
    func createIdentityPool(name: String) async throws -> String? {
        do {
            let cognitoInputCall = CreateIdentityPoolInput(developerProviderName:
 "com.exampleco.CognitoIdentityDemo",
                                                           identityPoolName:
 name)

            let result = try await
 cognitoIdentityClient.createIdentityPool(input: cognitoInputCall)
            guard let poolId = result.identityPoolId else {
                return nil
            }

            return poolId
        } catch {
            print("ERROR: createIdentityPool:", dump(error))
            throw error
        }
    }
```

- Para ter mais informações, consulte o Guia do desenvolvedor do AWS SDK para Swift.
- Para obter detalhes da API, consulte CreateIdentityPoola referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`DeleteIdentityPool`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `DeleteIdentityPool`.

CLI

AWS CLI

Como excluir um banco de identidades

O exemplo `delete-identity-pool` a seguir exclui o banco de identidades especificado.

Comando:

```
aws cognito-identity delete-identity-pool \
    --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111"
```

Este comando não produz saída.

- Para obter detalhes da API, consulte DeleteIdentityPoolem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.awscore.exception.AwsServiceException;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
 software.amazon.awssdk.services.cognitoidentity.model.DeleteIdentityPoolRequest;

/**
 * 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 DeleteIdentityPool {

    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <identityPoolId>\s

                Where:
                    identityPoolId - The Id value of your identity pool.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String identityPoold = args[0];
        CognitoIdentityClient cognitoIdClient = CognitoIdentityClient.builder()
                .region(Region.US_EAST_1)
                .credentialsProvider(ProfileCredentialsProvider.create())
                .build();

        deleteIdPool(cognitoIdClient, identityPoold);
        cognitoIdClient.close();
    }
```

```
    public static void deleteIdPool(CognitoIdentityClient cognitoIdClient, String
 identityPoold) {
        try {

            DeleteIdentityPoolRequest identityPoolRequest =
 DeleteIdentityPoolRequest.builder()
                    .identityPoolId(identityPoold)
                    .build();

            cognitoIdClient.deleteIdentityPool(identityPoolRequest);
            System.out.println("Done");

        } catch (AwsServiceException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte DeleteIdentityPoola Referência AWS SDK for Java 2.x da API.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: Exclui um banco de identidades específico.

```
Remove-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

- Para obter detalhes da API, consulte DeleteIdentityPoolem Referência de Ferramentas da AWS para PowerShell cmdlet (V4).

Ferramentas para PowerShell V5

Exemplo 1: Exclui um banco de identidades específico.

```
Remove-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

- Para obter detalhes da API, consulte DeleteIdentityPoolem Referência de Ferramentas da AWS para PowerShell cmdlet (V5).

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSCognitoIdentity


    /// Delete the specified identity pool.
    ///
    /// - Parameters:
    ///   - id: The ID of the identity pool to delete.
    ///
    func deleteIdentityPool(id: String) async throws {
        do {
            let input = DeleteIdentityPoolInput(
                identityPoolId: id
            )

            _ = try await cognitoIdentityClient.deleteIdentityPool(input: input)
        } catch {
            print("ERROR: deleteIdentityPool:", dump(error))
            throw error
        }
    }
```

- Para ter mais informações, consulte o Guia do desenvolvedor do AWS SDK para Swift.
- Para obter detalhes da API, consulte DeleteIdentityPoola referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`DescribeIdentityPool`** com uma CLI

Os exemplos de código a seguir mostram como usar o `DescribeIdentityPool`.

CLI

AWS CLI

Para descrever um banco de identidades

Este exemplo descreve um banco de identidades.

Comando:

```
aws cognito-identity describe-identity-pool --identity-pool-id "us-
west-2:11111111-1111-1111-1111-111111111111"
```

Saída:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
      {
          "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-
west-2_111111111",
          "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
          "ServerSideTokenCheck": false
      }
  ]
}
```

- Para obter detalhes da API, consulte DescribeIdentityPoolem Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: Recupera informações sobre um banco de identidades específico por meio de seu ID.

```
Get-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

Saída:

```
LoggedAt                       : 8/12/2015 4:29:40 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName          :
IdentityPoolId                 : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName               : CommonTests1
OpenIdConnectProviderARNs      : {}
SupportedLoginProviders        : {}
ResponseMetadata               : Amazon.Runtime.ResponseMetadata
ContentLength                  : 142
HttpStatusCode                 : OK
```

- Para obter detalhes da API, consulte DescribeIdentityPoolem Referência de Ferramentas da AWS para PowerShell cmdlet (V4).

Ferramentas para PowerShell V5

Exemplo 1: Recupera informações sobre um banco de identidades específico por meio de seu ID.

```
Get-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

Saída:

```
LoggedAt                       : 8/12/2015 4:29:40 PM
AllowUnauthenticatedIdentities : True
DeveloperProviderName          :
IdentityPoolId                 : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName               : CommonTests1
OpenIdConnectProviderARNs      : {}
SupportedLoginProviders        : {}
```

```
ResponseMetadata                : Amazon.Runtime.ResponseMetadata
ContentLength                   : 142
HttpStatusCode                  : OK
```

- Para obter detalhes da API, consulte DescribeIdentityPoolem Referência de Ferramentas da AWS para PowerShell cmdlet (V5).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`GetCredentialsForIdentity`** com um AWS SDK

O código de exemplo a seguir mostra como usar `GetCredentialsForIdentity`.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
 software.amazon.awssdk.services.cognitoidentity.model.GetCredentialsForIdentityRequest;
import
 software.amazon.awssdk.services.cognitoidentity.model.GetCredentialsForIdentityResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc

/**
 * 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 GetIdentityCredentials {
    public static void main(String[] args) {

        final String usage = """

                Usage:
                    <identityId>\s

            Where:
                identityId - The Id of an existing identity in the format
 REGION:GUID.
            """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String identityId = args[0];
        CognitoIdentityClient cognitoClient = CognitoIdentityClient.builder()
                .region(Region.US_EAST_1)
                .build();

        getCredsForIdentity(cognitoClient, identityId);
        cognitoClient.close();
    }

    public static void getCredsForIdentity(CognitoIdentityClient cognitoClient,
 String identityId) {
        try {
            GetCredentialsForIdentityRequest getCredentialsForIdentityRequest =
 GetCredentialsForIdentityRequest
                    .builder()
                    .identityId(identityId)
                    .build();

            GetCredentialsForIdentityResponse response = cognitoClient
                    .getCredentialsForIdentity(getCredentialsForIdentityRequest);
            System.out.println(
                    "Identity ID " + response.identityId() + ", Access key ID " +
 response.credentials().accessKeyId());
```

```
        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte GetCredentialsForIdentitya Referência AWS SDK for Java 2.x da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`GetIdentityPoolRoles`** com uma CLI

Os exemplos de código a seguir mostram como usar o `GetIdentityPoolRoles`.

CLI

AWS CLI

Para obter funções no banco de identidades

Este exemplo lista bancos de identidades.

Comando:

```
aws cognito-identity get-identity-pool-roles --identity-pool-id "us-
west-2:11111111-1111-1111-1111-111111111111"
```

Saída:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "Roles": {
      "authenticated": "arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolAuth_Role",
      "unauthenticated": "arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolUnauth_Role"
```

```
  }
}
```

- Para obter detalhes da API, consulte GetIdentityPoolRolesem Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: Obtém as informações sobre as funções de um banco de identidades específico.

```
Get-CGIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

Saída:

```
LoggedAt         : 8/12/2015 4:33:51 PM
IdentityPoolId   : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
Roles            : {[unauthenticated, arn:aws:iam::123456789012:role/
CommonTests1Role]}
ResponseMetadata : Amazon.Runtime.ResponseMetadata
ContentLength    : 165
HttpStatusCode   : OK
```

- Para obter detalhes da API, consulte GetIdentityPoolRolesem Referência de Ferramentas da AWS para PowerShell cmdlet (V4).

Ferramentas para PowerShell V5

Exemplo 1: Obtém as informações sobre as funções de um banco de identidades específico.

```
Get-CGIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1
```

Saída:

```
LoggedAt         : 8/12/2015 4:33:51 PM
IdentityPoolId   : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
Roles            : {[unauthenticated, arn:aws:iam::123456789012:role/
CommonTests1Role]}
ResponseMetadata : Amazon.Runtime.ResponseMetadata
```

```
ContentLength    : 165
HttpStatusCode   : OK
```

- Para obter detalhes da API, consulte GetIdentityPoolRolesem Referência de Ferramentas da AWS para PowerShell cmdlet (V5).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`ListIdentityPools`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `ListIdentityPools`.

CLI

AWS CLI

Para listar bancos de identidades

Este exemplo lista bancos de identidades. Há no máximo vinte identidades listadas.

Comando:

```
aws cognito-identity list-identity-pools --max-results 20
```

Saída:

```
{
  "IdentityPools": [
      {
          "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
          "IdentityPoolName": "MyIdentityPool"
      },
      {
          "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
          "IdentityPoolName": "AnotherIdentityPool"
      },
      {
          "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
          "IdentityPoolName": "IdentityPoolRegionA"
      }
```

```
  ]
}
```

- Para obter detalhes da API, consulte ListIdentityPoolsem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.cognitoidentity.CognitoIdentityClient;
import
 software.amazon.awssdk.services.cognitoidentity.model.ListIdentityPoolsRequest;
import
 software.amazon.awssdk.services.cognitoidentity.model.ListIdentityPoolsResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc

/**
 * 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 ListIdentityPools {
    public static void main(String[] args) {
        CognitoIdentityClient cognitoClient = CognitoIdentityClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listIdPools(cognitoClient);
```

```
        cognitoClient.close();
    }

    public static void listIdPools(CognitoIdentityClient cognitoClient) {
        try {
            ListIdentityPoolsRequest poolsRequest =
 ListIdentityPoolsRequest.builder()
                    .maxResults(15)
                    .build();

            ListIdentityPoolsResponse response =
 cognitoClient.listIdentityPools(poolsRequest);
            response.identityPools().forEach(pool -> {
                System.out.println("Pool ID: " + pool.identityPoolId());
                System.out.println("Pool name: " + pool.identityPoolName());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte ListIdentityPoolsa Referência AWS SDK for Java 2.x da API.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: Recupera uma lista de bancos de identidades existentes.

```
Get-CGIIdentityPoolList
```

Saída:

```
IdentityPoolId
 IdentityPoolName
--------------
 ----------------
```

```
us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1                    CommonTests1
us-east-1:118d242d-204e-4b88-b803-EXAMPLEGUID2                    Tests2
us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3                    CommonTests13
```

- Para obter detalhes da API, consulte ListIdentityPoolsem Referência de Ferramentas da AWS para PowerShell cmdlet (V4).

Ferramentas para PowerShell V5

Exemplo 1: Recupera uma lista de bancos de identidades existentes.

```
Get-CGIIdentityPoolList
```

Saída:

```
IdentityPoolId
 IdentityPoolName
--------------
 ----------------
us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1                    CommonTests1
us-east-1:118d242d-204e-4b88-b803-EXAMPLEGUID2                    Tests2
us-east-1:15d49393-ab16-431a-b26e-EXAMPLEGUID3                    CommonTests13
```

- Para obter detalhes da API, consulte ListIdentityPoolsem Referência de Ferramentas da AWS para PowerShell cmdlet (V5).

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSCognitoIdentity


    /// Return the ID of the identity pool with the specified name.
    ///
```

```
    /// - Parameters:
    ///   - name: The name of the identity pool whose ID should be returned.
    ///
    /// - Returns: A string containing the ID of the specified identity pool
    ///   or `nil` on error or if not found.
    ///
    func getIdentityPoolID(name: String) async throws -> String? {
        let listPoolsInput = ListIdentityPoolsInput(maxResults: 25)
        // Use "Paginated" to get all the objects.
        // This lets the SDK handle the 'nextToken' field in
"ListIdentityPoolsOutput".
        let pages = cognitoIdentityClient.listIdentityPoolsPaginated(input:
listPoolsInput)

        do {
            for try await page in pages {
                guard let identityPools = page.identityPools else {
                    print("ERROR: listIdentityPoolsPaginated returned nil
contents.")
                    continue
                }

                /// Read pages of identity pools from Cognito until one is found
                /// whose name matches the one specified in the `name` parameter.
                /// Return the matching pool's ID.

                for pool in identityPools {
                    if pool.identityPoolName == name {
                        return pool.identityPoolId!
                    }
                }
            }
        } catch {
            print("ERROR: getIdentityPoolID:", dump(error))
            throw error
        }

        return nil
    }
```

Obtenha o ID de um banco de identidades existente ou crie-o se ainda não existir.

```
import AWSCognitoIdentity


    /// Return the ID of the identity pool with the specified name.
    ///
    /// - Parameters:
    ///   - name: The name of the identity pool whose ID should be returned
    ///
    /// - Returns: A string containing the ID of the specified identity pool.
    ///   Returns `nil` if there's an error or if the pool isn't found.
    ///
    public func getOrCreateIdentityPoolID(name: String) async throws -> String? {
        // See if the pool already exists. If it doesn't, create it.

        do {
            guard let poolId = try await getIdentityPoolID(name: name) else {
                return try await createIdentityPool(name: name)
            }

            return poolId
        } catch {
            print("ERROR: getOrCreateIdentityPoolID:", dump(error))
            throw error
        }
    }
```

- Para ter mais informações, consulte o Guia do desenvolvedor do AWS SDK para Swift.
- Para obter detalhes da API, consulte ListIdentityPoolsa referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`SetIdentityPoolRoles`** com uma CLI

Os exemplos de código a seguir mostram como usar o `SetIdentityPoolRoles`.

CLI

AWS CLI

Para definir funções do banco de identidades

O exemplo `set-identity-pool-roles` a seguir define funções para um banco de identidades.

```
aws cognito-identity set-identity-pool-roles \
    --identity-pool-id "us-west-2:11111111-1111-1111-1111-111111111111" \
    --roles authenticated="arn:aws:iam::111111111111:role/
Cognito_MyIdentityPoolAuth_Role"
```

- Para obter detalhes da API, consulte SetIdentityPoolRolesem Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: Configura o banco de identidades específico para ter um perfil do IAM não autenticado.

```
Set-CGIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1 -Role @{ "unauthenticated" = "arn:aws:iam::123456789012:role/
CommonTests1Role" }
```

- Para obter detalhes da API, consulte SetIdentityPoolRolesem Referência de Ferramentas da AWS para PowerShell cmdlet (V4).

Ferramentas para PowerShell V5

Exemplo 1: Configura o banco de identidades específico para ter um perfil do IAM não autenticado.

```
Set-CGIIdentityPoolRole -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1 -Role @{ "unauthenticated" = "arn:aws:iam::123456789012:role/
CommonTests1Role" }
```

- Para obter detalhes da API, consulte SetIdentityPoolRolesem Referência de Ferramentas da AWS para PowerShell cmdlet (V5).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar **`UpdateIdentityPool`** com uma CLI

Os exemplos de código a seguir mostram como usar o `UpdateIdentityPool`.

CLI

AWS CLI

Para atualizar um banco de identidades

Este exemplo atualiza um banco de identidades. Ele define o nome como MyIdentityPool. Ele tem um provedor de banco de identidades Cognito. Ele não permite identidades não autenticadas.

Comando:

```
aws cognito-identity update-identity-pool --identity-pool-id "us-
west-2:11111111-1111-1111-1111-111111111111" --identity-pool-
name "MyIdentityPool" --no-allow-unauthenticated-identities --cognito-
identity-providers ProviderName="cognito-idp.us-west-2.amazonaws.com/us-
west-2_111111111",ClientId="3n4b5urk1ft4fl3mg5e62d9ado",ServerSideTokenCheck=false
```

Saída:

```
{
  "IdentityPoolId": "us-west-2:11111111-1111-1111-1111-111111111111",
  "IdentityPoolName": "MyIdentityPool",
  "AllowUnauthenticatedIdentities": false,
  "CognitoIdentityProviders": [
      {
          "ProviderName": "cognito-idp.us-west-2.amazonaws.com/us-
west-2_111111111",
          "ClientId": "3n4b5urk1ft4fl3mg5e62d9ado",
          "ServerSideTokenCheck": false
      }
  ]
}
```

- Para obter detalhes da API, consulte UpdateIdentityPoolem Referência de AWS CLI Comandos.

PowerShell

Ferramentas para PowerShell V4

Exemplo 1: Atualiza algumas das propriedades do banco de identidades, neste caso, o nome do banco de identidades.

```
Update-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1 -IdentityPoolName NewPoolName
```

Saída:

```
LoggedAt                       : 8/12/2015 4:53:33 PM
AllowUnauthenticatedIdentities : False
DeveloperProviderName          :
IdentityPoolId                 : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName               : NewPoolName
OpenIdConnectProviderARNs      : {}
SupportedLoginProviders        : {}
ResponseMetadata               : Amazon.Runtime.ResponseMetadata
ContentLength                  : 135
HttpStatusCode                 : OK
```

- Para obter detalhes da API, consulte UpdateIdentityPoolem Referência de Ferramentas da AWS para PowerShell cmdlet (V4).

Ferramentas para PowerShell V5

Exemplo 1: Atualiza algumas das propriedades do banco de identidades, neste caso, o nome do banco de identidades.

```
Update-CGIIdentityPool -IdentityPoolId us-east-1:0de2af35-2988-4d0b-b22d-
EXAMPLEGUID1 -IdentityPoolName NewPoolName
```

Saída:

```
LoggedAt                       : 8/12/2015 4:53:33 PM
AllowUnauthenticatedIdentities : False
DeveloperProviderName          :
```

```
IdentityPoolId                 : us-east-1:0de2af35-2988-4d0b-b22d-EXAMPLEGUID1
IdentityPoolName               : NewPoolName
OpenIdConnectProviderARNs      : {}
SupportedLoginProviders        : {}
ResponseMetadata               : Amazon.Runtime.ResponseMetadata
ContentLength                  : 135
HttpStatusCode                 : OK
```

- Para obter detalhes da API, consulte UpdateIdentityPoolem Referência de Ferramentas da AWS para PowerShell cmdlet (V5).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Cenários para o Amazon Cognito Identity usando AWS SDKs

Os exemplos de código a seguir mostram como implementar cenários comuns no Amazon Cognito Identity com. AWS SDKs Esses cenários mostram como realizar tarefas específicas chamando várias funções no Amazon Cognito Identity ou em combinação com outros Serviços da AWS. Cada cenário inclui um link para o código-fonte completo, onde podem ser encontradas instruções sobre como configurar e executar o código.

Os cenários têm como alvo um nível intermediário de experiência para ajudar você a compreender ações de serviço em contexto.

Exemplos

- Criar uma aplicação de exploração do Amazon Textract

Criar uma aplicação de exploração do Amazon Textract

Os exemplos de código a seguir mostram como explorar a saída do Amazon Textract por meio de uma aplicação interativa.

JavaScript

SDK para JavaScript (v3)

Mostra como usar o AWS SDK para JavaScript para criar um aplicativo React que usa o Amazon Textract para extrair dados de uma imagem de documento e exibi-los em uma

página da web interativa. Este exemplo é executado em um navegador da Web e requer uma identidade autenticada do Amazon Cognito como credenciais. Ele usa o Amazon Simple Storage Service (Amazon S3) para armazenamento e, para notificações, pesquisa uma fila do Amazon Simple Queue Service (Amazon SQS) que está inscrita em um tópico do Amazon Simple Notification Service (Amazon SNS).

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em GitHub.

Serviços usados neste exemplo

- Identidade do Amazon Cognito
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Python

SDK para Python (Boto3)

Mostra como usar o AWS SDK para Python (Boto3) com o Amazon Textract para detectar elementos de texto, formulário e tabela em uma imagem de documento. A imagem de entrada e a saída do Amazon Textract são mostradas em um aplicativo Tkinter que permite explorar os elementos detectados.

- Envie uma imagem de documento para o Amazon Textract e explore a saída dos elementos detectados.
- Envie imagens diretamente para o Amazon Textract ou por meio de um bucket do Amazon Simple Storage Service (Amazon S3).
- Use o modo assíncrono APIs para iniciar um trabalho que publica uma notificação em um tópico do Amazon Simple Notification Service (Amazon SNS) quando o trabalho for concluído.
- Faça uma pesquisa em uma fila do Amazon Simple Queue Service (Amazon SQS) para obter uma mensagem de conclusão do trabalho e exiba os resultados.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em GitHub.

Serviços usados neste exemplo

- Identidade do Amazon Cognito
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Textract

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código para o Amazon Cognito Identity Provider usando AWS SDKs

Os exemplos de código a seguir mostram como usar o Amazon Cognito Identity Provider com um kit de desenvolvimento AWS de software (SDK).

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar perfis de serviço individuais, você pode ver as ações no contexto em seus cenários relacionados.

Cenários são exemplos de código que mostram como realizar tarefas específicas chamando várias funções dentro de um serviço ou combinadas com outros Serviços da AWS.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Provedor de identidade do Amazon Cognito

- Exemplos básicos para o Amazon Cognito Identity Provider usando AWS SDKs
  - Olá, Amazon Cognito
  - Ações para o Amazon Cognito Identity Provider usando AWS SDKs
    - Use AdminCreateUser com um AWS SDK ou CLI
    - Use AdminGetUser com um AWS SDK ou CLI

- Use AdminInitiateAuth com um AWS SDK ou CLI
- Use AdminRespondToAuthChallenge com um AWS SDK ou CLI
- Use AdminSetUserPassword com um AWS SDK ou CLI
- Use AssociateSoftwareToken com um AWS SDK ou CLI
- Use ConfirmDevice com um AWS SDK ou CLI
- Use ConfirmForgotPassword com um AWS SDK ou CLI
- Use ConfirmSignUp com um AWS SDK ou CLI
- Use CreateUserPool com um AWS SDK ou CLI
- Use CreateUserPoolClient com um AWS SDK ou CLI
- Use DeleteUser com um AWS SDK ou CLI
- Use ForgotPassword com um AWS SDK ou CLI
- Use InitiateAuth com um AWS SDK ou CLI
- Use ListUserPools com um AWS SDK ou CLI
- Use ListUsers com um AWS SDK ou CLI
- Use ResendConfirmationCode com um AWS SDK ou CLI
- Use RespondToAuthChallenge com um AWS SDK ou CLI
- Use SignUp com um AWS SDK ou CLI
- Use UpdateUserPool com um AWS SDK ou CLI
- Use VerifySoftwareToken com um AWS SDK ou CLI

- Cenários para o Amazon Cognito Identity Provider usando AWS SDKs
  - Confirme automaticamente usuários conhecidos do Amazon Cognito com uma função Lambda usando um SDK AWS
  - Migre automaticamente usuários conhecidos do Amazon Cognito com uma função Lambda usando um SDK AWS
  - Cadastrar um usuário com um grupo de usuários do Amazon Cognito que exija MFA usando um SDK AWS
  - Usar bancos de identidades e fluxos de identidades do Amazon Cognito
  - Grave dados de atividades personalizados com uma função Lambda após a autenticação do usuário do Amazon Cognito usando um SDK AWS

Exemplos básicos para o Amazon Cognito Identity Provider usando AWS SDKs

Os exemplos de código a seguir mostram como usar os conceitos básicos do Amazon Cognito Identity Provider com. AWS SDKs

Exemplos

- Olá, Amazon Cognito
- Ações para o Amazon Cognito Identity Provider usando AWS SDKs
  - Use AdminCreateUser com um AWS SDK ou CLI
  - Use AdminGetUser com um AWS SDK ou CLI
  - Use AdminInitiateAuth com um AWS SDK ou CLI
  - Use AdminRespondToAuthChallenge com um AWS SDK ou CLI
  - Use AdminSetUserPassword com um AWS SDK ou CLI
  - Use AssociateSoftwareToken com um AWS SDK ou CLI
  - Use ConfirmDevice com um AWS SDK ou CLI
  - Use ConfirmForgotPassword com um AWS SDK ou CLI
  - Use ConfirmSignUp com um AWS SDK ou CLI
  - Use CreateUserPool com um AWS SDK ou CLI
  - Use CreateUserPoolClient com um AWS SDK ou CLI
  - Use DeleteUser com um AWS SDK ou CLI
  - Use ForgotPassword com um AWS SDK ou CLI
  - Use InitiateAuth com um AWS SDK ou CLI
  - Use ListUserPools com um AWS SDK ou CLI
  - Use ListUsers com um AWS SDK ou CLI
  - Use ResendConfirmationCode com um AWS SDK ou CLI
  - Use RespondToAuthChallenge com um AWS SDK ou CLI
  - Use SignUp com um AWS SDK ou CLI
  - Use UpdateUserPool com um AWS SDK ou CLI
  - Use VerifySoftwareToken com um AWS SDK ou CLI

Olá, Amazon Cognito

Os exemplos de código a seguir mostram como começar a usar o Amazon Cognito.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Código para o CMake arquivo CMake Lists.txt.

```
# 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 cognito-idp)

# Set this project's name.
project("hello_cognito")

# 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_cognito.cpp)

target_link_libraries(${PROJECT_NAME}
        ${AWSSDK_LINK_LIBRARIES})
```

Código para o arquivo de origem hello_cognito.cpp.

```
#include <aws/core/Aws.h>
#include <aws/cognito-idp/CognitoIdentityProviderClient.h>
#include <aws/cognito-idp/model/ListUserPoolsRequest.h>
#include <iostream>

/*
 *  A "Hello Cognito" starter application which initializes an Amazon Cognito
 client and lists the Amazon Cognito
 *  user pools.
 *
 *  main function
 *
 *  Usage: 'hello_cognito'
 *
 */

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;
```

```
    {
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

        Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
cognitoClient(clientConfig);

        Aws::String nextToken; // Used for pagination.
        std::vector<Aws::String> userPools;

        do {
            Aws::CognitoIdentityProvider::Model::ListUserPoolsRequest
listUserPoolsRequest;
            if (!nextToken.empty()) {
                listUserPoolsRequest.SetNextToken(nextToken);
            }

            Aws::CognitoIdentityProvider::Model::ListUserPoolsOutcome
listUserPoolsOutcome =
                    cognitoClient.ListUserPools(listUserPoolsRequest);

            if (listUserPoolsOutcome.IsSuccess()) {
                for (auto &userPool:
listUserPoolsOutcome.GetResult().GetUserPools()) {

                    userPools.push_back(userPool.GetName());
                }

                nextToken = listUserPoolsOutcome.GetResult().GetNextToken();
            } else {
                std::cerr << "ListUserPools error: " <<
listUserPoolsOutcome.GetError().GetMessage() << std::endl;
                result = 1;
                break;
            }


        } while (!nextToken.empty());
        std::cout << userPools.size() << " user pools found." << std::endl;
        for (auto &userPool: userPools) {
            std::cout << "   user pool: " << userPool << std::endl;
        }
    }
```

```
    Aws::ShutdownAPI(options); // Should only be called once.
    return result;
}
```

- Para obter detalhes da API, consulte ListUserPoolsa Referência AWS SDK para C++ da API.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
package main

import (
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

// main uses the AWS SDK for Go V2 to create an Amazon Simple Notification
 Service
// (Amazon SNS) client and list the topics in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
```

```
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 cognitoClient := cognitoidentityprovider.NewFromConfig(sdkConfig)
 fmt.Println("Let's list the user pools for your account.")
 var pools []types.UserPoolDescriptionType
 paginator := cognitoidentityprovider.NewListUserPoolsPaginator(
  cognitoClient, &cognitoidentityprovider.ListUserPoolsInput{MaxResults:
 aws.Int32(10)})
 for paginator.HasMorePages() {
  output, err := paginator.NextPage(ctx)
  if err != nil {
   log.Printf("Couldn't get user pools. Here's why: %v\n", err)
  } else {
   pools = append(pools, output.UserPools...)
  }
 }
 if len(pools) == 0 {
  fmt.Println("You don't have any user pools!")
 } else {
  for _, pool := range pools {
   fmt.Printf("\t%v: %v\n", *pool.Name, *pool.Id)
  }
 }
}
```

- Para obter detalhes da API, consulte ListUserPoolsa Referência AWS SDK para Go da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsRequest;

/**
 * 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 ListUserPools {
    public static void main(String[] args) {
        CognitoIdentityProviderClient cognitoClient =
 CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listAllUserPools(cognitoClient);
        cognitoClient.close();
    }

    public static void listAllUserPools(CognitoIdentityProviderClient
 cognitoClient) {
        try {
            ListUserPoolsRequest request = ListUserPoolsRequest.builder()
                    .maxResults(10)
                    .build();

            ListUserPoolsResponse response =
 cognitoClient.listUserPools(request);
            response.userPools().forEach(userpool -> {
                System.out.println("User pool " + userpool.name() + ", User ID "
 + userpool.id());
            });
```

```
        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte ListUserPoolsa Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import {
  paginateListUserPools,
  CognitoIdentityProviderClient,
} from "@aws-sdk/client-cognito-identity-provider";

const client = new CognitoIdentityProviderClient({});

export const helloCognito = async () => {
  const paginator = paginateListUserPools({ client }, {});

  const userPoolNames = [];

  for await (const page of paginator) {
    const names = page.UserPools.map((pool) => pool.Name);
    userPoolNames.push(...names);
  }

  console.log("User pool names: ");
  console.log(userPoolNames.join("\n"));
```

```
  return userPoolNames;
};
```

- Para obter detalhes da API, consulte ListUserPoolsa Referência AWS SDK para JavaScript da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import boto3

# Create a Cognito Identity Provider client
cognitoidp = boto3.client("cognito-idp")

# Initialize a paginator for the list_user_pools operation
paginator = cognitoidp.get_paginator("list_user_pools")

# Create a PageIterator from the paginator
page_iterator = paginator.paginate(MaxResults=10)

# Initialize variables for pagination
user_pools = []

# Handle pagination
for page in page_iterator:
    user_pools.extend(page.get("UserPools", []))

# Print the list of user pools
print("User Pools for the account:")
if user_pools:
    for pool in user_pools:
        print(f"Name: {pool['Name']}, ID: {pool['Id']}")
```

```
else:
    print("No user pools found.")
```

- Para obter detalhes da API, consulte a ListUserPoolsReferência da API AWS SDK for Python (Boto3).

Ruby

SDK para Ruby

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
require 'aws-sdk-cognitoidentityprovider'
require 'logger'

# CognitoManager is a class responsible for managing AWS Cognito operations
# such as listing all user pools in the current AWS account.
class CognitoManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all user pools associated with the AWS account.
  def list_user_pools
    paginator = @client.list_user_pools(max_results: 10)
    user_pools = []
    paginator.each_page do |page|
      user_pools.concat(page.user_pools)
    end

    if user_pools.empty?
      @logger.info('No Cognito user pools found.')
    else
```

```
      user_pools.each do |user_pool|
        @logger.info("User pool ID: #{user_pool.id}")
        @logger.info("User pool name: #{user_pool.name}")
        @logger.info("User pool status: #{user_pool.status}")
        @logger.info('---')
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  cognito_client = Aws::CognitoIdentityProvider::Client.new
  manager = CognitoManager.new(cognito_client)
  manager.list_user_pools
end
```

- Para obter detalhes da API, consulte ListUserPoolsa Referência AWS SDK para Ruby da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Ações para o Amazon Cognito Identity Provider usando AWS SDKs

Os exemplos de código a seguir demonstram como realizar ações individuais do Amazon Cognito Identity Provider com. AWS SDKs Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções para configurar e executar o código.

Esses trechos chamam a API do Provedor de identidade do Amazon Cognito e são trechos de código de programas maiores que devem ser executados no contexto. É possível ver as ações em contexto em Cenários para o Amazon Cognito Identity Provider usando AWS SDKs.

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para ter uma lista completa, consulte a Referência de API do Amazon Cognito Identity Provider.

Exemplos

- Use AdminCreateUser com um AWS SDK ou CLI

- Use AdminGetUser com um AWS SDK ou CLI
- Use AdminInitiateAuth com um AWS SDK ou CLI
- Use AdminRespondToAuthChallenge com um AWS SDK ou CLI
- Use AdminSetUserPassword com um AWS SDK ou CLI
- Use AssociateSoftwareToken com um AWS SDK ou CLI
- Use ConfirmDevice com um AWS SDK ou CLI
- Use ConfirmForgotPassword com um AWS SDK ou CLI
- Use ConfirmSignUp com um AWS SDK ou CLI
- Use CreateUserPool com um AWS SDK ou CLI
- Use CreateUserPoolClient com um AWS SDK ou CLI
- Use DeleteUser com um AWS SDK ou CLI
- Use ForgotPassword com um AWS SDK ou CLI
- Use InitiateAuth com um AWS SDK ou CLI
- Use ListUserPools com um AWS SDK ou CLI
- Use ListUsers com um AWS SDK ou CLI
- Use ResendConfirmationCode com um AWS SDK ou CLI
- Use RespondToAuthChallenge com um AWS SDK ou CLI
- Use SignUp com um AWS SDK ou CLI
- Use UpdateUserPool com um AWS SDK ou CLI
- Use VerifySoftwareToken com um AWS SDK ou CLI

Use **`AdminCreateUser`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `AdminCreateUser`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Grave dados de atividades personalizados com uma função do Lambda após a autenticação do usuário do Amazon Cognito

CLI

AWS CLI

Para criar um usuário

O `admin-create-user` exemplo a seguir cria um usuário com as configurações especificadas de endereço de e-mail e número de telefone.

```
aws cognito-idp admin-create-user \
    --user-pool-id us-west-2_aaaaaaaaa \
    --username diego \
    --user-attributes Name=email,Value=diego@example.com
 Name=phone_number,Value="+15555551212" \
    --message-action SUPPRESS
```

Saída:

```
{
    "User": {
        "Username": "diego",
        "Attributes": [
            {
                "Name": "sub",
                "Value": "7325c1de-b05b-4f84-b321-9adc6e61f4a2"
            },
            {
                "Name": "phone_number",
                "Value": "+15555551212"
            },
            {
                "Name": "email",
                "Value": "diego@example.com"
            }
        ],
        "UserCreateDate": 1548099495.428,
        "UserLastModifiedDate": 1548099495.428,
        "Enabled": true,
        "UserStatus": "FORCE_CHANGE_PASSWORD"
    }
}
```

- Para obter detalhes da API, consulte AdminCreateUserem Referência de AWS CLI Comandos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// AdminCreateUser uses administrator credentials to add a user to a user pool.
 This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(ctx context.Context, userPoolId
 string, userName string, userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(ctx,
 &cognitoidentityprovider.AdminCreateUserInput{
  UserPoolId:     aws.String(userPoolId),
  Username:       aws.String(userName),
  MessageAction:  types.MessageActionTypeSuppress,
```

```
  UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
 aws.String(userEmail)}},
 })
 if err != nil {
  var userExists *types.UsernameExistsException
  if errors.As(err, &userExists) {
   log.Printf("User %v already exists in the user pool.", userName)
   err = nil
  } else {
   log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
  }
 }
 return err
}
```

- Para obter detalhes da API, consulte AdminCreateUsera Referência AWS SDK para Go da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`AdminGetUser`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `AdminGetUser`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// Get the specified user from an Amazon Cognito user pool with
administrator access.
    /// </summary>
    /// <param name="userName">The name of the user.</param>
    /// <param name="poolId">The Id of the Amazon Cognito user pool.</param>
    /// <returns>Async task.</returns>
    public async Task<UserStatusType> GetAdminUserAsync(string userName, string
poolId)
    {
        AdminGetUserRequest userRequest = new AdminGetUserRequest
        {
            Username = userName,
            UserPoolId = poolId,
        };

        var response = await _cognitoService.AdminGetUserAsync(userRequest);

        Console.WriteLine($"User status {response.UserStatus}");
        return response.UserStatus;
    }
```

- Para obter detalhes da API, consulte AdminGetUsera Referência AWS SDK para .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

    Aws::CognitoIdentityProvider::Model::AdminGetUserRequest request;
    request.SetUsername(userName);
    request.SetUserPoolId(userPoolID);

    Aws::CognitoIdentityProvider::Model::AdminGetUserOutcome outcome =
            client.AdminGetUser(request);

    if (outcome.IsSuccess()) {
        std::cout << "The status for " << userName << " is " <<

Aws::CognitoIdentityProvider::Model::UserStatusTypeMapper::GetNameForUserStatusType(
                          outcome.GetResult().GetUserStatus()) << std::endl;
        std::cout << "Enabled is " << outcome.GetResult().GetEnabled() <<
std::endl;
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::AdminGetUser. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }
```

- Para obter detalhes da API, consulte AdminGetUsera Referência AWS SDK para C++ da API.

CLI

AWS CLI

Como obter um usuário

Este exemplo obtém informações sobre o nome de usuário jane@example.com.

Comando:

```
aws cognito-idp admin-get-user --user-pool-id us-west-2_aaaaaaaaa --
username jane@example.com
```

Saída:

```
{
  "Username": "4320de44-2322-4620-999b-5e2e1c8df013",
  "Enabled": true,
  "UserStatus": "FORCE_CHANGE_PASSWORD",
  "UserCreateDate": 1548108509.537,
  "UserAttributes": [
      {
          "Name": "sub",
          "Value": "4320de44-2322-4620-999b-5e2e1c8df013"
      },
      {
          "Name": "email_verified",
          "Value": "true"
      },
      {
          "Name": "phone_number_verified",
          "Value": "true"
      },
      {
          "Name": "phone_number",
          "Value": "+01115551212"
      },
      {
          "Name": "email",
          "Value": "jane@example.com"
      }
  ],
  "UserLastModifiedDate": 1548108509.537
```

```
}
```

- Para obter detalhes da API, consulte AdminGetUserem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
   public static void getAdminUser(CognitoIdentityProviderClient
identityProviderClient, String userName,
           String poolId) {
       try {
           AdminGetUserRequest userRequest = AdminGetUserRequest.builder()
                   .username(userName)
                   .userPoolId(poolId)
                   .build();

           AdminGetUserResponse response =
identityProviderClient.adminGetUser(userRequest);
           System.out.println("User status " + response.userStatusAsString());

       } catch (CognitoIdentityProviderException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Para obter detalhes da API, consulte AdminGetUsera Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const adminGetUser = ({ userPoolId, username }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new AdminGetUserCommand({
    UserPoolId: userPoolId,
    Username: username,
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte AdminGetUsera Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
suspend fun getAdminUser(
    userNameVal: String?,
    poolIdVal: String?,
) {
    val userRequest =
```

```
        AdminGetUserRequest {
            username = userNameVal
            userPoolId = poolIdVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.adminGetUser(userRequest)
        println("User status ${response.userStatus}")
    }
}
```

- Para obter detalhes da API, consulte a AdminGetUserreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
```

```
        self.client_id = client_id
        self.client_secret = client_secret


    def sign_up_user(self, user_name, password, user_email):
        """
        Signs up a new user with Amazon Cognito. This action prompts Amazon
Cognito
        to send an email to the specified email address. The email contains a
code that
        can be used to confirm the user.

        When the user already exists, the user status is checked to determine
whether
        the user has been confirmed.

        :param user_name: The user name that identifies the new user.
        :param password: The password for the new user.
        :param user_email: The email address for the new user.
        :return: True when the user is already confirmed with Amazon Cognito.
                 Otherwise, false.
        """
        try:
            kwargs = {
                "ClientId": self.client_id,
                "Username": user_name,
                "Password": password,
                "UserAttributes": [{"Name": "email", "Value": user_email}],
            }
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            response = self.cognito_idp_client.sign_up(**kwargs)
            confirmed = response["UserConfirmed"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "UsernameExistsException":
                response = self.cognito_idp_client.admin_get_user(
                    UserPoolId=self.user_pool_id, Username=user_name
                )
                logger.warning(
                    "User %s exists and is %s.", user_name,
response["UserStatus"]
                )
                confirmed = response["UserStatus"] == "CONFIRMED"
            else:
```

```
                logger.error(
                    "Couldn't sign up %s. Here's why: %s: %s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return confirmed
```

- Para obter detalhes da API, consulte a AdminGetUserReferência da API AWS SDK for Python (Boto3).

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Get information about a specific user in a user pool.
    ///
    /// - Parameters:
    ///   - cipClient: The Amazon Cognito Identity Provider client to use.
    ///   - userName: The user to retrieve information about.
    ///   - userPoolId: The user pool to search for the specified user.
    ///
    /// - Returns: `true` if the user's information was successfully
    ///   retrieved. Otherwise returns `false`.
    func adminGetUser(cipClient: CognitoIdentityProviderClient, userName: String,
                      userPoolId: String) async -> Bool {
        do {
            let output = try await cipClient.adminGetUser(
                input: AdminGetUserInput(
```

```
                userPoolId: userPoolId,
                username: userName
            )
        )

        guard let userStatus = output.userStatus else {
            print("*** Unable to get the user's status.")
            return false
        }

        print("User status: \(userStatus)")
        return true
    } catch {
        return false
    }
}
```

- Para obter detalhes da API, consulte AdminGetUsera referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`AdminInitiateAuth`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `AdminInitiateAuth`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// Initiate an admin auth request.
    /// </summary>
    /// <param name="clientId">The client ID to use.</param>
    /// <param name="userPoolId">The ID of the user pool.</param>
    /// <param name="userName">The username to authenticate.</param>
    /// <param name="password">The user's password.</param>
    /// <returns>The session to use in challenge-response.</returns>
    public async Task<string> AdminInitiateAuthAsync(string clientId, string
userPoolId, string userName, string password)
    {
        var authParameters = new Dictionary<string, string>();
        authParameters.Add("USERNAME", userName);
        authParameters.Add("PASSWORD", password);

        var request = new AdminInitiateAuthRequest
        {
            ClientId = clientId,
            UserPoolId = userPoolId,
            AuthParameters = authParameters,
            AuthFlow = AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
        };

        var response = await _cognitoService.AdminInitiateAuthAsync(request);
        return response.Session;
    }
```

- Para obter detalhes da API, consulte AdminInitiateAutha Referência AWS SDK para .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

    Aws::CognitoIdentityProvider::Model::AdminInitiateAuthRequest request;
    request.SetClientId(clientID);
    request.SetUserPoolId(userPoolID);
    request.AddAuthParameters("USERNAME", userName);
    request.AddAuthParameters("PASSWORD", password);
    request.SetAuthFlow(

Aws::CognitoIdentityProvider::Model::AuthFlowType::ADMIN_USER_PASSWORD_AUTH);


    Aws::CognitoIdentityProvider::Model::AdminInitiateAuthOutcome outcome =
            client.AdminInitiateAuth(request);

    if (outcome.IsSuccess()) {
        std::cout << "Call to AdminInitiateAuth was successful." << std::endl;
        sessionResult = outcome.GetResult().GetSession();
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::AdminInitiateAuth. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }
```

- Para obter detalhes da API, consulte AdminInitiateAutha Referência AWS SDK para C++ da API.

CLI

AWS CLI

Como fazer login como usuário administrador

O exemplo de `admin-initiate-auth` a seguir faz login com o usuário diego@example.com. Esse exemplo também inclui metadados para proteção contra ameaças

e ClientMetadata para acionadores Lambda. O usuário está configurado para MFA TOTP e recebe um desafio para fornecer um código da aplicação autenticadora antes de concluir a autenticação.

```
aws cognito-idp admin-initiate-auth \
    --user-pool-id us-west-2_EXAMPLE \
    --client-id 1example23456789 \
    --auth-flow ADMIN_USER_PASSWORD_AUTH \
    --auth-parameters USERNAME=diego@example.com,PASSWORD="My@Example
$Password3!",SECRET_HASH=ExampleEncodedClientIdSecretAndUsername= \
    --context-data="{\"EncodedData\":\"abc123example\",\"HttpHeaders\":
[{\"headerName\":\"UserAgent\",\"headerValue\":\"Mozilla/5.0 (Windows NT
 6.1; Win64; x64; rv:47.0) Gecko/20100101 Firefox/47.0\"}],\"IpAddress\":
\"192.0.2.1\",\"ServerName\":\"example.com\",\"ServerPath\":\"/login\"}" \
    --client-metadata="{\"MyExampleKey\": \"MyExampleValue\"}"
```

Saída:

```
{
    "ChallengeName": "SOFTWARE_TOKEN_MFA",
    "Session": "AYABeExample...",
    "ChallengeParameters": {
        "FRIENDLY_DEVICE_NAME": "MyAuthenticatorApp",
        "USER_ID_FOR_SRP": "diego@example.com"
    }
}
```

Consulte mais informações em Fluxo de autenticação de administração no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte AdminInitiateAuthem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
   public static AdminInitiateAuthResponse
initiateAuth(CognitoIdentityProviderClient identityProviderClient,
           String clientId, String userName, String password, String userPoolId)
{
       try {
           Map<String, String> authParameters = new HashMap<>();
           authParameters.put("USERNAME", userName);
           authParameters.put("PASSWORD", password);

           AdminInitiateAuthRequest authRequest =
AdminInitiateAuthRequest.builder()
                   .clientId(clientId)
                   .userPoolId(userPoolId)
                   .authParameters(authParameters)
                   .authFlow(AuthFlowType.ADMIN_USER_PASSWORD_AUTH)
                   .build();

           AdminInitiateAuthResponse response =
identityProviderClient.adminInitiateAuth(authRequest);
           System.out.println("Result Challenge is : " +
response.challengeName());
           return response;

       } catch (CognitoIdentityProviderException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }

       return null;
   }
```

- Para obter detalhes da API, consulte AdminInitiateAutha Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const adminInitiateAuth = ({ clientId, userPoolId, username, password }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new AdminInitiateAuthCommand({
    ClientId: clientId,
    UserPoolId: userPoolId,
    AuthFlow: AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
    AuthParameters: { USERNAME: username, PASSWORD: password },
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte AdminInitiateAutha Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
suspend fun checkAuthMethod(
    clientIdVal: String,
    userNameVal: String,
    passwordVal: String,
    userPoolIdVal: String,
): AdminInitiateAuthResponse {
    val authParas = mutableMapOf<String, String>()
    authParas["USERNAME"] = userNameVal
    authParas["PASSWORD"] = passwordVal

    val authRequest =
        AdminInitiateAuthRequest {
            clientId = clientIdVal
            userPoolId = userPoolIdVal
            authParameters = authParas
            authFlow = AuthFlowType.AdminUserPasswordAuth
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.adminInitiateAuth(authRequest)
        println("Result Challenge is ${response.challengeName}")
        return response
    }
}
```

- Para obter detalhes da API, consulte a AdminInitiateAuthreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class CognitoIdentityProviderWrapper:
```

```
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def start_sign_in(self, user_name, password):
        """
        Starts the sign-in process for a user by using administrator credentials.
        This method of signing in is appropriate for code running on a secure
 server.

        If the user pool is configured to require MFA and this is the first sign-
in
        for the user, Amazon Cognito returns a challenge response to set up an
        MFA application. When this occurs, this function gets an MFA secret from
        Amazon Cognito and returns it to the caller.

        :param user_name: The name of the user to sign in.
        :param password: The user's password.
        :return: The result of the sign-in attempt. When sign-in is successful,
 this
                 returns an access token that can be used to get AWS credentials.
 Otherwise,
                 Amazon Cognito returns a challenge to set up an MFA application,
                 or a challenge to enter an MFA code from a registered MFA
 application.
        """
        try:
            kwargs = {
                "UserPoolId": self.user_pool_id,
                "ClientId": self.client_id,
```

```
                "AuthFlow": "ADMIN_USER_PASSWORD_AUTH",
                "AuthParameters": {"USERNAME": user_name, "PASSWORD": password},
            }
            if self.client_secret is not None:
                kwargs["AuthParameters"]["SECRET_HASH"] =
self._secret_hash(user_name)
            response = self.cognito_idp_client.admin_initiate_auth(**kwargs)
            challenge_name = response.get("ChallengeName", None)
            if challenge_name == "MFA_SETUP":
                if (
                    "SOFTWARE_TOKEN_MFA"
                    in response["ChallengeParameters"]["MFAS_CAN_SETUP"]
                ):
                    response.update(self.get_mfa_secret(response["Session"]))
                else:
                    raise RuntimeError(
                        "The user pool requires MFA setup, but the user pool is
not "
                        "configured for TOTP MFA. This example requires TOTP
MFA."
                    )
        except ClientError as err:
            logger.error(
                "Couldn't start sign in for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response
```

- Para obter detalhes da API, consulte a AdminInitiateAuthReferência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    TRY.
        " Set up authentication parameters
        DATA(lt_auth_params) = VALUE /aws1/
cl_cgpauthparamstype_w=>tt_authparameterstype(
          ( VALUE /aws1/cl_cgpauthparamstype_w=>ts_authparameterstype_maprow(
              key = 'USERNAME'
              value = NEW /aws1/cl_cgpauthparamstype_w( iv_user_name ) ) )
          ( VALUE /aws1/cl_cgpauthparamstype_w=>ts_authparameterstype_maprow(
              key = 'PASSWORD'
              value = NEW /aws1/cl_cgpauthparamstype_w( iv_password ) ) )
        ).

        " Add SECRET_HASH if provided
        IF iv_secret_hash IS NOT INITIAL.
          INSERT VALUE #(
            key = 'SECRET_HASH'
            value = NEW /aws1/cl_cgpauthparamstype_w( iv_secret_hash )
          ) INTO TABLE lt_auth_params.
        ENDIF.

        oo_result = lo_cgp->admininitiateauth(
          iv_userpoolid = iv_user_pool_id
          iv_clientid = iv_client_id
          iv_authflow = 'ADMIN_USER_PASSWORD_AUTH'
          it_authparameters = lt_auth_params
        ).

        DATA(lv_challenge) = oo_result->get_challengename( ).

        IF lv_challenge IS INITIAL.
          MESSAGE 'User successfully signed in.' TYPE 'I'.
        ELSE.
```

```
          MESSAGE |Authentication challenge required: { lv_challenge }.| TYPE
'I'.
        ENDIF.

      CATCH /aws1/cx_cgpusernotfoundex INTO DATA(lo_user_ex).
        MESSAGE |User { iv_user_name } not found.| TYPE 'E'.

      CATCH /aws1/cx_cgpnotauthorizedex INTO DATA(lo_auth_ex).
        MESSAGE 'Not authorized. Check credentials.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte a AdminInitiateAuthreferência da API AWS SDK for SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Begin an authentication session.
    ///
    /// - Parameters:
    ///   - cipClient: The `CongitoIdentityProviderClient` to use.
    ///   - clientId: The app client ID to use.
    ///   - userName: The username to check.
    ///   - password: The user's password.
    ///   - userPoolId: The user pool to use.
    ///
    /// - Returns: The session token associated with this authentication
    ///   session.
    func initiateAuth(cipClient: CognitoIdentityProviderClient, clientId: String,
                      userName: String, password: String,
```

```
                        userPoolId: String) async -> String? {
        var authParams: [String: String] = [:]

        authParams["USERNAME"] = userName
        authParams["PASSWORD"] = password

        do {
            let output = try await cipClient.adminInitiateAuth(
                input: AdminInitiateAuthInput(
                    authFlow:
CognitoIdentityProviderClientTypes.AuthFlowType.adminUserPasswordAuth,
                    authParameters: authParams,
                    clientId: clientId,
                    userPoolId: userPoolId
                )
            )

            guard let challengeName = output.challengeName else {
                print("*** Invalid response from the auth service.")
                return nil
            }

            print("=====> Response challenge is \(challengeName)")

            return output.session
        } catch _ as UserNotFoundException {
            print("*** The specified username, \(userName), doesn't exist.")
            return nil
        } catch _ as UserNotConfirmedException {
            print("*** The user \(userName) has not been confirmed.")
            return nil
        } catch {
            print("*** An unexpected error occurred.")
            return nil
        }
    }
```

- Para obter detalhes da API, consulte AdminInitiateAutha referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`AdminRespondToAuthChallenge`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `AdminRespondToAuthChallenge`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
/// <summary>
/// Respond to an admin authentication challenge.
/// </summary>
/// <param name="userName">The name of the user.</param>
/// <param name="clientId">The client ID.</param>
/// <param name="mfaCode">The multi-factor authentication code.</param>
/// <param name="session">The current application session.</param>
/// <param name="clientId">The user pool ID.</param>
/// <returns>The result of the authentication response.</returns>
public async Task<AuthenticationResultType> AdminRespondToAuthChallengeAsync(
    string userName,
    string clientId,
    string mfaCode,
    string session,
    string userPoolId)
{
    Console.WriteLine("SOFTWARE_TOKEN_MFA challenge is generated");
```

```
        var challengeResponses = new Dictionary<string, string>();
        challengeResponses.Add("USERNAME", userName);
        challengeResponses.Add("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

        var respondToAuthChallengeRequest = new
AdminRespondToAuthChallengeRequest
        {
            ChallengeName = ChallengeNameType.SOFTWARE_TOKEN_MFA,
            ClientId = clientId,
            ChallengeResponses = challengeResponses,
            Session = session,
            UserPoolId = userPoolId,
        };

        var response = await
_cognitoService.AdminRespondToAuthChallengeAsync(respondToAuthChallengeRequest);
        Console.WriteLine($"Response to Authentication
{response.AuthenticationResult.TokenType}");
        return response.AuthenticationResult;
    }
```

- Para obter detalhes da API, consulte AdminRespondToAuthChallengea Referência AWS SDK para .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";
```

```
   Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

       Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeRequest
request;
       request.AddChallengeResponses("USERNAME", userName);
       request.AddChallengeResponses("SOFTWARE_TOKEN_MFA_CODE", mfaCode);
       request.SetChallengeName(

Aws::CognitoIdentityProvider::Model::ChallengeNameType::SOFTWARE_TOKEN_MFA);
       request.SetClientId(clientID);
       request.SetUserPoolId(userPoolID);
       request.SetSession(session);

       Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeOutcome
outcome =
               client.AdminRespondToAuthChallenge(request);

       if (outcome.IsSuccess()) {
           std::cout << "Here is the response to the challenge.\n" <<

outcome.GetResult().GetAuthenticationResult().Jsonize().View().WriteReadable()
                     << std::endl;

           accessToken =
outcome.GetResult().GetAuthenticationResult().GetAccessToken();
       }
       else {
           std::cerr << "Error with
CognitoIdentityProvider::AdminRespondToAuthChallenge. "
                     << outcome.GetError().GetMessage()
                     << std::endl;
           return false;
       }
```

- Para obter detalhes da API, consulte AdminRespondToAuthChallengea Referência AWS SDK para C++ da API.

CLI

AWS CLI

Como responder a um desafio de autenticação

Há muitas maneiras de responder a diferentes desafios de autenticação, dependendo do fluxo de autenticação, da configuração do grupo de usuários e das configurações do usuário. O exemplo de `admin-respond-to-auth-challenge` a seguir fornece um código de MFA TOTP para diego@example.com e conclui o login. Esse grupo de usuários tem a memorização de dispositivos ativada, então o resultado da autenticação também retorna uma nova chave de dispositivo.

```
aws cognito-idp admin-respond-to-auth-challenge \
    --user-pool-id us-west-2_EXAMPLE \
    --client-id 1example23456789 \
    --challenge-name SOFTWARE_TOKEN_MFA \
    --challenge-
responses USERNAME=diego@example.com,SOFTWARE_TOKEN_MFA_CODE=000000 \
    --session AYABeExample...
```

Saída:

```
{
    "ChallengeParameters": {},
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "eyJra123abcEXAMPLE",
        "IdToken": "eyJra789ghiEXAMPLE",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-ExAmPlE1"
        }
    }
}
```

Consulte mais informações em Fluxo de autenticação de administração no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte AdminRespondToAuthChallengeem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
   // Respond to an authentication challenge.
   public static void adminRespondToAuthChallenge(CognitoIdentityProviderClient
identityProviderClient,
           String userName, String clientId, String mfaCode, String session) {
       System.out.println("SOFTWARE_TOKEN_MFA challenge is generated");
       Map<String, String> challengeResponses = new HashMap<>();

       challengeResponses.put("USERNAME", userName);
       challengeResponses.put("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

       AdminRespondToAuthChallengeRequest respondToAuthChallengeRequest =
AdminRespondToAuthChallengeRequest.builder()
               .challengeName(ChallengeNameType.SOFTWARE_TOKEN_MFA)
               .clientId(clientId)
               .challengeResponses(challengeResponses)
               .session(session)
               .build();

       AdminRespondToAuthChallengeResponse respondToAuthChallengeResult =
identityProviderClient
               .adminRespondToAuthChallenge(respondToAuthChallengeRequest);

System.out.println("respondToAuthChallengeResult.getAuthenticationResult()"
               + respondToAuthChallengeResult.authenticationResult());
   }
```

- Para obter detalhes da API, consulte AdminRespondToAuthChallengea Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const adminRespondToAuthChallenge = ({
  userPoolId,
  clientId,
  username,
  totp,
  session,
}) => {
  const client = new CognitoIdentityProviderClient({});
  const command = new AdminRespondToAuthChallengeCommand({
    ChallengeName: ChallengeNameType.SOFTWARE_TOKEN_MFA,
    ChallengeResponses: {
      SOFTWARE_TOKEN_MFA_CODE: totp,
      USERNAME: username,
    },
    ClientId: clientId,
    UserPoolId: userPoolId,
    Session: session,
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte AdminRespondToAuthChallengea Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
// Respond to an authentication challenge.
suspend fun adminRespondToAuthChallenge(
    userName: String,
    clientIdVal: String?,
    mfaCode: String,
    sessionVal: String?,
) {
    println("SOFTWARE_TOKEN_MFA challenge is generated")
    val challengeResponsesOb = mutableMapOf<String, String>()
    challengeResponsesOb["USERNAME"] = userName
    challengeResponsesOb["SOFTWARE_TOKEN_MFA_CODE"] = mfaCode

    val adminRespondToAuthChallengeRequest =
        AdminRespondToAuthChallengeRequest {
            challengeName = ChallengeNameType.SoftwareTokenMfa
            clientId = clientIdVal
            challengeResponses = challengeResponsesOb
            session = sessionVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val respondToAuthChallengeResult =
 identityProviderClient.adminRespondToAuthChallenge(adminRespondToAuthChallengeRequest)
        println("respondToAuthChallengeResult.getAuthenticationResult()
 ${respondToAuthChallengeResult.authenticationResult}")
    }
}
```

- Para obter detalhes da API, consulte a AdminRespondToAuthChallengereferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Responda a um desafio de MFA fornecendo um código gerado por uma aplicação de MFA associada.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def respond_to_mfa_challenge(self, user_name, session, mfa_code):
        """
        Responds to a challenge for an MFA code. This completes the second step
 of
        a two-factor sign-in. When sign-in is successful, it returns an access
 token
        that can be used to get AWS credentials from Amazon Cognito.

        :param user_name: The name of the user who is signing in.
```

```
        :param session: Session information returned from a previous call to
initiate
                        authentication.
        :param mfa_code: A code generated by the associated MFA application.
        :return: The result of the authentication. When successful, this contains
an
                 access token for the user.
        """
        try:
            kwargs = {
                "UserPoolId": self.user_pool_id,
                "ClientId": self.client_id,
                "ChallengeName": "SOFTWARE_TOKEN_MFA",
                "Session": session,
                "ChallengeResponses": {
                    "USERNAME": user_name,
                    "SOFTWARE_TOKEN_MFA_CODE": mfa_code,
                },
            }
            if self.client_secret is not None:
                kwargs["ChallengeResponses"]["SECRET_HASH"] = self._secret_hash(
                    user_name
                )
            response =
self.cognito_idp_client.admin_respond_to_auth_challenge(**kwargs)
            auth_result = response["AuthenticationResult"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "ExpiredCodeException":
                logger.warning(
                    "Your MFA code has expired or has been used already. You
might have "
                    "to wait a few seconds until your app shows you a new code."
                )
            else:
                logger.error(
                    "Couldn't respond to mfa challenge for %s. Here's why: %s:
%s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        else:
            return auth_result
```

- Para obter detalhes da API, consulte a AdminRespondToAuthChallengeReferência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    TRY.
        " Build challenge responses
        DATA(lt_challenge_responses) = VALUE /aws1/
cl_cgpchallengerspstyp00=>tt_challengeresponsestype(
          ( VALUE /aws1/cl_cgpchallengerspstyp00=>ts_challengerspstype_maprow(
              key = 'USERNAME'
              value = NEW /aws1/cl_cgpchallengerspstyp00( iv_user_name ) ) )
          ( VALUE /aws1/cl_cgpchallengerspstyp00=>ts_challengerspstype_maprow(
              key = 'SOFTWARE_TOKEN_MFA_CODE'
              value = NEW /aws1/cl_cgpchallengerspstyp00( iv_mfa_code ) ) )
        ).

        " Add SECRET_HASH if provided
        IF iv_secret_hash IS NOT INITIAL.
          INSERT VALUE #(
            key = 'SECRET_HASH'
            value = NEW /aws1/cl_cgpchallengerspstyp00( iv_secret_hash )
          ) INTO TABLE lt_challenge_responses.
        ENDIF.

        DATA(lo_result) = lo_cgp->adminrespondtoauthchallenge(
          iv_userpoolid = iv_user_pool_id
          iv_clientid = iv_client_id
          iv_challengename = 'SOFTWARE_TOKEN_MFA'
          it_challengeresponses = lt_challenge_responses
```

```
          iv_session = iv_session
        ).

        oo_auth_result = lo_result->get_authenticationresult( ).

        IF oo_auth_result IS BOUND.
          MESSAGE 'MFA challenge completed successfully.' TYPE 'I'.
        ELSE.
          " Another challenge might be required
          DATA(lv_next_challenge) = lo_result->get_challengename( ).
          MESSAGE |Additional challenge required: { lv_next_challenge }.| TYPE
'I'.
        ENDIF.

      CATCH /aws1/cx_cgpcodemismatchex INTO DATA(lo_code_ex).
        MESSAGE 'Invalid MFA code provided.' TYPE 'E'.

      CATCH /aws1/cx_cgpexpiredcodeex INTO DATA(lo_expired_ex).
        MESSAGE 'MFA code has expired.' TYPE 'E'.

      CATCH /aws1/cx_cgpnotauthorizedex INTO DATA(lo_auth_ex).
        MESSAGE 'Not authorized. Check MFA configuration.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte a AdminRespondToAuthChallengereferência da API AWS SDK for SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSClientRuntime
import AWSCognitoIdentityProvider
```

```
    /// Respond to the authentication challenge received from Cognito after
    /// initiating an authentication session. This involves sending a current
    /// MFA code to the service.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - userName: The user's username.
    ///   - clientId: The app client ID.
    ///   - userPoolId: The user pool to sign into.
    ///   - mfaCode: The 6-digit MFA code currently displayed by the user's
    ///     authenticator.
    ///   - session: The authentication session to continue processing.
    func adminRespondToAuthChallenge(cipClient: CognitoIdentityProviderClient,
userName: String,
                                     clientId: String, userPoolId: String,
mfaCode: String,
                                     session: String) async {
        print("=====> SOFTWARE_TOKEN_MFA challenge is generated...")

        var challengeResponsesOb: [String: String] = [:]
        challengeResponsesOb["USERNAME"] = userName
        challengeResponsesOb["SOFTWARE_TOKEN_MFA_CODE"] = mfaCode

        do {
            let output = try await cipClient.adminRespondToAuthChallenge(
                input: AdminRespondToAuthChallengeInput(
                    challengeName:
CognitoIdentityProviderClientTypes.ChallengeNameType.softwareTokenMfa,
                    challengeResponses: challengeResponsesOb,
                    clientId: clientId,
                    session: session,
                    userPoolId: userPoolId
                )
            )

            guard let authenticationResult = output.authenticationResult else {
                print("*** Unable to get authentication result.")
                return
            }

            print("=====> Authentication result (JWTs are redacted):")
            print(authenticationResult)
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
```

```
            return
        } catch _ as CodeMismatchException {
            print("*** The specified MFA code doesn't match the expected value.")
            return
        } catch _ as UserNotFoundException {
            print("*** The specified username, \(userName), doesn't exist.")
            return
        } catch _ as UserNotConfirmedException {
            print("*** The user \(userName) has not been confirmed.")
            return
        } catch let error as NotAuthorizedException {
            print("*** Unauthorized access. Reason: \(error.properties.message ??
"<unknown>")")
        } catch {
            print("*** Error responding to the MFA challenge.")
            return
        }
    }
```

- Para obter detalhes da API, consulte AdminRespondToAuthChallengea referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`AdminSetUserPassword`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `AdminSetUserPassword`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Grave dados de atividades personalizados com uma função do Lambda após a autenticação do usuário do Amazon Cognito

CLI

AWS CLI

Como definir uma senha de usuário como administrador

O exemplo de `admin-set-user-password` a seguir define permanentemente a senha para diego@example.com.

```
aws cognito-idp admin-set-user-password \
    --user-pool-id us-west-2_EXAMPLE \
    --username diego@example.com \
    --password MyExamplePassword1! \
    --permanent
```

Este comando não produz saída.

Consulte mais informações em Passwords, password recovery, and password policies no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte AdminSetUserPasswordem Referência de AWS CLI Comandos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}
```

```
// AdminSetUserPassword uses administrator credentials to set a password for a
 user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(ctx context.Context, userPoolId
 string, userName string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(ctx,
 &cognitoidentityprovider.AdminSetUserPasswordInput{
  Password:   aws.String(password),
  UserPoolId: aws.String(userPoolId),
  Username:   aws.String(userName),
  Permanent:  true,
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
 err)
  }
 }
 return err
}
```

- Para obter detalhes da API, consulte AdminSetUserPassworda Referência AWS SDK para Go da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`AssociateSoftwareToken`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `AssociateSoftwareToken`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// Get an MFA token to authenticate the user with the authenticator.
    /// </summary>
    /// <param name="session">The session name.</param>
    /// <returns>The session name.</returns>
    public async Task<string> AssociateSoftwareTokenAsync(string session)
    {
        var softwareTokenRequest = new AssociateSoftwareTokenRequest
        {
            Session = session,
        };

        var tokenResponse = await
_cognitoService.AssociateSoftwareTokenAsync(softwareTokenRequest);
        var secretCode = tokenResponse.SecretCode;

        Console.WriteLine($"Use the following secret code to set up the
authenticator: {secretCode}");

        return tokenResponse.Session;
    }
```

- Para obter detalhes da API, consulte AssociateSoftwareTokena Referência AWS SDK para .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
 client(clientConfig);

        Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenRequest
 request;
        request.SetSession(session);

        Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenOutcome
 outcome =
                client.AssociateSoftwareToken(request);

        if (outcome.IsSuccess()) {
            std::cout
                    << "Enter this setup key into an authenticator app, for
 example Google Authenticator."
                    << std::endl;
            std::cout << "Setup key: " << outcome.GetResult().GetSecretCode()
                      << std::endl;
#ifdef USING_QR
            printAsterisksLine();
            std::cout << "\nOr scan the QR code in the file '" << QR_CODE_PATH <<
 "."
                      << std::endl;

            saveQRCode(std::string("otpauth://totp/") + userName + "?secret=" +
                       outcome.GetResult().GetSecretCode());
#endif // USING_QR
            session = outcome.GetResult().GetSession();
```

```
        }
        else {
            std::cerr << "Error with
CognitoIdentityProvider::AssociateSoftwareToken. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- Para obter detalhes da API, consulte AssociateSoftwareTokena Referência AWS SDK para C++ da API.

CLI

AWS CLI

Como gerar uma chave secreta para uma aplicação autenticadora de MFA

O exemplo de `associate-software-token` a seguir gera uma chave privada TOTP para um usuário que fez login e recebeu um token de acesso. A chave privada resultante pode ser inserida manualmente em uma aplicação autenticadora ou as aplicações podem renderizá-la como um código QR que o usuário pode escanear.

```
aws cognito-idp associate-software-token \
    --access-token eyJra456defEXAMPLE
```

Saída:

```
{
    "SecretCode": "QWERTYUIOP123456EXAMPLE"
}
```

Consulte mais informações em MFA de token de software TOTP no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte AssociateSoftwareTokenem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
   public static String getSecretForAppMFA(CognitoIdentityProviderClient
identityProviderClient, String session) {
       AssociateSoftwareTokenRequest softwareTokenRequest =
AssociateSoftwareTokenRequest.builder()
               .session(session)
               .build();

       AssociateSoftwareTokenResponse tokenResponse = identityProviderClient
               .associateSoftwareToken(softwareTokenRequest);
       String secretCode = tokenResponse.secretCode();
       System.out.println("Enter this token into Google Authenticator");
       System.out.println(secretCode);
       return tokenResponse.session();
   }
```

- Para obter detalhes da API, consulte AssociateSoftwareTokena Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const associateSoftwareToken = (session) => {
```

```
  const client = new CognitoIdentityProviderClient({});
  const command = new AssociateSoftwareTokenCommand({
    Session: session,
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte AssociateSoftwareTokena Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
suspend fun getSecretForAppMFA(sessionVal: String?): String? {
    val softwareTokenRequest =
        AssociateSoftwareTokenRequest {
            session = sessionVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val tokenResponse =
 identityProviderClient.associateSoftwareToken(softwareTokenRequest)
        val secretCode = tokenResponse.secretCode
        println("Enter this token into Google Authenticator")
        println(secretCode)
        return tokenResponse.session
    }
}
```

- Para obter detalhes da API, consulte a AssociateSoftwareTokenreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def get_mfa_secret(self, session):
        """
        Gets a token that can be used to associate an MFA application with the
 user.

        :param session: Session information returned from a previous call to
 initiate
                        authentication.
        :return: An MFA token that can be used to set up an MFA application.
```

```
        """
        try:
            response =
self.cognito_idp_client.associate_software_token(Session=session)
        except ClientError as err:
            logger.error(
                "Couldn't get MFA secret. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response
```

- Para obter detalhes da API, consulte a AssociateSoftwareTokenReferência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    TRY.
        DATA(lo_result) = lo_cgp->associatesoftwaretoken(
          iv_session = iv_session
        ).

        ov_secret_code = lo_result->get_secretcode( ).

        MESSAGE 'MFA secret code generated successfully.' TYPE 'I'.

      CATCH /aws1/cx_cgpresourcenotfoundex INTO DATA(lo_ex).
        MESSAGE 'Session not found or expired.' TYPE 'E'.
```

```
      CATCH /aws1/cx_cgpnotauthorizedex INTO DATA(lo_auth_ex).
        MESSAGE 'Not authorized to associate software token.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte a AssociateSoftwareTokenreferência da API AWS SDK for SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Request and display an MFA secret token that the user should enter
    /// into their authenticator to set it up for the user account.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - authSession: The authentication session to request an MFA secret
    ///     for.
    ///
    /// - Returns: A string containing the MFA secret token that should be
    ///   entered into the authenticator software.
    func getSecretForAppMFA(cipClient: CognitoIdentityProviderClient,
 authSession: String?) async -> String? {
        do {
            let output = try await cipClient.associateSoftwareToken(
                input: AssociateSoftwareTokenInput(
                    session: authSession
                )
            )
```

```
            guard let secretCode = output.secretCode else {
                print("*** Unable to get the secret code")
                return nil
            }

            print("=====> Enter this token into Google Authenticator:
\(secretCode)")
            return output.session
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
            return nil
        } catch {
            print("*** An unexpected error occurred getting the secret for the
app's MFA.")
            return nil
        }
    }
```

- Para obter detalhes da API, consulte AssociateSoftwareTokena referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`ConfirmDevice`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `ConfirmDevice`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// Initiates and confirms tracking of the device.
    /// </summary>
    /// <param name="accessToken">The user's access token.</param>
    /// <param name="deviceKey">The key of the device from Amazon Cognito.</
param>
    /// <param name="deviceName">The device name.</param>
    /// <returns></returns>
    public async Task<bool> ConfirmDeviceAsync(string accessToken, string
 deviceKey, string deviceName)
    {
        var request = new ConfirmDeviceRequest
        {
            AccessToken = accessToken,
            DeviceKey = deviceKey,
            DeviceName = deviceName
        };

        var response = await _cognitoService.ConfirmDeviceAsync(request);
        return response.UserConfirmationNecessary;
    }
```

- Para obter detalhes da API, consulte ConfirmDevicea Referência AWS SDK para .NET da API.

CLI

AWS CLI

Como confirmar um dispositivo de usuário

O exemplo de `confirm-device` a seguir adiciona um novo dispositivo memorizado para o usuário atual.

```
aws cognito-idp confirm-device \
   --access-token eyJra456defEXAMPLE \
   --device-key us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 \
   --device-secret-verifier-
config PasswordVerifier=TXlWZXJpZmllclN0cmluZw,Salt=TXlTUlBTYWx0
```

Saída:

```
{
     "UserConfirmationNecessary": false
}
```

Consulte mais informações em Trabalhar com dispositivos de usuários no grupo de usuários no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte ConfirmDeviceem Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const confirmDevice = ({ deviceKey, accessToken, passwordVerifier, salt }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ConfirmDeviceCommand({
    DeviceKey: deviceKey,
```

```
    AccessToken: accessToken,
    DeviceSecretVerifierConfig: {
      PasswordVerifier: passwordVerifier,
      Salt: salt,
    },
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte ConfirmDevicea Referência AWS SDK para JavaScript da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret
```

```
   def confirm_mfa_device(
       self,
       user_name,
       device_key,
       device_group_key,
       device_password,
       access_token,
       aws_srp,
   ):
       """
       Confirms an MFA device to be tracked by Amazon Cognito. When a device is
       tracked, its key and password can be used to sign in without requiring a
new
       MFA code from the MFA application.

       :param user_name: The user that is associated with the device.
       :param device_key: The key of the device, returned by Amazon Cognito.
       :param device_group_key: The group key of the device, returned by Amazon
Cognito.
       :param device_password: The password that is associated with the device.
       :param access_token: The user's access token.
       :param aws_srp: A class that helps with Secure Remote Password (SRP)
                       calculations. The scenario associated with this example
uses
                       the warrant package.
       :return: True when the user must confirm the device. Otherwise, False.
When
                False, the device is automatically confirmed and tracked.
       """
       srp_helper = aws_srp.AWSSRP(
           username=user_name,
           password=device_password,
           pool_id="_",
           client_id=self.client_id,
           client_secret=None,
           client=self.cognito_idp_client,
       )
       device_and_pw = f"{device_group_key}{device_key}:{device_password}"
       device_and_pw_hash = aws_srp.hash_sha256(device_and_pw.encode("utf-8"))
       salt = aws_srp.pad_hex(aws_srp.get_random(16))
       x_value = aws_srp.hex_to_long(aws_srp.hex_hash(salt +
device_and_pw_hash))
```

```
        verifier = aws_srp.pad_hex(pow(srp_helper.val_g, x_value,
srp_helper.big_n))
        device_secret_verifier_config = {
            "PasswordVerifier": base64.standard_b64encode(
                bytearray.fromhex(verifier)
            ).decode("utf-8"),
            "Salt":
base64.standard_b64encode(bytearray.fromhex(salt)).decode("utf-8"),
        }
        try:
            response = self.cognito_idp_client.confirm_device(
                AccessToken=access_token,
                DeviceKey=device_key,
                DeviceSecretVerifierConfig=device_secret_verifier_config,
            )
            user_confirm = response["UserConfirmationNecessary"]
        except ClientError as err:
            logger.error(
                "Couldn't confirm mfa device %s. Here's why: %s: %s",
                device_key,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return user_confirm
```

- Para obter detalhes da API, consulte a ConfirmDeviceReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`ConfirmForgotPassword`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `ConfirmForgotPassword`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Migre automaticamente usuários conhecidos com uma função do Lambda

CLI

AWS CLI

Para confirmar uma senha esquecida

Este exemplo confirma a inscrição para o nome de usuário diego@example.com.

Comando:

```
aws cognito-idp confirm-forgot-password --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username=diego@example.com --password PASSWORD --confirmation-code CONF_CODE
```

- Para obter detalhes da API, consulte ConfirmForgotPasswordem Referência de AWS CLI Comandos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

type CognitoActions struct {
```

```
 CognitoClient *cognitoidentityprovider.Client
}



// ConfirmForgotPassword confirms a user with a confirmation code and a new
 password.
func (actor CognitoActions) ConfirmForgotPassword(ctx context.Context, clientId
 string, code string, userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(ctx,
 &cognitoidentityprovider.ConfirmForgotPasswordInput{
  ClientId:         aws.String(clientId),
  ConfirmationCode: aws.String(code),
  Password:         aws.String(password),
  Username:         aws.String(userName),
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
  }
 }
 return err
}
```

- Para obter detalhes da API, consulte ConfirmForgotPassworda Referência AWS SDK para Go da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`ConfirmSignUp`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `ConfirmSignUp`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// Confirm that the user has signed up.
    /// </summary>
    /// <param name="clientId">The Id of this application.</param>
    /// <param name="code">The confirmation code sent to the user.</param>
    /// <param name="userName">The username.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ConfirmSignupAsync(string clientId, string code,
string userName)
    {
        var signUpRequest = new ConfirmSignUpRequest
        {
            ClientId = clientId,
            ConfirmationCode = code,
            Username = userName,
        };

        var response = await _cognitoService.ConfirmSignUpAsync(signUpRequest);
        if (response.HttpStatusCode == HttpStatusCode.OK)
        {
            Console.WriteLine($"{userName} was confirmed");
            return true;
        }
        return false;
    }
```

- Para obter detalhes da API, consulte ConfirmSignUpa Referência AWS SDK para .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

        Aws::CognitoIdentityProvider::Model::ConfirmSignUpRequest request;
        request.SetClientId(clientID);
        request.SetConfirmationCode(confirmationCode);
        request.SetUsername(userName);

        Aws::CognitoIdentityProvider::Model::ConfirmSignUpOutcome outcome =
                client.ConfirmSignUp(request);

        if (outcome.IsSuccess()) {
            std::cout << "ConfirmSignup was Successful."
                      << std::endl;
        }
        else {
            std::cerr << "Error with CognitoIdentityProvider::ConfirmSignUp. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- Para obter detalhes da API, consulte ConfirmSignUpa Referência AWS SDK para C++ da API.

CLI

AWS CLI

Como confirmar a inscrição

Este exemplo confirma a inscrição para o nome de usuário diego@example.com.

Comando:

```
aws cognito-idp confirm-sign-up --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username=diego@example.com --confirmation-code CONF_CODE
```

- Para obter detalhes da API, consulte ConfirmSignUpem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
   public static void confirmSignUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String code,
           String userName) {
       try {
           ConfirmSignUpRequest signUpRequest = ConfirmSignUpRequest.builder()
                   .clientId(clientId)
                   .confirmationCode(code)
                   .username(userName)
                   .build();

           identityProviderClient.confirmSignUp(signUpRequest);
           System.out.println(userName + " was confirmed");
```

```
        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Para obter detalhes da API, consulte ConfirmSignUpa Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const confirmSignUp = ({ clientId, username, code }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ConfirmSignUpCommand({
    ClientId: clientId,
    Username: username,
    ConfirmationCode: code,
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte ConfirmSignUpa Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
suspend fun confirmSignUp(
    clientIdVal: String?,
    codeVal: String?,
    userNameVal: String?,
) {
    val signUpRequest =
        ConfirmSignUpRequest {
            clientId = clientIdVal
            confirmationCode = codeVal
            username = userNameVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.confirmSignUp(signUpRequest)
        println("$userNameVal  was confirmed")
    }
}
```

- Para obter detalhes da API, consulte a ConfirmSignUpreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def confirm_user_sign_up(self, user_name, confirmation_code):
        """
        Confirms a previously created user. A user must be confirmed before they
        can sign in to Amazon Cognito.

        :param user_name: The name of the user to confirm.
        :param confirmation_code: The confirmation code sent to the user's
 registered
                                  email address.
        :return: True when the confirmation succeeds.
        """
        try:
            kwargs = {
```

```
                "ClientId": self.client_id,
                "Username": user_name,
                "ConfirmationCode": confirmation_code,
            }
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            self.cognito_idp_client.confirm_sign_up(**kwargs)
        except ClientError as err:
            logger.error(
                "Couldn't confirm sign up for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return True
```

- Para obter detalhes da API, consulte a ConfirmSignUpReferência da API AWS SDK for Python (Boto3).

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Submit a confirmation code for the specified user. This is the code as
    /// entered by the user after they've received it by email or text
    /// message.
    ///
    /// - Parameters:
```

```
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The app client ID the user is signing up for.
    ///   - userName: The username of the user whose code is being sent.
    ///   - code: The user's confirmation code.
    ///
    /// - Returns: `true` if the code was successfully confirmed; otherwise
`false`.
    func confirmSignUp(cipClient: CognitoIdentityProviderClient, clientId:
String,
                       userName: String, code: String) async -> Bool {
        do {
            _ = try await cipClient.confirmSignUp(
                input: ConfirmSignUpInput(
                    clientId: clientId,
                    confirmationCode: code,
                    username: userName
                )
            )

            print("=====> \(userName) has been confirmed.")
            return true
        } catch {
            print("=====> \(userName)'s code was entered incorrectly.")
            return false
        }
    }
```

- Para obter detalhes da API, consulte ConfirmSignUpa referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`CreateUserPool`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `CreateUserPool`.

CLI

AWS CLI

Como criar um grupo de usuários minimamente configurado

Este exemplo cria um grupo de usuários chamado MyUserPool usando valores padrão. Não há atributos nem clientes da aplicação obrigatórios. A MFA e a segurança avançada estão desabilitadas.

Comando:

```
aws cognito-idp create-user-pool --pool-name MyUserPool
```

Saída:

```
{
  "UserPool": {
      "SchemaAttributes": [
          {
              "Name": "sub",
              "StringAttributeConstraints": {
                  "MinLength": "1",
                  "MaxLength": "2048"
              },
              "DeveloperOnlyAttribute": false,
              "Required": true,
              "AttributeDataType": "String",
              "Mutable": false
          },
          {
              "Name": "name",
              "StringAttributeConstraints": {
                  "MinLength": "0",
                  "MaxLength": "2048"
              },
              "DeveloperOnlyAttribute": false,
              "Required": false,
              "AttributeDataType": "String",
              "Mutable": true
          },
          {
              "Name": "given_name",
              "StringAttributeConstraints": {
                  "MinLength": "0",
                  "MaxLength": "2048"
              },
              "DeveloperOnlyAttribute": false,
              "Required": false,
```

```
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "family_name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "middle_name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "nickname",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "preferred_username",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
```

```
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "profile",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "picture",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "website",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "email",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
```

```
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "AttributeDataType": "Boolean",
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "Name": "email_verified",
            "Mutable": true
        },
        {
            "Name": "gender",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "birthdate",
            "StringAttributeConstraints": {
                "MinLength": "10",
                "MaxLength": "10"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "zoneinfo",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
```

```
            "Name": "locale",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "phone_number",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "AttributeDataType": "Boolean",
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "Name": "phone_number_verified",
            "Mutable": true
        },
        {
            "Name": "address",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "updated_at",
            "NumberAttributeConstraints": {
                "MinValue": "0"
            },
```

```
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "Number",
                "Mutable": true
            }
        ],
        "MfaConfiguration": "OFF",
        "Name": "MyUserPool",
        "LastModifiedDate": 1547833345.777,
        "AdminCreateUserConfig": {
            "UnusedAccountValidityDays": 7,
            "AllowAdminCreateUserOnly": false
        },
        "EmailConfiguration": {},
        "Policies": {
            "PasswordPolicy": {
                "RequireLowercase": true,
                "RequireSymbols": true,
                "RequireNumbers": true,
                "MinimumLength": 8,
                "RequireUppercase": true
            }
        },
        "CreationDate": 1547833345.777,
        "EstimatedNumberOfUsers": 0,
        "Id": "us-west-2_aaaaaaaaa",
        "LambdaConfig": {}
    }
}
```

Como criar um grupo de usuários com dois atributos obrigatórios

Este exemplo cria um grupo de usuários MyUserPool. O grupo é configurado para aceitar o e-mail como o atributo de nome de usuário. Ele também define o endereço de origem do e-mail como um endereço validado usando o Amazon Simple Email Service.

Comando:

```
aws cognito-idp create-user-pool --pool-name MyUserPool --username-
attributes "email" --email-configuration=SourceArn="arn:aws:ses:us-
east-1:111111111111:identity/
jane@example.com",ReplyToEmailAddress="jane@example.com"
```

Saída:


```
{
  "UserPool": {
      "SchemaAttributes": [
          {
              "Name": "sub",
              "StringAttributeConstraints": {
                  "MinLength": "1",
                  "MaxLength": "2048"
              },
              "DeveloperOnlyAttribute": false,
              "Required": true,
              "AttributeDataType": "String",
              "Mutable": false
          },
          {
              "Name": "name",
              "StringAttributeConstraints": {
                  "MinLength": "0",
                  "MaxLength": "2048"
              },
              "DeveloperOnlyAttribute": false,
              "Required": false,
              "AttributeDataType": "String",
              "Mutable": true
          },
          {
              "Name": "given_name",
              "StringAttributeConstraints": {
                  "MinLength": "0",
                  "MaxLength": "2048"
              },
              "DeveloperOnlyAttribute": false,
              "Required": false,
              "AttributeDataType": "String",
              "Mutable": true
          },
          {
              "Name": "family_name",
              "StringAttributeConstraints": {
                  "MinLength": "0",
                  "MaxLength": "2048"
              },
```

```
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "middle_name",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "nickname",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "preferred_username",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "profile",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
```

```
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "picture",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "website",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "Name": "email",
            "StringAttributeConstraints": {
                "MinLength": "0",
                "MaxLength": "2048"
            },
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "AttributeDataType": "String",
            "Mutable": true
        },
        {
            "AttributeDataType": "Boolean",
            "DeveloperOnlyAttribute": false,
            "Required": false,
            "Name": "email_verified",
            "Mutable": true
```

```
            },
            {
                "Name": "gender",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "birthdate",
                "StringAttributeConstraints": {
                    "MinLength": "10",
                    "MaxLength": "10"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "zoneinfo",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "locale",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
```

```
            },
            {
                "Name": "phone_number",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "AttributeDataType": "Boolean",
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "Name": "phone_number_verified",
                "Mutable": true
            },
            {
                "Name": "address",
                "StringAttributeConstraints": {
                    "MinLength": "0",
                    "MaxLength": "2048"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "String",
                "Mutable": true
            },
            {
                "Name": "updated_at",
                "NumberAttributeConstraints": {
                    "MinValue": "0"
                },
                "DeveloperOnlyAttribute": false,
                "Required": false,
                "AttributeDataType": "Number",
                "Mutable": true
            }
        ],
        "MfaConfiguration": "OFF",
        "Name": "MyUserPool",
        "LastModifiedDate": 1547837788.189,
```

```
      "AdminCreateUserConfig": {
          "UnusedAccountValidityDays": 7,
          "AllowAdminCreateUserOnly": false
      },
      "EmailConfiguration": {
          "ReplyToEmailAddress": "jane@example.com",
          "SourceArn": "arn:aws:ses:us-east-1:111111111111:identity/
jane@example.com"
      },
      "Policies": {
          "PasswordPolicy": {
              "RequireLowercase": true,
              "RequireSymbols": true,
              "RequireNumbers": true,
              "MinimumLength": 8,
              "RequireUppercase": true
          }
      },
      "UsernameAttributes": [
          "email"
      ],
      "CreationDate": 1547837788.189,
      "EstimatedNumberOfUsers": 0,
      "Id": "us-west-2_aaaaaaaaa",
      "LambdaConfig": {}
  }
}
```

- Para obter detalhes da API, consulte CreateUserPoolem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolResponse;

/**
 * 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 CreateUserPool {
    public static void main(String[] args) {

        final String usage = """

                Usage:
                    <userPoolName>\s

                Where:
                    userPoolName - The name to give your user pool when it's
 created.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userPoolName = args[0];
        CognitoIdentityProviderClient cognitoClient =
 CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();
```

```
        String id = createPool(cognitoClient, userPoolName);
        System.out.println("User pool ID: " + id);
        cognitoClient.close();
    }

    public static String createPool(CognitoIdentityProviderClient cognitoClient,
 String userPoolName) {
        try {
            CreateUserPoolRequest request = CreateUserPoolRequest.builder()
                    .poolName(userPoolName)
                    .build();

            CreateUserPoolResponse response =
 cognitoClient.createUserPool(request);
            return response.userPool().id();

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
        return "";
    }
}
```

- Para obter detalhes da API, consulte CreateUserPoola Referência AWS SDK for Java 2.x da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`CreateUserPoolClient`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `CreateUserPoolClient`.

CLI

AWS CLI

Para criar um cliente de grupo de usuários

O `create-user-pool-client` exemplo a seguir cria um novo cliente de grupo de usuários com um segredo de cliente, atributos explícitos de leitura e gravação, login com fluxos de nome de usuário, senha e SRP, login com três, acesso a um subconjunto de OAuth escopos IdPs, PinPoint análises e uma validade estendida da sessão de autenticação.

```
aws cognito-idp create-user-pool-client \
    --user-pool-id us-west-2_EXAMPLE \
    --client-name MyTestClient \
    --generate-secret \
    --refresh-token-validity 10 \
    --access-token-validity 60 \
    --id-token-validity 60 \
    --token-validity-units AccessToken=minutes,IdToken=minutes,RefreshToken=days
 \
    --read-attributes email phone_number email_verified phone_number_verified \
    --write-attributes email phone_number \
    --explicit-auth-
flows ALLOW_USER_PASSWORD_AUTH ALLOW_USER_SRP_AUTH ALLOW_REFRESH_TOKEN_AUTH \
    --supported-identity-providers Google Facebook MyOIDC \
    --callback-urls https://www.amazon.com https://example.com http://
localhost:8001 myapp://example \
    --allowed-o-auth-flows code implicit \
    --allowed-o-auth-scopes openid profile aws.cognito.signin.user.admin solar-
system-data/asteroids.add \
    --allowed-o-auth-flows-user-pool-client \
    --analytics-configuration ApplicationArn=arn:aws:mobiletargeting:us-
west-2:767671399759:apps/thisisanexamplepinpointapplicationid,UserDataShared=TRUE
 \
    --prevent-user-existence-errors ENABLED \
    --enable-token-revocation \
    --enable-propagate-additional-user-context-data \
    --auth-session-validity 4
```

Saída:

```
{
    "UserPoolClient": {
        "UserPoolId": "us-west-2_EXAMPLE",
        "ClientName": "MyTestClient",
        "ClientId": "123abc456defEXAMPLE",
        "ClientSecret": "this1234is5678my91011example1213client1415secret",
        "LastModifiedDate": 1726788459.464,
```

```
        "CreationDate": 1726788459.464,
        "RefreshTokenValidity": 10,
        "AccessTokenValidity": 60,
        "IdTokenValidity": 60,
        "TokenValidityUnits": {
            "AccessToken": "minutes",
            "IdToken": "minutes",
            "RefreshToken": "days"
        },
        "ReadAttributes": [
            "email_verified",
            "phone_number_verified",
            "phone_number",
            "email"
        ],
        "WriteAttributes": [
            "phone_number",
            "email"
        ],
        "ExplicitAuthFlows": [
            "ALLOW_USER_PASSWORD_AUTH",
            "ALLOW_USER_SRP_AUTH",
            "ALLOW_REFRESH_TOKEN_AUTH"
        ],
        "SupportedIdentityProviders": [
            "Google",
            "MyOIDC",
            "Facebook"
        ],
        "CallbackURLs": [
            "https://example.com",
            "https://www.amazon.com",
            "myapp://example",
            "http://localhost:8001"
        ],
        "AllowedOAuthFlows": [
            "implicit",
            "code"
        ],
        "AllowedOAuthScopes": [
            "aws.cognito.signin.user.admin",
            "openid",
            "profile",
            "solar-system-data/asteroids.add"
```

```
        ],
        "AllowedOAuthFlowsUserPoolClient": true,
        "AnalyticsConfiguration": {
            "ApplicationArn": "arn:aws:mobiletargeting:us-
west-2:123456789012:apps/thisisanexamplepinpointapplicationid",
            "RoleArn": "arn:aws:iam::123456789012:role/aws-service-role/cognito-
idp.amazonaws.com/AWSServiceRoleForAmazonCognitoIdp",
            "UserDataShared": true
        },
        "PreventUserExistenceErrors": "ENABLED",
        "EnableTokenRevocation": true,
        "EnablePropagateAdditionalUserContextData": true,
        "AuthSessionValidity": 4
    }
}
```

Consulte mais informações em Application-specific settings with app clients no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte CreateUserPoolClientem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolClientReques
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CreateUserPoolClientRespon
```

```
/**
 * A user pool client app is an application that authenticates with Amazon
 * Cognito user pools.
 * When you create a user pool, you can configure app clients that allow mobile
 * or web applications
 * to call API operations to authenticate users, manage user attributes and
 * profiles,
 * and implement sign-up and sign-in flows.
 *
 * 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 CreateUserPoolClient {
    public static void main(String[] args) {
        final String usage = """

                Usage:
                    <clientName> <userPoolId>\s

                Where:
                    clientName - The name for the user pool client to create.
                    userPoolId - The ID for the user pool.
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String clientName = args[0];
        String userPoolId = args[1];
        CognitoIdentityProviderClient cognitoClient =
 CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();

        createPoolClient(cognitoClient, clientName, userPoolId);
        cognitoClient.close();
    }
```

```
    public static void createPoolClient(CognitoIdentityProviderClient
 cognitoClient, String clientName,
            String userPoolId) {
        try {
            CreateUserPoolClientRequest request =
 CreateUserPoolClientRequest.builder()
                    .clientName(clientName)
                    .userPoolId(userPoolId)
                    .build();

            CreateUserPoolClientResponse response =
 cognitoClient.createUserPoolClient(request);
            System.out.println("User pool " +
 response.userPoolClient().clientName() + " created. ID: "
                    + response.userPoolClient().clientId());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte CreateUserPoolClienta Referência AWS SDK for Java 2.x da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`DeleteUser`** com um AWS SDK ou 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:

- Confirme automaticamente usuários conhecidos com uma função do Lambda
- Migre automaticamente usuários conhecidos com uma função do Lambda

- Grave dados de atividades personalizados com uma função do Lambda após a autenticação do usuário do Amazon Cognito

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

        Aws::CognitoIdentityProvider::Model::DeleteUserRequest request;
        request.SetAccessToken(accessToken);

        Aws::CognitoIdentityProvider::Model::DeleteUserOutcome outcome =
                client.DeleteUser(request);

        if (outcome.IsSuccess()) {
            std::cout << "The user " << userName << " was deleted."
                      << std::endl;
        }
        else {
            std::cerr << "Error with CognitoIdentityProvider::DeleteUser. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
        }
```

- Para obter detalhes da API, consulte DeleteUsera Referência AWS SDK para C++ da API.

CLI

AWS CLI

Como excluir um usuário

Este exemplo exclui um usuário.

Comando:

```
aws cognito-idp delete-user --access-token ACCESS_TOKEN
```

- Para obter detalhes da API, consulte DeleteUserem Referência de AWS CLI Comandos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}
```

```
// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(ctx context.Context, userAccessToken
 string) error {
 _, err := actor.CognitoClient.DeleteUser(ctx,
 &cognitoidentityprovider.DeleteUserInput{
  AccessToken: aws.String(userAccessToken),
 })
 if err != nil {
  log.Printf("Couldn't delete user. Here's why: %v\n", err)
 }
 return err
}
```

- Para obter detalhes da API, consulte DeleteUsera Referência AWS SDK para Go da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
/**
 * Delete the signed-in user. Useful for allowing a user to delete their
 * own profile.
 * @param {{ region: string, accessToken: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").DeleteUserCommandOutput | null, unknown]>}
 */
export const deleteUser = async ({ region, accessToken }) => {
  try {
    const client = new CognitoIdentityProviderClient({ region });
    const response = await client.send(
      new DeleteUserCommand({ AccessToken: accessToken }),
    );
    return [response, null];
  } catch (err) {
```

```
    return [null, err];
  }
};
```

- Para obter detalhes da API, consulte DeleteUsera Referência AWS SDK para JavaScript da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`ForgotPassword`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `ForgotPassword`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Migre automaticamente usuários conhecidos com uma função do Lambda

CLI

AWS CLI

Para forçar uma alteração de senha

O `forgot-password` exemplo a seguir envia uma mensagem para jane@example.com para alterar a senha.

```
aws cognito-idp forgot-password --client-id 38fjsnc484p94kpqsnet7mpld0 --
username jane@example.com
```

Saída:

```
{
    "CodeDeliveryDetails": {
        "Destination": "j***@e***.com",
        "DeliveryMedium": "EMAIL",
        "AttributeName": "email"
```

```
    }
}
```

- Para obter detalhes da API, consulte ForgotPasswordem Referência de AWS CLI Comandos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// ForgotPassword starts a password recovery flow for a user. This flow typically
 sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(ctx context.Context, clientId string,
 userName string) (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(ctx,
 &cognitoidentityprovider.ForgotPasswordInput{
  ClientId: aws.String(clientId),
```

```
  Username: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't start password reset for user '%v'. Here;s why: %v\n",
 userName, err)
 }
 return output.CodeDeliveryDetails, err
}
```

- Para obter detalhes da API, consulte ForgotPassworda Referência AWS SDK para Go da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`InitiateAuth`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `InitiateAuth`.

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:

- Confirme automaticamente usuários conhecidos com uma função do Lambda
- Migre automaticamente usuários conhecidos com uma função do Lambda
- Inscrever um usuário em um grupo de usuários que exija MFA
- Grave dados de atividades personalizados com uma função do Lambda após a autenticação do usuário do Amazon Cognito

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// Initiate authorization.
    /// </summary>
    /// <param name="clientId">The client Id of the application.</param>
    /// <param name="userName">The name of the user who is authenticating.</
param>
    /// <param name="password">The password for the user who is authenticating.</
param>
    /// <returns>The response from the initiate auth request.</returns>
    public async Task<InitiateAuthResponse> InitiateAuthAsync(string clientId,
 string userName, string password)
    {
        var authParameters = new Dictionary<string, string>();
        authParameters.Add("USERNAME", userName);
        authParameters.Add("PASSWORD", password);

        var authRequest = new InitiateAuthRequest

        {
            ClientId = clientId,
            AuthParameters = authParameters,
            AuthFlow = AuthFlowType.USER_PASSWORD_AUTH,
        };

        var response = await _cognitoService.InitiateAuthAsync(authRequest);
        Console.WriteLine($"Result Challenge is : {response.ChallengeName}");

        return response;
    }
```

- Para obter detalhes da API, consulte InitiateAutha Referência AWS SDK para .NET da API.

CLI

AWS CLI

Como conectar um usuário

O exemplo de `initiate-auth` a seguir conecta um usuário com o fluxo básico de nome de usuário e senha e sem desafios adicionais.

```
aws cognito-idp initiate-auth \
    --auth-flow USER_PASSWORD_AUTH \
    --client-id 1example23456789 \
    --analytics-metadata AnalyticsEndpointId=d70b2ba36a8c4dc5a04a0451aEXAMPLE \
    --auth-parameters USERNAME=testuser,PASSWORD=[Password] --user-context-
data EncodedData=mycontextdata --client-metadata MyTestKey=MyTestValue
```

Saída:

```
{
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "eyJra123abcEXAMPLE",
        "IdToken": "eyJra789ghiEXAMPLE",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-v7w9UcY6"
        }
    }
}
```

Consulte mais informações em Authentication no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte InitiateAuthem Referência de AWS CLI Comandos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import (
 "context"
```

```
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// SignIn signs in a user to Amazon Cognito using a username and password
 authentication flow.
func (actor CognitoActions) SignIn(ctx context.Context, clientId string, userName
 string, password string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(ctx,
 &cognitoidentityprovider.InitiateAuthInput{
  AuthFlow:       "USER_PASSWORD_AUTH",
  ClientId:       aws.String(clientId),
  AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
 })
 if err != nil {
  var resetRequired *types.PasswordResetRequiredException
  if errors.As(err, &resetRequired) {
   log.Println(*resetRequired.Message)
  } else {
   log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
  }
 } else {
  authResult = output.AuthenticationResult
 }
 return authResult, err
}
```

- Para obter detalhes da API, consulte InitiateAutha Referência AWS SDK para Go da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const initiateAuth = ({ username, password, clientId }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new InitiateAuthCommand({
    AuthFlow: AuthFlowType.USER_PASSWORD_AUTH,
    AuthParameters: {
      USERNAME: username,
      PASSWORD: password,
    },
    ClientId: clientId,
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte InitiateAutha Referência AWS SDK para JavaScript da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Este exemplo mostra como iniciar a autenticação com um dispositivo rastreado. Para concluir o login, o cliente deve responder corretamente aos desafios de Secure Remote Password (SRP).

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def sign_in_with_tracked_device(
        self,
        user_name,
        password,
        device_key,
        device_group_key,
        device_password,
        aws_srp,
    ):
        """
        Signs in to Amazon Cognito as a user who has a tracked device. Signing in
        with a tracked device lets a user sign in without entering a new MFA
 code.

        Signing in with a tracked device requires that the client respond to the
 SRP
        protocol. The scenario associated with this example uses the warrant
 package
        to help with SRP calculations.
```

```
        For more information on SRP, see https://en.wikipedia.org/wiki/
Secure_Remote_Password_protocol.

        :param user_name: The user that is associated with the device.
        :param password: The user's password.
        :param device_key: The key of a tracked device.
        :param device_group_key: The group key of a tracked device.
        :param device_password: The password that is associated with the device.
        :param aws_srp: A class that helps with SRP calculations. The scenario
                        associated with this example uses the warrant package.
        :return: The result of the authentication. When successful, this contains
 an
                 access token for the user.
        """
        try:
            srp_helper = aws_srp.AWSSRP(
                username=user_name,
                password=device_password,
                pool_id="_",
                client_id=self.client_id,
                client_secret=None,
                client=self.cognito_idp_client,
            )

            response_init = self.cognito_idp_client.initiate_auth(
                ClientId=self.client_id,
                AuthFlow="USER_PASSWORD_AUTH",
                AuthParameters={
                    "USERNAME": user_name,
                    "PASSWORD": password,
                    "DEVICE_KEY": device_key,
                },
            )
            if response_init["ChallengeName"] != "DEVICE_SRP_AUTH":
                raise RuntimeError(
                    f"Expected DEVICE_SRP_AUTH challenge but got
 {response_init['ChallengeName']}."
                )

            auth_params = srp_helper.get_auth_params()
            auth_params["DEVICE_KEY"] = device_key
            response_auth = self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_SRP_AUTH",
```

```
                ChallengeResponses=auth_params,
            )
            if response_auth["ChallengeName"] != "DEVICE_PASSWORD_VERIFIER":
                raise RuntimeError(
                    f"Expected DEVICE_PASSWORD_VERIFIER challenge but got "
                    f"{response_init['ChallengeName']}."
                )

            challenge_params = response_auth["ChallengeParameters"]
            challenge_params["USER_ID_FOR_SRP"] = device_group_key + device_key
            cr = srp_helper.process_challenge(challenge_params, {"USERNAME":
user_name})
            cr["USERNAME"] = user_name
            cr["DEVICE_KEY"] = device_key
            response_verifier =
self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_PASSWORD_VERIFIER",
                ChallengeResponses=cr,
            )
            auth_tokens = response_verifier["AuthenticationResult"]
        except ClientError as err:
            logger.error(
                "Couldn't start client sign in for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return auth_tokens
```

- Para obter detalhes da API, consulte a InitiateAuthReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`ListUserPools`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `ListUserPools`.

.NET

SDK para .NET (v4)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// List the Amazon Cognito user pools for an account.
    /// </summary>
    /// <returns>A list of UserPoolDescriptionType objects.</returns>
    public async Task<List<UserPoolDescriptionType>> ListUserPoolsAsync()
    {
        var userPools = new List<UserPoolDescriptionType>();

        var userPoolsPaginator = _cognitoService.Paginators.ListUserPools(new
ListUserPoolsRequest());

        await foreach (var response in userPoolsPaginator.Responses)
        {
            userPools.AddRange(response.UserPools);
        }

        return userPools;
    }
```

- Para obter detalhes da API, consulte ListUserPoolsa Referência AWS SDK para .NET da API.

CLI

AWS CLI

Para listar grupos de usuários

O `list-user-pools` exemplo a seguir lista três dos grupos de usuários disponíveis na AWS conta das credenciais atuais da CLI.

```
aws cognito-idp list-user-pools \
    --max-results 3
```

Saída:

```
{
    "NextToken": "[Pagination token]",
    "UserPools": [
        {
            "CreationDate": 1681502497.741,
            "Id": "us-west-2_EXAMPLE1",
            "LambdaConfig": {
                "CustomMessage": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                "PreSignUp": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                "PreTokenGeneration": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                "PreTokenGenerationConfig": {
                    "LambdaArn": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction",
                    "LambdaVersion": "V1_0"
                }
            },
            "LastModifiedDate": 1681502497.741,
            "Name": "user pool 1"
        },
        {
            "CreationDate": 1686064178.717,
            "Id": "us-west-2_EXAMPLE2",
            "LambdaConfig": {
            },
            "LastModifiedDate": 1686064178.873,
            "Name": "user pool 2"
```

```
        },
        {
            "CreationDate": 1627681712.237,
            "Id": "us-west-2_EXAMPLE3",
            "LambdaConfig": {
                "UserMigration": "arn:aws:lambda:us-
east-1:123456789012:function:MyFunction"
            },
            "LastModifiedDate": 1678486942.479,
            "Name": "user pool 3"
        }
    ]
}
```

Para obter mais informações, consulte Grupos de usuários do Amazon Cognito no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte ListUserPoolsem Referência de AWS CLI Comandos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
package main

import (
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)
```

```
// main uses the AWS SDK for Go V2 to create an Amazon Simple Notification
 Service
// (Amazon SNS) client and list the topics in your account.
// This example uses the default settings specified in your shared credentials
// and config files.
func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 cognitoClient := cognitoidentityprovider.NewFromConfig(sdkConfig)
 fmt.Println("Let's list the user pools for your account.")
 var pools []types.UserPoolDescriptionType
 paginator := cognitoidentityprovider.NewListUserPoolsPaginator(
  cognitoClient, &cognitoidentityprovider.ListUserPoolsInput{MaxResults:
 aws.Int32(10)})
 for paginator.HasMorePages() {
  output, err := paginator.NextPage(ctx)
  if err != nil {
   log.Printf("Couldn't get user pools. Here's why: %v\n", err)
  } else {
   pools = append(pools, output.UserPools...)
  }
 }
 if len(pools) == 0 {
  fmt.Println("You don't have any user pools!")
 } else {
  for _, pool := range pools {
   fmt.Printf("\t%v: %v\n", *pool.Name, *pool.Id)
  }
 }
}
```

- Para obter detalhes da API, consulte ListUserPoolsa Referência AWS SDK para Go da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUserPoolsRequest;

/**
 * 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 ListUserPools {
    public static void main(String[] args) {
        CognitoIdentityProviderClient cognitoClient =
 CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listAllUserPools(cognitoClient);
        cognitoClient.close();
    }

    public static void listAllUserPools(CognitoIdentityProviderClient
 cognitoClient) {
```

```
        try {
            ListUserPoolsRequest request = ListUserPoolsRequest.builder()
                    .maxResults(10)
                    .build();

            ListUserPoolsResponse response =
 cognitoClient.listUserPools(request);
            response.userPools().forEach(userpool -> {
                System.out.println("User pool " + userpool.name() + ", User ID "
 + userpool.id());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte ListUserPoolsa Referência AWS SDK for Java 2.x da API.

Rust

SDK para Rust

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
async fn show_pools(client: &Client) -> Result<(), Error> {
    let response = client.list_user_pools().max_results(10).send().await?;
    let pools = response.user_pools();
    println!("User pools:");
    for pool in pools {
        println!("  ID:             {}", pool.id().unwrap_or_default());
        println!("  Name:           {}", pool.name().unwrap_or_default());
        println!("  Lambda Config:  {:?}", pool.lambda_config().unwrap());
```

```
        println!(
            "  Last modified:   {}",
            pool.last_modified_date().unwrap().to_chrono_utc()?
        );
        println!(
            "  Creation date:   {:?}",
            pool.creation_date().unwrap().to_chrono_utc()
        );
        println!();
    }
    println!("Next token: {}", response.next_token().unwrap_or_default());

    Ok(())
}
```

- Para obter detalhes da API, consulte a ListUserPoolsreferência da API AWS SDK for Rust.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`ListUsers`** com um AWS SDK ou 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 em contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// Get a list of users for the Amazon Cognito user pool.
    /// </summary>
    /// <param name="userPoolId">The user pool ID.</param>
    /// <returns>A list of users.</returns>
    public async Task<List<UserType>> ListUsersAsync(string userPoolId)
    {
        var request = new ListUsersRequest
        {
            UserPoolId = userPoolId
        };

        var users = new List<UserType>();

        var usersPaginator = _cognitoService.Paginators.ListUsers(request);
        await foreach (var response in usersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }
```

- Para obter detalhes da API, consulte ListUsersa Referência AWS SDK para .NET da API.

CLI

AWS CLI

Exemplo 1: como listar usuários com um filtro do lado do servidor

O exemplo de `list-users` a seguir lista três usuários no grupo de usuários solicitado cujos endereços de e-mail começam com `testuser`.

```
aws cognito-idp list-users \
    --user-pool-id us-west-2_EXAMPLE \
    --filter email^=\"testuser\" \
    --max-items 3
```

Saída:

```
{
    "PaginationToken": "efgh5678EXAMPLE",
    "Users": [
        {
            "Attributes": [
                {
                    "Name": "sub",
                    "Value": "eaad0219-2117-439f-8d46-4db20e59268f"
                },
                {
                    "Name": "email",
                    "Value": "testuser@example.com"
                }
            ],
            "Enabled": true,
            "UserCreateDate": 1682955829.578,
            "UserLastModifiedDate": 1689030181.63,
            "UserStatus": "CONFIRMED",
            "Username": "testuser"
        },
        {
            "Attributes": [
                {
                    "Name": "sub",
                    "Value": "3b994cfd-0b07-4581-be46-3c82f9a70c90"
                },
                {
                    "Name": "email",
                    "Value": "testuser2@example.com"
                }
            ],
            "Enabled": true,
            "UserCreateDate": 1684427979.201,
            "UserLastModifiedDate": 1684427979.201,
            "UserStatus": "UNCONFIRMED",
            "Username": "testuser2"
        },
        {
            "Attributes": [
                {
                    "Name": "sub",
                    "Value": "5929e0d1-4c34-42d1-9b79-a5ecacfe66f7"
                },
```

```
                {
                    "Name": "email",
                    "Value": "testuser3@example.com"
                }
            ],
            "Enabled": true,
            "UserCreateDate": 1684427823.641,
            "UserLastModifiedDate": 1684427823.641,
            "UserStatus": "UNCONFIRMED",
            "Username": "testuser3@example.com"
        }
    ]
}
```

Consulte mais informações em Managing and searching for users no Guia do desenvolvedor do Amazon Cognito.

Exemplo 2: como listar usuários com um filtro do lado do cliente

O exemplo de `list-users` a seguir lista os atributos de três usuários que têm um atributo, nesse caso, o endereço de e-mail, que contém o domínio de e-mail "@example.com". Se outros atributos contivessem essa string, eles também seriam exibidos. O segundo usuário não tem atributos que correspondam à consulta e é excluído da saída exibida, mas não da resposta do servidor.

```
aws cognito-idp list-users \
    --user-pool-id us-west-2_EXAMPLE \
    --max-items 3
    --query Users\[\*\].Attributes\[\?Value\.contains\(\@\,\'@example.com\'\)\]
```

Saída:

```
[
    [
        {
            "Name": "email",
            "Value": "admin@example.com"
        }
    ],
    [],
    [
        {
```

```
            "Name": "email",
            "Value": "operator@example.com"
        }
    ]
]
```

Consulte mais informações em Managing and searching for users no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte ListUsersem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUsersRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ListUsersResponse;

/**
 * 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) {
```

```
        final String usage = """

                Usage:
                    <userPoolId>\s

                Where:
                    userPoolId - The ID given to your user pool when it's
created.
                """;

        if (args.length != 1) {
            System.out.println(usage);
            System.exit(1);
        }

        String userPoolId = args[0];
        CognitoIdentityProviderClient cognitoClient =
CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();

        listAllUsers(cognitoClient, userPoolId);
        listUsersFilter(cognitoClient, userPoolId);
        cognitoClient.close();
    }

    public static void listAllUsers(CognitoIdentityProviderClient cognitoClient,
String userPoolId) {
        try {
            ListUsersRequest usersRequest = ListUsersRequest.builder()
                    .userPoolId(userPoolId)
                    .build();

            ListUsersResponse response = cognitoClient.listUsers(usersRequest);
            response.users().forEach(user -> {
                System.out.println("User " + user.username() + " Status " +
user.userStatus() + " Created "
                        + user.userCreateDate());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
```

```
        }
    }

    // Shows how to list users by using a filter.
    public static void listUsersFilter(CognitoIdentityProviderClient
 cognitoClient, String userPoolId) {

        try {
            String filter = "email = \"tblue@noserver.com\"";
            ListUsersRequest usersRequest = ListUsersRequest.builder()
                    .userPoolId(userPoolId)
                    .filter(filter)
                    .build();

            ListUsersResponse response = cognitoClient.listUsers(usersRequest);
            response.users().forEach(user -> {
                System.out.println("User with filter applied " + user.username()
 + " Status " + user.userStatus()
                        + " Created " + user.userCreateDate());
            });

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
}
```

- Para obter detalhes da API, consulte ListUsersa Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const listUsers = ({ userPoolId }) => {
```

```
  const client = new CognitoIdentityProviderClient({});

  const command = new ListUsersCommand({
    UserPoolId: userPoolId,
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte ListUsersa Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
suspend fun listAllUsers(userPoolId: String) {
    val request =
        ListUsersRequest {
            this.userPoolId = userPoolId
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { cognitoClient ->
        val response = cognitoClient.listUsers(request)
        response.users?.forEach { user ->
            println("The user name is ${user.username}")
        }
    }
}
```

- Para obter detalhes da API, consulte a ListUsersreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def list_users(self):
        """
        Returns a list of the users in the current user pool.

        :return: The list of users.
        """
        try:
            response =
 self.cognito_idp_client.list_users(UserPoolId=self.user_pool_id)
            users = response["Users"]
        except ClientError as err:
            logger.error(
                "Couldn't list users for %s. Here's why: %s: %s",
```

```
                self.user_pool_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return users
```

- Para obter detalhes da API, consulte a ListUsersReferência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
TRY.
    DATA(lo_result) = lo_cgp->listusers(
      iv_userpoolid = iv_user_pool_id
    ).

    ot_users = lo_result->get_users( ).

    MESSAGE |Found { lines( ot_users ) } users in the pool.| TYPE 'I'.

  CATCH /aws1/cx_cgpresourcenotfoundex INTO DATA(lo_ex).
    MESSAGE |User pool { iv_user_pool_id } not found.| TYPE 'E'.

  CATCH /aws1/cx_cgpnotauthorizedex INTO DATA(lo_auth_ex).
    MESSAGE 'Not authorized to list users.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a ListUsersreferência da API AWS SDK for SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        do {
            let output = try await cognitoClient.listUsers(
                input: ListUsersInput(
                    userPoolId: poolId
                )
            )

            guard let users = output.users else {
                print("No users found.")
                return
            }

            print("\(users.count) user(s) found.")
            for user in users {
                print("  \(user.username ?? "<unknown>")")
            }
        } catch _ as NotAuthorizedException {
            print("*** Please authenticate with AWS before using this command.")
            return
        } catch _ as ResourceNotFoundException {
            print("*** The specified User Pool was not found.")
            return
        } catch {
            print("*** An unexpected type of error occurred.")
            return
        }
```

- Para obter detalhes da API, consulte ListUsersa referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`ResendConfirmationCode`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `ResendConfirmationCode`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// Send a new confirmation code to a user.
    /// </summary>
    /// <param name="clientId">The Id of the client application.</param>
    /// <param name="userName">The username of user who will receive the code.</
param>
    /// <returns>The delivery details.</returns>
    public async Task<CodeDeliveryDetailsType> ResendConfirmationCodeAsync(string
 clientId, string userName)
    {
        var codeRequest = new ResendConfirmationCodeRequest
        {
            ClientId = clientId,
            Username = userName,
        };
```

```
        var response = await
_cognitoService.ResendConfirmationCodeAsync(codeRequest);

        Console.WriteLine($"Method of delivery is
{response.CodeDeliveryDetails.DeliveryMedium}");

        return response.CodeDeliveryDetails;
    }
```

- Para obter detalhes da API, consulte ResendConfirmationCodea Referência AWS SDK para .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

        Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeRequest
request;
        request.SetUsername(userName);
        request.SetClientId(clientID);

        Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeOutcome
outcome =
                client.ResendConfirmationCode(request);

        if (outcome.IsSuccess()) {
```

```
            std::cout
                    << "CognitoIdentityProvider::ResendConfirmationCode was
successful."
                    << std::endl;
        }
        else {
            std::cerr << "Error with
CognitoIdentityProvider::ResendConfirmationCode. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- Para obter detalhes da API, consulte ResendConfirmationCodea Referência AWS SDK para C++ da API.

CLI

AWS CLI

Como reenviar um código de confirmação

O exemplo `resend-confirmation-code` a seguir envia um código de confirmação ao usuário `jane`.

```
aws cognito-idp resend-confirmation-code \
    --client-id 12a3b456c7de890f11g123hijk \
    --username jane
```

Saída:

```
{
    "CodeDeliveryDetails": {
        "Destination": "j***@e***.com",
        "DeliveryMedium": "EMAIL",
        "AttributeName": "email"
    }
}
```

Para obter mais informações, consulte Como cadastrar e confirmar contas de usuários no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte ResendConfirmationCodeem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
   public static void resendConfirmationCode(CognitoIdentityProviderClient
identityProviderClient, String clientId,
           String userName) {
       try {
           ResendConfirmationCodeRequest codeRequest =
ResendConfirmationCodeRequest.builder()
                   .clientId(clientId)
                   .username(userName)
                   .build();

           ResendConfirmationCodeResponse response =
identityProviderClient.resendConfirmationCode(codeRequest);
           System.out.println("Method of delivery is " +
response.codeDeliveryDetails().deliveryMediumAsString());

       } catch (CognitoIdentityProviderException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
           System.exit(1);
       }
   }
```

- Para obter detalhes da API, consulte ResendConfirmationCodea Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const resendConfirmationCode = ({ clientId, username }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ResendConfirmationCodeCommand({
    ClientId: clientId,
    Username: username,
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte ResendConfirmationCodea Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
suspend fun resendConfirmationCode(
    clientIdVal: String?,
    userNameVal: String?,
) {
    val codeRequest =
```

```
        ResendConfirmationCodeRequest {
            clientId = clientIdVal
            username = userNameVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.resendConfirmationCode(codeRequest)
        println("Method of delivery is " +
 (response.codeDeliveryDetails?.deliveryMedium))
    }
}
```

- Para obter detalhes da API, consulte a ResendConfirmationCodereferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
```

```
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def resend_confirmation(self, user_name):
        """
        Prompts Amazon Cognito to resend an email with a new confirmation code.

        :param user_name: The name of the user who will receive the email.
        :return: Delivery information about where the email is sent.
        """
        try:
            kwargs = {"ClientId": self.client_id, "Username": user_name}
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            response = self.cognito_idp_client.resend_confirmation_code(**kwargs)
            delivery = response["CodeDeliveryDetails"]
        except ClientError as err:
            logger.error(
                "Couldn't resend confirmation to %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return delivery
```

- Para obter detalhes da API, consulte a ResendConfirmationCodeReferência da API AWS SDK for Python (Boto3).

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Requests a new confirmation code be sent to the given user's contact
    /// method.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The application client ID.
    ///   - userName: The user to resend a code for.
    ///
    /// - Returns: `true` if a new code was sent successfully, otherwise
    ///   `false`.
    func resendConfirmationCode(cipClient: CognitoIdentityProviderClient,
 clientId: String,
                                userName: String) async -> Bool {
        do {
            let output = try await cipClient.resendConfirmationCode(
                input: ResendConfirmationCodeInput(
                    clientId: clientId,
                    username: userName
                )
            )

            guard let deliveryMedium = output.codeDeliveryDetails?.deliveryMedium
 else {
                print("*** Unable to get the delivery method for the resent
 code.")
                return false
            }

            print("=====> A new code has been sent by \(deliveryMedium)")
```

```
            return true
        } catch {
            print("*** Unable to resend the confirmation code to user
\(userName).")
            return false
        }
    }
```

- Para obter detalhes da API, consulte ResendConfirmationCodea referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`RespondToAuthChallenge`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `RespondToAuthChallenge`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

CLI

AWS CLI

Exemplo 1: como responder ao desafio NEW_PASSWORD_REQUIRED

O exemplo de `respond-to-auth-challenge` a seguir responde a um desafio NEW_PASSWORD_REQUIRED que initiate-auth retornou. Ele define uma senha para o usuário `jane@example.com`.

```
aws cognito-idp respond-to-auth-challenge \
    --client-id 1example23456789 \
    --challenge-name NEW_PASSWORD_REQUIRED \
    --challenge-responses USERNAME=jane@example.com,NEW_PASSWORD=[Password] \
    --session AYABeEv5HklEXAMPLE
```

Saída:

```
{
    "ChallengeParameters": {},
    "AuthenticationResult": {
        "AccessToken": "ACCESS_TOKEN",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "REFRESH_TOKEN",
        "IdToken": "ID_TOKEN",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-wt2ha1Zd"
        }
    }
}
```

Consulte mais informações em Authentication no Guia do desenvolvedor do Amazon Cognito.

Exemplo 2: como responder a um desafio SELECT_MFA_TYPE

O exemplo de `respond-to-auth-challenge` a seguir escolhe a MFA TOTP como a opção de MFA para o usuário atual. O usuário foi solicitado a selecionar um tipo de MFA e, depois, será solicitado a inserir o código da MFA.

```
aws cognito-idp respond-to-auth-challenge \
    --client-id 1example23456789
    --session AYABeEv5HklEXAMPLE
    --challenge-name SELECT_MFA_TYPE
    --challenge-responses USERNAME=testuser,ANSWER=SOFTWARE_TOKEN_MFA
```

Saída:

```
{
    "ChallengeName": "SOFTWARE_TOKEN_MFA",
    "Session": "AYABeEv5HklEXAMPLE",
    "ChallengeParameters": {
        "FRIENDLY_DEVICE_NAME": "transparent"
    }
}
```

Consulte mais informações em Adding MFA no Guia do desenvolvedor do Amazon Cognito.

Exemplo 3: como responder a um desafio SOFTWARE_TOKEN_MFA

O exemplo de `respond-to-auth-challenge` a seguir fornece um código de MFA TOTP e conclui o login.

```
aws cognito-idp respond-to-auth-challenge \
    --client-id 1example23456789 \
    --session AYABeEv5HklEXAMPLE \
    --challenge-name SOFTWARE_TOKEN_MFA \
    --challenge-responses USERNAME=testuser,SOFTWARE_TOKEN_MFA_CODE=123456
```

Saída:

```
{
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "TokenType": "Bearer",
        "RefreshToken": "eyJra123abcEXAMPLE",
        "IdToken": "eyJra789ghiEXAMPLE",
        "NewDeviceMetadata": {
            "DeviceKey": "us-west-2_a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "DeviceGroupKey": "-v7w9UcY6"
        }
    }
}
```

Consulte mais informações em Adding MFA no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte RespondToAuthChallengeem Referência de AWS CLI Comandos.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const respondToAuthChallenge = ({
  clientId,
  username,
  session,
  userPoolId,
  code,
}) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new RespondToAuthChallengeCommand({
    ChallengeName: ChallengeNameType.SOFTWARE_TOKEN_MFA,
    ChallengeResponses: {
      SOFTWARE_TOKEN_MFA_CODE: code,
      USERNAME: username,
    },
    ClientId: clientId,
    UserPoolId: userPoolId,
    Session: session,
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte RespondToAuthChallengea Referência AWS SDK para JavaScript da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Faça login com um dispositivo rastreado. Para concluir o login, o cliente deve responder corretamente aos desafios de Secure Remote Password (SRP).

```
class CognitoIdentityProviderWrapper:
```

```
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def sign_in_with_tracked_device(
        self,
        user_name,
        password,
        device_key,
        device_group_key,
        device_password,
        aws_srp,
    ):
        """
        Signs in to Amazon Cognito as a user who has a tracked device. Signing in
        with a tracked device lets a user sign in without entering a new MFA
 code.

        Signing in with a tracked device requires that the client respond to the
 SRP
        protocol. The scenario associated with this example uses the warrant
 package
        to help with SRP calculations.

        For more information on SRP, see https://en.wikipedia.org/wiki/
Secure_Remote_Password_protocol.

        :param user_name: The user that is associated with the device.
        :param password: The user's password.
        :param device_key: The key of a tracked device.
```

```
        :param device_group_key: The group key of a tracked device.
        :param device_password: The password that is associated with the device.
        :param aws_srp: A class that helps with SRP calculations. The scenario
                        associated with this example uses the warrant package.
        :return: The result of the authentication. When successful, this contains
an
                 access token for the user.
        """
        try:
            srp_helper = aws_srp.AWSSRP(
                username=user_name,
                password=device_password,
                pool_id="_",
                client_id=self.client_id,
                client_secret=None,
                client=self.cognito_idp_client,
            )

            response_init = self.cognito_idp_client.initiate_auth(
                ClientId=self.client_id,
                AuthFlow="USER_PASSWORD_AUTH",
                AuthParameters={
                    "USERNAME": user_name,
                    "PASSWORD": password,
                    "DEVICE_KEY": device_key,
                },
            )
            if response_init["ChallengeName"] != "DEVICE_SRP_AUTH":
                raise RuntimeError(
                    f"Expected DEVICE_SRP_AUTH challenge but got
{response_init['ChallengeName']}."
                )

            auth_params = srp_helper.get_auth_params()
            auth_params["DEVICE_KEY"] = device_key
            response_auth = self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_SRP_AUTH",
                ChallengeResponses=auth_params,
            )
            if response_auth["ChallengeName"] != "DEVICE_PASSWORD_VERIFIER":
                raise RuntimeError(
                    f"Expected DEVICE_PASSWORD_VERIFIER challenge but got "
                    f"{response_init['ChallengeName']}."
```

```
                )

            challenge_params = response_auth["ChallengeParameters"]
            challenge_params["USER_ID_FOR_SRP"] = device_group_key + device_key
            cr = srp_helper.process_challenge(challenge_params, {"USERNAME":
user_name})
            cr["USERNAME"] = user_name
            cr["DEVICE_KEY"] = device_key
            response_verifier =
self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_PASSWORD_VERIFIER",
                ChallengeResponses=cr,
            )
            auth_tokens = response_verifier["AuthenticationResult"]
        except ClientError as err:
            logger.error(
                "Couldn't start client sign in for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return auth_tokens
```

- Para obter detalhes da API, consulte a RespondToAuthChallengeReferência da API AWS SDK for Python (Boto3).

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`SignUp`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `SignUp`.

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:

- Confirme automaticamente usuários conhecidos com uma função do Lambda
- Migre automaticamente usuários conhecidos com uma função do Lambda
- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// Sign up a new user.
    /// </summary>
    /// <param name="clientId">The client Id of the application.</param>
    /// <param name="userName">The username to use.</param>
    /// <param name="password">The user's password.</param>
    /// <param name="email">The email address of the user.</param>
    /// <returns>A Boolean value indicating whether the user was confirmed.</
returns>
    public async Task<bool> SignUpAsync(string clientId, string userName, string
 password, string email)
    {
        var userAttrs = new AttributeType
        {
            Name = "email",
            Value = email,
        };

        var userAttrsList = new List<AttributeType>();

        userAttrsList.Add(userAttrs);

        var signUpRequest = new SignUpRequest
        {
            UserAttributes = userAttrsList,
            Username = userName,
            ClientId = clientId,
```

```
            Password = password
        };

        var response = await _cognitoService.SignUpAsync(signUpRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }
```

- Para obter detalhes da API, consulte SignUpa Referência AWS SDK para .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

        Aws::CognitoIdentityProvider::Model::SignUpRequest request;
        request.AddUserAttributes(
                Aws::CognitoIdentityProvider::Model::AttributeType().WithName(
                        "email").WithValue(email));
        request.SetUsername(userName);
        request.SetPassword(password);
        request.SetClientId(clientID);
        Aws::CognitoIdentityProvider::Model::SignUpOutcome outcome =
                client.SignUp(request);

        if (outcome.IsSuccess()) {
            std::cout << "The signup request for " << userName << " was
successful."
                      << std::endl;
```

```
        }
        else if (outcome.GetError().GetErrorType() ==

Aws::CognitoIdentityProvider::CognitoIdentityProviderErrors::USERNAME_EXISTS) {
            std::cout
                    << "The username already exists. Please enter a different
username."
                    << std::endl;
            userExists = true;
        }
        else {
            std::cerr << "Error with CognitoIdentityProvider::SignUpRequest. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- Para obter detalhes da API, consulte SignUpa Referência AWS SDK para C++ da API.

CLI

AWS CLI

Como inscrever um usuário

Este exemplo inscreve jane@example.com.

Comando:

```
aws cognito-idp sign-up --client-id 3n4b5urk1ft4fl3mg5e62d9ado --
username jane@example.com --password PASSWORD --user-attributes
 Name="email",Value="jane@example.com" Name="name",Value="Jane"
```

Saída:

```
{
  "UserConfirmed": false,
  "UserSub": "e04d60a6-45dc-441c-a40b-e25a787d4862"
}
```

- Para obter detalhes da API, consulte SignUpem Referência de AWS CLI Comandos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(ctx context.Context, clientId string, userName
 string, password string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(ctx,
 &cognitoidentityprovider.SignUpInput{
  ClientId: aws.String(clientId),
  Password: aws.String(password),
  Username: aws.String(userName),
  UserAttributes: []types.AttributeType{
   {Name: aws.String("email"), Value: aws.String(userEmail)},
  },
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
```

```
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
  }
 } else {
  confirmed = output.UserConfirmed
 }
 return confirmed, err
}
```

- Para obter detalhes da API, consulte SignUpa Referência AWS SDK para Go da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
   public static void signUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String userName,
           String password, String email) {
       AttributeType userAttrs = AttributeType.builder()
               .name("email")
               .value(email)
               .build();

       List<AttributeType> userAttrsList = new ArrayList<>();
       userAttrsList.add(userAttrs);
       try {
           SignUpRequest signUpRequest = SignUpRequest.builder()
                   .userAttributes(userAttrsList)
                   .username(userName)
                   .clientId(clientId)
                   .password(password)
                   .build();
```

```
            identityProviderClient.signUp(signUpRequest);
            System.out.println("User has been signed up ");

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Para obter detalhes da API, consulte SignUpa Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const signUp = ({ clientId, username, password, email }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new SignUpCommand({
    ClientId: clientId,
    Username: username,
    Password: password,
    UserAttributes: [{ Name: "email", Value: email }],
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte SignUpa Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
suspend fun signUp(
    clientIdVal: String?,
    userNameVal: String?,
    passwordVal: String?,
    emailVal: String?,
) {
    val userAttrs =
        AttributeType {
            name = "email"
            value = emailVal
        }

    val userAttrsList = mutableListOf<AttributeType>()
    userAttrsList.add(userAttrs)
    val signUpRequest =
        SignUpRequest {
            userAttributes = userAttrsList
            username = userNameVal
            clientId = clientIdVal
            password = passwordVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.signUp(signUpRequest)
        println("User has been signed up")
    }
}
```

- Para obter detalhes da API, consulte a SignUpreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def sign_up_user(self, user_name, password, user_email):
        """
        Signs up a new user with Amazon Cognito. This action prompts Amazon
 Cognito
        to send an email to the specified email address. The email contains a
 code that
        can be used to confirm the user.

        When the user already exists, the user status is checked to determine
 whether
        the user has been confirmed.

        :param user_name: The user name that identifies the new user.
```

```
        :param password: The password for the new user.
        :param user_email: The email address for the new user.
        :return: True when the user is already confirmed with Amazon Cognito.
                 Otherwise, false.
        """
        try:
            kwargs = {
                "ClientId": self.client_id,
                "Username": user_name,
                "Password": password,
                "UserAttributes": [{"Name": "email", "Value": user_email}],
            }
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            response = self.cognito_idp_client.sign_up(**kwargs)
            confirmed = response["UserConfirmed"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "UsernameExistsException":
                response = self.cognito_idp_client.admin_get_user(
                    UserPoolId=self.user_pool_id, Username=user_name
                )
                logger.warning(
                    "User %s exists and is %s.", user_name,
response["UserStatus"]
                )
                confirmed = response["UserStatus"] == "CONFIRMED"
            else:
                logger.error(
                    "Couldn't sign up %s. Here's why: %s: %s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return confirmed
```

- Para obter detalhes da API, consulte a SignUpReferência da API AWS SDK for Python (Boto3).

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Create a new user in a user pool.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The ID of the app client to create a user for.
    ///   - userName: The username for the new user.
    ///   - password: The new user's password.
    ///   - email: The new user's email address.
    ///
    /// - Returns: `true` if successful; otherwise `false`.
    func signUp(cipClient: CognitoIdentityProviderClient, clientId: String,
 userName: String, password: String, email: String) async -> Bool {
        let emailAttr = CognitoIdentityProviderClientTypes.AttributeType(
            name: "email",
            value: email
        )

        let userAttrsList = [emailAttr]

        do {
            _ = try await cipClient.signUp(
                input: SignUpInput(
                    clientId: clientId,
                    password: password,
                    userAttributes: userAttrsList,
                    username: userName
                )

            )
```

```
            print("=====> User \(userName) signed up.")
        } catch _ as AWSCognitoIdentityProvider.UsernameExistsException {
            print("*** The username \(userName) already exists. Please use a
different one.")
            return false
        } catch let error as AWSCognitoIdentityProvider.InvalidPasswordException
{
            print("*** Error: The specified password is invalid. Reason:
\(error.properties.message ?? "<none available>").")
            return false
        } catch _ as AWSCognitoIdentityProvider.ResourceNotFoundException {
            print("*** Error: The specified client ID (\(clientId)) doesn't
exist.")
            return false
        } catch {
            print("*** Unexpected error: \(error)")
            return false
        }

        return true
    }
```

- Para obter detalhes da API, consulte SignUpa referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`UpdateUserPool`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `UpdateUserPool`.

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:

- Confirme automaticamente usuários conhecidos com uma função do Lambda
- Migre automaticamente usuários conhecidos com uma função do Lambda
- Grave dados de atividades personalizados com uma função do Lambda após a autenticação do usuário do Amazon Cognito

CLI

AWS CLI

Para atualizar um grupo de usuários

O `update-user-pool` exemplo a seguir modifica um grupo de usuários com um exemplo de sintaxe para cada uma das opções de configuração disponíveis. Para atualizar um grupo de usuários, você deve especificar todas as opções configuradas anteriormente ou elas serão redefinidas para um valor padrão.

```
aws cognito-idp update-user-pool --user-pool-id us-west-2_EXAMPLE \
    --policies PasswordPolicy=
\{MinimumLength=6,RequireUppercase=true,RequireLowercase=true,RequireNumbers=true,Require
 \
    --deletion-protection ACTIVE \
    --lambda-config PreSignUp="arn:aws:lambda:us-
west-2:123456789012:function:cognito-test-presignup-
function",PreTokenGeneration="arn:aws:lambda:us-
west-2:123456789012:function:cognito-test-pretoken-function" \
    --auto-verified-attributes "phone_number" "email" \
    --verification-message-template \{\"SmsMessage\":\""Your code is
 {####}"\",\"EmailMessage\":\""Your code is {####}"\",\"EmailSubject\":\""Your
 verification code"\",\"EmailMessageByLink\":\""Click {##here##} to verify
 your email address."\",\"EmailSubjectByLink\":\""Your verification link"\",
\"DefaultEmailOption\":\"CONFIRM_WITH_LINK\"\} \
    --sms-authentication-message "Your code is {####}" \
    --user-attribute-update-settings
 AttributesRequireVerificationBeforeUpdate="email","phone_number" \
    --mfa-configuration "OPTIONAL" \
    --device-
configuration ChallengeRequiredOnNewDevice=true,DeviceOnlyRememberedOnUserPrompt=true
 \
    --email-configuration SourceArn="arn:aws:ses:us-
west-2:123456789012:identity/admin@example.com",ReplyToEmailAddress="amdin
+noreply@example.com",EmailSendingAccount=DEVELOPER,From="admin@amazon.com",Configuration
configuration-set" \
    --sms-configuration SnsCallerArn="arn:aws:iam::123456789012:role/service-
role/SNS-SMS-Role",ExternalId="12345",SnsRegion="us-west-2" \
    --admin-create-user-config
 AllowAdminCreateUserOnly=false,InviteMessageTemplate=\{SMSMessage=\""Welcome
 {username}. Your confirmation code is {####}"\",EmailMessage=\""Welcome
```

```
 {username}. Your confirmation code is {####}"\",EmailSubject=\""Welcome to
 MyMobileGame"\"\} \
    --user-pool-tags "Function"="MyMobileGame","Developers"="Berlin" \
    --admin-create-user-config
 AllowAdminCreateUserOnly=false,InviteMessageTemplate=\{SMSMessage=\""Welcome
 {username}. Your confirmation code is {####}"\",EmailMessage=\""Welcome
 {username}. Your confirmation code is {####}"\",EmailSubject=\""Welcome to
 MyMobileGame"\"\} \
    --user-pool-add-ons AdvancedSecurityMode="AUDIT" \
    --account-recovery-setting RecoveryMechanisms=
\[\{Priority=1,Name="verified_email"\},
\{Priority=2,Name="verified_phone_number"\}\]
```

Este comando não produz saída.

Para obter mais informações, consulte Configurar um cliente de aplicação de grupo de usuários no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte UpdateUserPoolem Referência de AWS CLI Comandos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)
```

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
 trigger.
type Trigger int

const (
 PreSignUp Trigger = iota
 UserMigration
 PostAuthentication
)

type TriggerInfo struct {
 Trigger    Trigger
 HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
 is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(ctx context.Context, userPoolId
 string, triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(ctx,
 &cognitoidentityprovider.DescribeUserPoolInput{
  UserPoolId: aws.String(userPoolId),
 })
 if err != nil {
  log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
 userPoolId, err)
  return err
 }
 lambdaConfig := output.UserPool.LambdaConfig
 for _, trigger := range triggers {
  switch trigger.Trigger {
  case PreSignUp:
   lambdaConfig.PreSignUp = trigger.HandlerArn
  case UserMigration:
   lambdaConfig.UserMigration = trigger.HandlerArn
  case PostAuthentication:
   lambdaConfig.PostAuthentication = trigger.HandlerArn
```

```
  }
 }
 _, err = actor.CognitoClient.UpdateUserPool(ctx,
 &cognitoidentityprovider.UpdateUserPoolInput{
  UserPoolId:   aws.String(userPoolId),
  LambdaConfig: lambdaConfig,
 })
 if err != nil {
  log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
 }
 return err
}
```

- Para obter detalhes da API, consulte UpdateUserPoola Referência AWS SDK para Go da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
/**
 * Connect a Lambda function to the PreSignUp trigger for a Cognito user pool
 * @param {{ region: string, userPoolId: string, handlerArn: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").UpdateUserPoolCommandOutput | null, unknown]>}
 */
export const addPreSignUpHandler = async ({
  region,
  userPoolId,
  handlerArn,
}) => {
  try {
    const cognitoClient = new CognitoIdentityProviderClient({
```

```
      region,
    });

    const command = new UpdateUserPoolCommand({
      UserPoolId: userPoolId,
      LambdaConfig: {
        PreSignUp: handlerArn,
      },
    });

    const response = await cognitoClient.send(command);
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};
```

- Para obter detalhes da API, consulte UpdateUserPoola Referência AWS SDK para JavaScript da API.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Use **`VerifySoftwareToken`** com um AWS SDK ou CLI

Os exemplos de código a seguir mostram como usar o `VerifySoftwareToken`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação em contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    /// <summary>
    /// Verify the TOTP and register for MFA.
    /// </summary>
    /// <param name="session">The name of the session.</param>
    /// <param name="code">The MFA code.</param>
    /// <returns>The status of the software token.</returns>
    public async Task<VerifySoftwareTokenResponseType>
VerifySoftwareTokenAsync(string session, string code)
    {
        var tokenRequest = new VerifySoftwareTokenRequest
        {
            UserCode = code,
            Session = session,
        };

        var verifyResponse = await
_cognitoService.VerifySoftwareTokenAsync(tokenRequest);

        return verifyResponse.Status;
    }
```

- Para obter detalhes da API, consulte VerifySoftwareTokena Referência AWS SDK para .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

    Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);

        Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenRequest request;
        request.SetUserCode(userCode);
        request.SetSession(session);

        Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenOutcome outcome =
                client.VerifySoftwareToken(request);

        if (outcome.IsSuccess()) {
            std::cout << "Verification of the code was successful."
                      << std::endl;
            session = outcome.GetResult().GetSession();
        }
        else {
            std::cerr << "Error with
CognitoIdentityProvider::VerifySoftwareToken. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
```

- Para obter detalhes da API, consulte VerifySoftwareTokena Referência AWS SDK para C++ da API.

CLI

AWS CLI

Como confirmar o registro de um autenticador TOTP

O exemplo de `verify-software-token` a seguir completa o registro TOTP para o usuário atual.

```
aws cognito-idp verify-software-token \
    --access-token eyJra456defEXAMPLE \
    --user-code 123456
```

Saída:

```
{
    "Status": "SUCCESS"
}
```

Para obter mais informações, consulte Adicionar MFA a um grupo de usuários no Guia do desenvolvedor do Amazon Cognito.

- Para obter detalhes da API, consulte VerifySoftwareTokenem Referência de AWS CLI Comandos.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    // Verify the TOTP and register for MFA.
    public static void verifyTOTP(CognitoIdentityProviderClient
identityProviderClient, String session, String code) {
        try {
```

```
            VerifySoftwareTokenRequest tokenRequest =
 VerifySoftwareTokenRequest.builder()
                    .userCode(code)
                    .session(session)
                    .build();

            VerifySoftwareTokenResponse verifyResponse =
 identityProviderClient.verifySoftwareToken(tokenRequest);
            System.out.println("The status of the token is " +
 verifyResponse.statusAsString());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }
```

- Para obter detalhes da API, consulte VerifySoftwareTokena Referência AWS SDK for Java 2.x da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
const verifySoftwareToken = (totp) => {
  const client = new CognitoIdentityProviderClient({});

  // The 'Session' is provided in the response to 'AssociateSoftwareToken'.
  const session = process.env.SESSION;

  if (!session) {
    throw new Error(
      "Missing a valid Session. Did you run 'admin-initiate-auth'?",
    );
```

```
  }

  const command = new VerifySoftwareTokenCommand({
    Session: session,
    UserCode: totp,
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte VerifySoftwareTokena Referência AWS SDK para JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
// Verify the TOTP and register for MFA.
suspend fun verifyTOTP(
    sessionVal: String?,
    codeVal: String?,
) {
    val tokenRequest =
        VerifySoftwareTokenRequest {
            userCode = codeVal
            session = sessionVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val verifyResponse =
 identityProviderClient.verifySoftwareToken(tokenRequest)
        println("The status of the token is ${verifyResponse.status}")
    }
```

```
}
```

- Para obter detalhes da API, consulte a VerifySoftwareTokenreferência da API AWS SDK for Kotlin.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def verify_mfa(self, session, user_code):
        """
        Verify a new MFA application that is associated with a user.

        :param session: Session information returned from a previous call to
 initiate
```

```
                authentication.
        :param user_code: A code generated by the associated MFA application.
        :return: Status that indicates whether the MFA application is verified.
        """
        try:
            response = self.cognito_idp_client.verify_software_token(
                Session=session, UserCode=user_code
            )
        except ClientError as err:
            logger.error(
                "Couldn't verify MFA. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response
```

- Para obter detalhes da API, consulte a VerifySoftwareTokenReferência da API AWS SDK for Python (Boto3).

SAP ABAP

SDK para SAP ABAP

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
    TRY.
        DATA(lo_result) = lo_cgp->verifysoftwaretoken(
          iv_session = iv_session
          iv_usercode = iv_user_code
        ).

        ov_status = lo_result->get_status( ).
```

```
        IF ov_status = 'SUCCESS'.
          MESSAGE 'MFA token verified successfully.' TYPE 'I'.
        ELSE.
          MESSAGE |MFA verification status: { ov_status }.| TYPE 'I'.
        ENDIF.

      CATCH /aws1/cx_cgpcodemismatchex INTO DATA(lo_code_ex).
        MESSAGE 'Invalid MFA code provided.' TYPE 'E'.

      CATCH /aws1/cx_cgpenbsoftwaretokmf00 INTO DATA(lo_enabled_ex).
        MESSAGE 'Software token MFA is already enabled.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte a VerifySoftwareTokenreferência da API AWS SDK for SAP ABAP.

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import AWSClientRuntime
import AWSCognitoIdentityProvider

    /// Confirm that the user's TOTP authenticator is configured correctly by
    /// sending a code to it to check that it matches successfully.
    ///
    /// - Parameters:
    ///   - cipClient: The `CongnitoIdentityProviderClient` to use.
    ///   - session: An authentication session previously returned by an
    ///     `associateSoftwareToken()` call.
    ///   - mfaCode: The 6-digit code currently displayed by the user's
    ///     authenticator, as provided by the user.
```

```
    func verifyTOTP(cipClient: CognitoIdentityProviderClient, session: String?,
mfaCode: String?) async {
        do {
            let output = try await cipClient.verifySoftwareToken(
                input: VerifySoftwareTokenInput(
                    session: session,
                    userCode: mfaCode
                )
            )

            guard let tokenStatus = output.status else {
                print("*** Unable to get the token's status.")
                return
            }
            print("=====> The token's status is: \(tokenStatus)")
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
            return
        } catch _ as CodeMismatchException {
            print("*** The specified MFA code doesn't match the expected value.")
            return
        } catch _ as UserNotFoundException {
            print("*** The specified username doesn't exist.")
            return
        } catch _ as UserNotConfirmedException {
            print("*** The user has not been confirmed.")
            return
        } catch {
            print("*** Error verifying the MFA token!")
            return
        }
    }
```

- Para obter detalhes da API, consulte VerifySoftwareTokena referência da API AWS SDK for Swift.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Cenários para o Amazon Cognito Identity Provider usando AWS SDKs

Os exemplos de código a seguir mostram como implementar cenários comuns no Amazon Cognito Identity Provider com. AWS SDKs Esses cenários mostram como realizar tarefas específicas chamando várias funções no Amazon Cognito Identity Provider ou em combinação com outros Serviços da AWS. Cada cenário inclui um link para o código-fonte completo, onde podem ser encontradas instruções sobre como configurar e executar o código.

Os cenários têm como alvo um nível intermediário de experiência para ajudar você a compreender ações de serviço em contexto.

Exemplos

- Confirme automaticamente usuários conhecidos do Amazon Cognito com uma função Lambda usando um SDK AWS
- Migre automaticamente usuários conhecidos do Amazon Cognito com uma função Lambda usando um SDK AWS
- Cadastrar um usuário com um grupo de usuários do Amazon Cognito que exija MFA usando um SDK AWS
- Usar bancos de identidades e fluxos de identidades do Amazon Cognito
- Grave dados de atividades personalizados com uma função Lambda após a autenticação do usuário do Amazon Cognito usando um SDK AWS

Confirme automaticamente usuários conhecidos do Amazon Cognito com uma função Lambda usando um SDK AWS

Os exemplos de código a seguir mostram como confirmar automaticamente usuários conhecidos do Amazon Cognito com uma função do Lambda.

- Configure um grupo de usuários para chamar uma função do Lambda para o acionador `PreSignUp`.
- Inscreva-se para ser um usuário no Amazon Cognito.
- A função do Lambda verifica uma tabela do DynamoDB e confirma automaticamente os usuários conhecidos.
- Faça login como o novo usuário e, em seguida, limpe os recursos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Execute um cenário interativo em um prompt de comando.

```
import (
 "context"
 "errors"
 "log"
 "strings"
 "user_pools_and_lambda_triggers/actions"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// AutoConfirm separates the steps of this scenario into individual functions so
 that
// they are simpler to read and understand.
type AutoConfirm struct {
 helper       IScenarioHelper
 questioner   demotools.IQuestioner
 resources    Resources
 cognitoActor *actions.CognitoActions
}

// NewAutoConfirm constructs a new auto confirm runner.
func NewAutoConfirm(sdkConfig aws.Config, questioner demotools.IQuestioner,
 helper IScenarioHelper) AutoConfirm {
 scenario := AutoConfirm{
  helper:       helper,
  questioner:   questioner,
  resources:    Resources{},
```

```
  cognitoActor: &actions.CognitoActions{CognitoClient:
 cognitoidentityprovider.NewFromConfig(sdkConfig)},
 }
 scenario.resources.init(scenario.cognitoActor, questioner)
 return scenario
}

// AddPreSignUpTrigger adds a Lambda handler as an invocation target for the
 PreSignUp trigger.
func (runner *AutoConfirm) AddPreSignUpTrigger(ctx context.Context, userPoolId
 string, functionArn string) {
 log.Printf("Let's add a Lambda function to handle the PreSignUp trigger from
 Cognito.\n" +
  "This trigger happens when a user signs up, and lets your function take action
 before the main Cognito\n" +
  "sign up processing occurs.\n")
 err := runner.cognitoActor.UpdateTriggers(
  ctx, userPoolId,
  actions.TriggerInfo{Trigger: actions.PreSignUp, HandlerArn:
 aws.String(functionArn)})
 if err != nil {
  panic(err)
 }
 log.Printf("Lambda function %v added to user pool %v to handle the PreSignUp
 trigger.\n",
  functionArn, userPoolId)
}

// SignUpUser signs up a user from the known user table with a password you
 specify.
func (runner *AutoConfirm) SignUpUser(ctx context.Context, clientId string,
 usersTable string) (string, string) {
 log.Println("Let's sign up a user to your Cognito user pool. When the user's
 email matches an email in the\n" +
  "DynamoDB known users table, it is automatically verified and the user is
 confirmed.")

 knownUsers, err := runner.helper.GetKnownUsers(ctx, usersTable)
 if err != nil {
  panic(err)
 }
 userChoice := runner.questioner.AskChoice("Which user do you want to use?\n",
 knownUsers.UserNameList())
 user := knownUsers.Users[userChoice]
```

```
 var signedUp bool
 var userConfirmed bool
 password := runner.questioner.AskPassword("Enter a password that has at least
 eight characters, uppercase, lowercase, numbers and symbols.\n"+
  "(the password will not display as you type):", 8)
 for !signedUp {
  log.Printf("Signing up user '%v' with email '%v' to Cognito.\n", user.UserName,
 user.UserEmail)
  userConfirmed, err = runner.cognitoActor.SignUp(ctx, clientId, user.UserName,
 password, user.UserEmail)
  if err != nil {
   var invalidPassword *types.InvalidPasswordException
   if errors.As(err, &invalidPassword) {
    password = runner.questioner.AskPassword("Enter another password:", 8)
   } else {
    panic(err)
   }
  } else {
   signedUp = true
  }
 }
 log.Printf("User %v signed up, confirmed = %v.\n", user.UserName, userConfirmed)

 log.Println(strings.Repeat("-", 88))

 return user.UserName, password
}

// SignInUser signs in a user.
func (runner *AutoConfirm) SignInUser(ctx context.Context, clientId string,
 userName string, password string) string {
 runner.questioner.Ask("Press Enter when you're ready to continue.")
 log.Printf("Let's sign in as %v...\n", userName)
 authResult, err := runner.cognitoActor.SignIn(ctx, clientId, userName, password)
 if err != nil {
  panic(err)
 }
 log.Printf("Successfully signed in. Your access token starts with: %v...\n",
 (*authResult.AccessToken)[:10])
 log.Println(strings.Repeat("-", 88))
 return *authResult.AccessToken
}
```

```
// Run runs the scenario.
func (runner *AutoConfirm) Run(ctx context.Context, stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup(ctx)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(ctx, stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]
 runner.helper.PopulateUserTable(ctx, stackOutputs["TableName"])

 runner.AddPreSignUpTrigger(ctx, stackOutputs["UserPoolId"],
 stackOutputs["AutoConfirmFunctionArn"])
 runner.resources.triggers = append(runner.resources.triggers, actions.PreSignUp)
 userName, password := runner.SignUpUser(ctx, stackOutputs["UserPoolClientId"],
 stackOutputs["TableName"])
 runner.helper.ListRecentLogEvents(ctx, stackOutputs["AutoConfirmFunction"])
 runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
  runner.SignInUser(ctx, stackOutputs["UserPoolClientId"], userName, password))

 runner.resources.Cleanup(ctx)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Aborde o acionador `PreSignUp` com uma função do Lambda.

```
import (
```

```
 "context"
 "log"
 "os"

 "github.com/aws/aws-lambda-go/events"
 "github.com/aws/aws-lambda-go/lambda"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 dynamodbtypes "github.com/aws/aws-sdk-go-v2/service/dynamodb/types"
)

const TABLE_NAME = "TABLE_NAME"

// UserInfo defines structured user data that can be marshalled to a DynamoDB
 format.
type UserInfo struct {
 UserName  string `dynamodbav:"UserName"`
 UserEmail string `dynamodbav:"UserEmail"`
}

// GetKey marshals the user email value to a DynamoDB key format.
func (user UserInfo) GetKey() map[string]dynamodbtypes.AttributeValue {
 userEmail, err := attributevalue.Marshal(user.UserEmail)
 if err != nil {
  panic(err)
 }
 return map[string]dynamodbtypes.AttributeValue{"UserEmail": userEmail}
}

type handler struct {
 dynamoClient *dynamodb.Client
}

// HandleRequest handles the PreSignUp event by looking up a user in an Amazon
 DynamoDB table and
// specifying whether they should be confirmed and verified.
func (h *handler) HandleRequest(ctx context.Context, event
 events.CognitoEventUserPoolsPreSignup) (events.CognitoEventUserPoolsPreSignup,
 error) {
 log.Printf("Received presignup from %v for user '%v'", event.TriggerSource,
 event.UserName)
 if event.TriggerSource != "PreSignUp_SignUp" {
```

```
  // Other trigger sources, such as PreSignUp_AdminInitiateAuth, ignore the
 response from this handler.
  return event, nil
 }
 tableName := os.Getenv(TABLE_NAME)
 user := UserInfo{
  UserEmail: event.Request.UserAttributes["email"],
 }
 log.Printf("Looking up email %v in table %v.\n", user.UserEmail, tableName)
 output, err := h.dynamoClient.GetItem(ctx, &dynamodb.GetItemInput{
  Key:       user.GetKey(),
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Error looking up email %v.\n", user.UserEmail)
  return event, err
 }
 if output.Item == nil {
  log.Printf("Email %v not found. Email verification is required.\n",
 user.UserEmail)
  return event, err
 }

 err = attributevalue.UnmarshalMap(output.Item, &user)
 if err != nil {
  log.Printf("Couldn't unmarshal DynamoDB item. Here's why: %v\n", err)
  return event, err
 }

 if user.UserName != event.UserName {
  log.Printf("UserEmail %v found, but stored UserName '%v' does not match
 supplied UserName '%v'. Verification is required.\n",
   user.UserEmail, user.UserName, event.UserName)
 } else {
  log.Printf("UserEmail %v found with matching UserName %v. User is confirmed.
\n", user.UserEmail, user.UserName)
  event.Response.AutoConfirmUser = true
  event.Response.AutoVerifyEmail = true
 }

 return event, err
}

func main() {
```

```
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Crie uma struct que realize tarefas comuns.

```
import (
 "context"
 "log"
 "strings"
 "time"
 "user_pools_and_lambda_triggers/actions"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudformation"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
 GetStackOutputs(ctx context.Context, stackName string) (actions.StackOutputs,
 error)
 PopulateUserTable(ctx context.Context, tableName string)
 GetKnownUsers(ctx context.Context, tableName string) (actions.UserList, error)
 AddKnownUser(ctx context.Context, tableName string, user actions.User)
 ListRecentLogEvents(ctx context.Context, functionName string)
}
```

```
// ScenarioHelper contains AWS wrapper structs used by the workflows in this
 example.
type ScenarioHelper struct {
 questioner  demotools.IQuestioner
 dynamoActor *actions.DynamoActions
 cfnActor    *actions.CloudFormationActions
 cwlActor    *actions.CloudWatchLogsActions
 isTestRun   bool
}

// NewScenarioHelper constructs a new scenario helper.
func NewScenarioHelper(sdkConfig aws.Config, questioner demotools.IQuestioner)
 ScenarioHelper {
 scenario := ScenarioHelper{
  questioner:  questioner,
  dynamoActor: &actions.DynamoActions{DynamoClient:
 dynamodb.NewFromConfig(sdkConfig)},
  cfnActor:    &actions.CloudFormationActions{CfnClient:
 cloudformation.NewFromConfig(sdkConfig)},
  cwlActor:    &actions.CloudWatchLogsActions{CwlClient:
 cloudwatchlogs.NewFromConfig(sdkConfig)},
 }
 return scenario
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 if !helper.isTestRun {
  time.Sleep(time.Duration(secs) * time.Second)
 }
}

// GetStackOutputs gets the outputs from the specified CloudFormation stack in a
 structured format.
func (helper ScenarioHelper) GetStackOutputs(ctx context.Context, stackName
 string) (actions.StackOutputs, error) {
 return helper.cfnActor.GetOutputs(ctx, stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(ctx context.Context, tableName
 string) {
 log.Printf("First, let's add some users to the DynamoDB %v table we'll use for
 this example.\n", tableName)
```

```
 err := helper.dynamoActor.PopulateTable(ctx, tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(ctx context.Context, tableName string)
 (actions.UserList, error) {
 knownUsers, err := helper.dynamoActor.Scan(ctx, tableName)
 if err != nil {
  log.Printf("Couldn't get known users from table %v. Here's why: %v\n",
 tableName, err)
 }
 return knownUsers, err
}

// AddKnownUser adds a user to the known users table.
func (helper ScenarioHelper) AddKnownUser(ctx context.Context, tableName string,
 user actions.User) {
 log.Printf("Adding user '%v' with email '%v' to the DynamoDB known users
 table...\n",
  user.UserName, user.UserEmail)
 err := helper.dynamoActor.AddUser(ctx, tableName, user)
 if err != nil {
  panic(err)
 }
}

// ListRecentLogEvents gets the most recent log stream and events for the
 specified Lambda function and displays them.
func (helper ScenarioHelper) ListRecentLogEvents(ctx context.Context,
 functionName string) {
 log.Println("Waiting a few seconds to let Lambda write to CloudWatch Logs...")
 helper.Pause(10)
 log.Println("Okay, let's check the logs to find what's happened recently with
 your Lambda function.")
 logStream, err := helper.cwlActor.GetLatestLogStream(ctx, functionName)
 if err != nil {
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
```

```
 events, err := helper.cwlActor.GetLogEvents(ctx, functionName,
 *logStream.LogStreamName, 10)
 if err != nil {
  panic(err)
 }
 for _, event := range events {
  log.Printf("\t%v", *event.Message)
 }
 log.Println(strings.Repeat("-", 88))
}
```

Crie uma struct que encapsule ações do Amazon Cognito.

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
 trigger.
type Trigger int

const (
 PreSignUp Trigger = iota
 UserMigration
 PostAuthentication
)

type TriggerInfo struct {
```

```
 Trigger    Trigger
 HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
 is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(ctx context.Context, userPoolId
 string, triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(ctx,
 &cognitoidentityprovider.DescribeUserPoolInput{
  UserPoolId: aws.String(userPoolId),
 })
 if err != nil {
  log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
 userPoolId, err)
  return err
 }
 lambdaConfig := output.UserPool.LambdaConfig
 for _, trigger := range triggers {
  switch trigger.Trigger {
  case PreSignUp:
   lambdaConfig.PreSignUp = trigger.HandlerArn
  case UserMigration:
   lambdaConfig.UserMigration = trigger.HandlerArn
  case PostAuthentication:
   lambdaConfig.PostAuthentication = trigger.HandlerArn
  }
 }
 _, err = actor.CognitoClient.UpdateUserPool(ctx,
 &cognitoidentityprovider.UpdateUserPoolInput{
  UserPoolId:   aws.String(userPoolId),
  LambdaConfig: lambdaConfig,
 })
 if err != nil {
  log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
 }
 return err
}



// SignUp signs up a user with Amazon Cognito.
```

```
func (actor CognitoActions) SignUp(ctx context.Context, clientId string, userName
 string, password string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(ctx,
 &cognitoidentityprovider.SignUpInput{
  ClientId: aws.String(clientId),
  Password: aws.String(password),
  Username: aws.String(userName),
  UserAttributes: []types.AttributeType{
   {Name: aws.String("email"), Value: aws.String(userEmail)},
  },
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
  }
 } else {
  confirmed = output.UserConfirmed
 }
 return confirmed, err
}



// SignIn signs in a user to Amazon Cognito using a username and password
 authentication flow.
func (actor CognitoActions) SignIn(ctx context.Context, clientId string, userName
 string, password string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(ctx,
 &cognitoidentityprovider.InitiateAuthInput{
  AuthFlow:       "USER_PASSWORD_AUTH",
  ClientId:       aws.String(clientId),
  AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
 })
 if err != nil {
  var resetRequired *types.PasswordResetRequiredException
  if errors.As(err, &resetRequired) {
   log.Println(*resetRequired.Message)
  } else {
   log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
```

```
  }
 } else {
  authResult = output.AuthenticationResult
 }
 return authResult, err
}



// ForgotPassword starts a password recovery flow for a user. This flow typically
 sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(ctx context.Context, clientId string,
 userName string) (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(ctx,
 &cognitoidentityprovider.ForgotPasswordInput{
  ClientId: aws.String(clientId),
  Username: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't start password reset for user '%v'. Here;s why: %v\n",
 userName, err)
 }
 return output.CodeDeliveryDetails, err
}



// ConfirmForgotPassword confirms a user with a confirmation code and a new
 password.
func (actor CognitoActions) ConfirmForgotPassword(ctx context.Context, clientId
 string, code string, userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(ctx,
 &cognitoidentityprovider.ConfirmForgotPasswordInput{
  ClientId:         aws.String(clientId),
  ConfirmationCode: aws.String(code),
  Password:         aws.String(password),
  Username:         aws.String(userName),
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
```

```
   log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
  }
 }
 return err
}



// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(ctx context.Context, userAccessToken
 string) error {
 _, err := actor.CognitoClient.DeleteUser(ctx,
 &cognitoidentityprovider.DeleteUserInput{
  AccessToken: aws.String(userAccessToken),
 })
 if err != nil {
  log.Printf("Couldn't delete user. Here's why: %v\n", err)
 }
 return err
}



// AdminCreateUser uses administrator credentials to add a user to a user pool.
 This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(ctx context.Context, userPoolId
 string, userName string, userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(ctx,
 &cognitoidentityprovider.AdminCreateUserInput{
  UserPoolId:     aws.String(userPoolId),
  Username:       aws.String(userName),
  MessageAction:  types.MessageActionTypeSuppress,
  UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
 aws.String(userEmail)}},
 })
 if err != nil {
  var userExists *types.UsernameExistsException
  if errors.As(err, &userExists) {
   log.Printf("User %v already exists in the user pool.", userName)
   err = nil
  } else {
   log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
  }
```

```
 }
 return err
}



// AdminSetUserPassword uses administrator credentials to set a password for a
 user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(ctx context.Context, userPoolId
 string, userName string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(ctx,
 &cognitoidentityprovider.AdminSetUserPasswordInput{
  Password:   aws.String(password),
  UserPoolId: aws.String(userPoolId),
  Username:   aws.String(userName),
  Permanent:  true,
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
 err)
  }
 }
 return err
}
```

Crie uma struct que encapsule ações do DynamoDB.

```
import (
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
```

```
 "github.com/aws/aws-sdk-go-v2/service/dynamodb/types"
)

// DynamoActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
 actions
// used in the examples.
type DynamoActions struct {
 DynamoClient *dynamodb.Client
}

// User defines structured user data.
type User struct {
 UserName  string
 UserEmail string
 LastLogin *LoginInfo `dynamodbav:",omitempty"`
}

// LoginInfo defines structured custom login data.
type LoginInfo struct {
 UserPoolId string
 ClientId   string
 Time       string
}

// UserList defines a list of users.
type UserList struct {
 Users []User
}

// UserNameList returns the usernames contained in a UserList as a list of
 strings.
func (users *UserList) UserNameList() []string {
 names := make([]string, len(users.Users))
 for i := 0; i < len(users.Users); i++ {
  names[i] = users.Users[i].UserName
 }
 return names
}

// PopulateTable adds a set of test users to the table.
func (actor DynamoActions) PopulateTable(ctx context.Context, tableName string)
 error {
 var err error
 var item map[string]types.AttributeValue
```

```
 var writeReqs []types.WriteRequest
 for i := 1; i < 4; i++ {
  item, err = attributevalue.MarshalMap(User{UserName: fmt.Sprintf("test_user_
%v", i), UserEmail: fmt.Sprintf("test_email_%v@example.com", i)})
  if err != nil {
   log.Printf("Couldn't marshall user into DynamoDB format. Here's why: %v\n",
 err)
   return err
  }
  writeReqs = append(writeReqs, types.WriteRequest{PutRequest:
 &types.PutRequest{Item: item}})
 }
 _, err = actor.DynamoClient.BatchWriteItem(ctx, &dynamodb.BatchWriteItemInput{
  RequestItems: map[string][]types.WriteRequest{tableName: writeReqs},
 })
 if err != nil {
  log.Printf("Couldn't populate table %v with users. Here's why: %v\n",
 tableName, err)
 }
 return err
}

// Scan scans the table for all items.
func (actor DynamoActions) Scan(ctx context.Context, tableName string) (UserList,
 error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(ctx, &dynamodb.ScanInput{
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't scan table %v for items. Here's why: %v\n", tableName,
 err)
 } else {
  err = attributevalue.UnmarshalListOfMaps(output.Items, &userList.Users)
  if err != nil {
   log.Printf("Couldn't unmarshal items into users. Here's why: %v\n", err)
  }
 }
 return userList, err
}

// AddUser adds a user item to a table.
func (actor DynamoActions) AddUser(ctx context.Context, tableName string, user
 User) error {
```

```
 userItem, err := attributevalue.MarshalMap(user)
 if err != nil {
  log.Printf("Couldn't marshall user to item. Here's why: %v\n", err)
 }
 _, err = actor.DynamoClient.PutItem(ctx, &dynamodb.PutItemInput{
  Item:      userItem,
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't put item in table %v. Here's why: %v", tableName, err)
 }
 return err
}
```

Crie uma estrutura que envolva as ações do CloudWatch Logs.

```
import (
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs/types"
)

type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(ctx context.Context,
 functionName string) (types.LogStream, error) {
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(ctx,
 &cloudwatchlogs.DescribeLogStreamsInput{
  Descending:   aws.Bool(true),
  Limit:        aws.Int32(1),
  LogGroupName: aws.String(logGroupName),
```

```
  OrderBy:      types.OrderByLastEventTime,
 })
 if err != nil {
  log.Printf("Couldn't get log streams for log group %v. Here's why: %v\n",
 logGroupName, err)
 } else {
  logStream = output.LogStreams[0]
 }
 return logStream, err
}

// GetLogEvents gets the most recent eventCount events from the specified log
 stream.
func (actor CloudWatchLogsActions) GetLogEvents(ctx context.Context, functionName
 string, logStreamName string, eventCount int32) (
 []types.OutputLogEvent, error) {
 var events []types.OutputLogEvent
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.GetLogEvents(ctx,
 &cloudwatchlogs.GetLogEventsInput{
  LogStreamName: aws.String(logStreamName),
  Limit:         aws.Int32(eventCount),
  LogGroupName:  aws.String(logGroupName),
 })
 if err != nil {
  log.Printf("Couldn't get log event for log stream %v. Here's why: %v\n",
 logStreamName, err)
 } else {
  events = output.Events
 }
 return events, err
}
```

Crie uma estrutura que envolva as ações. CloudFormation

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
```

```
 "github.com/aws/aws-sdk-go-v2/service/cloudformation"
)

// StackOutputs defines a map of outputs from a specific stack.
type StackOutputs map[string]string

type CloudFormationActions struct {
 CfnClient *cloudformation.Client
}

// GetOutputs gets the outputs from a CloudFormation stack and puts them into a
 structured format.
func (actor CloudFormationActions) GetOutputs(ctx context.Context, stackName
 string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(ctx,
 &cloudformation.DescribeStacksInput{
  StackName: aws.String(stackName),
 })
 if err != nil || len(output.Stacks) == 0 {
  log.Panicf("Couldn't find a CloudFormation stack named %v. Here's why: %v\n",
 stackName, err)
 }
 stackOutputs := StackOutputs{}
 for _, out := range output.Stacks[0].Outputs {
  stackOutputs[*out.OutputKey] = *out.OutputValue
 }
 return stackOutputs
}
```

Limpe recursos.

```
import (
 "context"
 "log"
 "user_pools_and_lambda_triggers/actions"

 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// Resources keeps track of AWS resources created during an example and handles
```

```
// cleanup when the example finishes.
type Resources struct {
 userPoolId       string
 userAccessTokens []string
 triggers         []actions.Trigger

 cognitoActor *actions.CognitoActions
 questioner   demotools.IQuestioner
}

func (resources *Resources) init(cognitoActor *actions.CognitoActions, questioner
 demotools.IQuestioner) {
 resources.userAccessTokens = []string{}
 resources.triggers = []actions.Trigger{}
 resources.cognitoActor = cognitoActor
 resources.questioner = questioner
}

// Cleanup deletes all AWS resources created during an example.
func (resources *Resources) Cleanup(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong during cleanup.\n%v\n", r)
   log.Println("Use the AWS Management Console to remove any remaining resources
 \n" +
    "that were created for this scenario.")
  }
 }()

 wantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS
 resources that were created "+
  "during this demo (y/n)?", "y")
 if wantDelete {
  for _, accessToken := range resources.userAccessTokens {
   err := resources.cognitoActor.DeleteUser(ctx, accessToken)
   if err != nil {
    log.Println("Couldn't delete user during cleanup.")
    panic(err)
   }
   log.Println("Deleted user.")
  }
  triggerList := make([]actions.TriggerInfo, len(resources.triggers))
  for i := 0; i < len(resources.triggers); i++ {
```

```
    triggerList[i] = actions.TriggerInfo{Trigger: resources.triggers[i],
  HandlerArn: nil}
   }
   err := resources.cognitoActor.UpdateTriggers(ctx, resources.userPoolId,
  triggerList...)
   if err != nil {
    log.Println("Couldn't update Cognito triggers during cleanup.")
    panic(err)
   }
   log.Println("Removed Cognito triggers from user pool.")
  } else {
   log.Println("Be sure to remove resources when you're done with them to avoid
  unexpected charges!")
  }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Go .
  - DeleteUser
  - InitiateAuth
  - SignUp
  - UpdateUserPool

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Configure uma execução interativa de “Cenário”. Os exemplos JavaScript (v3) compartilham um executor de cenários para simplificar exemplos complexos. O código-fonte completo está ativado GitHub.

```
import { AutoConfirm } from "./scenario-auto-confirm.js";

/**
 * The context is passed to every scenario. Scenario steps
 * will modify the context.
 */
const context = {
  errors: [],
  users: [
    {
      UserName: "test_user_1",
      UserEmail: "test_email_1@example.com",
    },
    {
      UserName: "test_user_2",
      UserEmail: "test_email_2@example.com",
    },
    {
      UserName: "test_user_3",
      UserEmail: "test_email_3@example.com",
    },
  ],
};

/**
 * Three Scenarios are created for the workflow. A Scenario is an orchestration
 class
 * that simplifies running a series of steps.
 */
export const scenarios = {
  // Demonstrate automatically confirming known users in a database.
  "auto-confirm": AutoConfirm(context),
};

// Call function if run directly
import { fileURLToPath } from "node:url";
import { parseScenarioArgs } from "@aws-doc-sdk-examples/lib/scenario/index.js";

if (process.argv[1] === fileURLToPath(import.meta.url)) {
  parseScenarioArgs(scenarios, {
    name: "Cognito user pools and triggers",
    description:
```

```
      "Demonstrate how to use the AWS SDKs to customize Amazon Cognito
 authentication behavior.",
  });
}
```

Esse cenário demonstra a confirmação automática de um usuário conhecido. Ele orquestra as etapas do exemplo.

```
import { wait } from "@aws-doc-sdk-examples/lib/utils/util-timers.js";
import {
  Scenario,
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";

import {
  getStackOutputs,
  logCleanUpReminder,
  promptForStackName,
  promptForStackRegion,
  skipWhenErrors,
} from "./steps-common.js";
import { populateTable } from "./actions/dynamodb-actions.js";
import {
  addPreSignUpHandler,
  deleteUser,
  getUser,
  signIn,
  signUpUser,
} from "./actions/cognito-actions.js";
import {
  getLatestLogStreamForLambda,
  getLogEvents,
} from "./actions/cloudwatch-logs-actions.js";

/**
 * @typedef {{
 *   errors: Error[],
 *   password: string,
 *   users: { UserName: string, UserEmail: string }[],
 *   selectedUser?: string,
```

```
 *   stackName?: string,
 *   stackRegion?: string,
 *   token?: string,
 *   confirmDeleteSignedInUser?: boolean,
 *   TableName?: string,
 *   UserPoolClientId?: string,
 *   UserPoolId?: string,
 *   UserPoolArn?: string,
 *   AutoConfirmHandlerArn?: string,
 *   AutoConfirmHandlerName?: string
 * }} State
 */

const greeting = new ScenarioOutput(
  "greeting",
  (/** @type {State} */ state) => `This demo will populate some users into the \
database created as part of the "${state.stackName}" stack. \
Then the AutoConfirmHandler will be linked to the PreSignUp \
trigger from Cognito. Finally, you will choose a user to sign up.`,
  { skipWhen: skipWhenErrors },
);

const logPopulatingUsers = new ScenarioOutput(
  "logPopulatingUsers",
  "Populating the DynamoDB table with some users.",
  { skipWhenErrors: skipWhenErrors },
);

const logPopulatingUsersComplete = new ScenarioOutput(
  "logPopulatingUsersComplete",
  "Done populating users.",
  { skipWhen: skipWhenErrors },
);

const populateUsers = new ScenarioAction(
  "populateUsers",
  async (/** @type {State} */ state) => {
    const [_, err] = await populateTable({
      region: state.stackRegion,
      tableName: state.TableName,
      items: state.users,
    });
    if (err) {
      state.errors.push(err);
```

```
    }
  },
  {
    skipWhen: skipWhenErrors,
  },
);

const logSetupSignUpTrigger = new ScenarioOutput(
  "logSetupSignUpTrigger",
  "Setting up the PreSignUp trigger for the Cognito User Pool.",
  { skipWhen: skipWhenErrors },
);

const setupSignUpTrigger = new ScenarioAction(
  "setupSignUpTrigger",
  async (/** @type {State} */ state) => {
    const [_, err] = await addPreSignUpHandler({
      region: state.stackRegion,
      userPoolId: state.UserPoolId,
      handlerArn: state.AutoConfirmHandlerArn,
    });
    if (err) {
      state.errors.push(err);
    }
  },
  {
    skipWhen: skipWhenErrors,
  },
);

const logSetupSignUpTriggerComplete = new ScenarioOutput(
  "logSetupSignUpTriggerComplete",
  (
    /** @type {State} */ state,
  ) => `The lambda function "${state.AutoConfirmHandlerName}" \
has been configured as the PreSignUp trigger handler for the user pool
 "${state.UserPoolId}".`,
  { skipWhen: skipWhenErrors },
);

const selectUser = new ScenarioInput(
  "selectedUser",
  "Select a user to sign up.",
  {
```

```
    type: "select",
    choices: (/** @type {State} */ state) => state.users.map((u) => u.UserName),
    skipWhen: skipWhenErrors,
    default: (/** @type {State} */ state) => state.users[0].UserName,
  },
);

const checkIfUserAlreadyExists = new ScenarioAction(
  "checkIfUserAlreadyExists",
  async (/** @type {State} */ state) => {
    const [user, err] = await getUser({
      region: state.stackRegion,
      userPoolId: state.UserPoolId,
      username: state.selectedUser,
    });

    if (err?.name === "UserNotFoundException") {
      // Do nothing. We're not expecting the user to exist before
      // sign up is complete.
      return;
    }

    if (err) {
      state.errors.push(err);
      return;
    }

    if (user) {
      state.errors.push(
        new Error(
          `The user "${state.selectedUser}" already exists in the user pool
 "${state.UserPoolId}".`,
        ),
      );
    }
  },
  {
    skipWhen: skipWhenErrors,
  },
);

const createPassword = new ScenarioInput(
  "password",
```

```
  "Enter a password that has at least eight characters, uppercase, lowercase,
 numbers and symbols.",
  { type: "password", skipWhen: skipWhenErrors, default: "Abcd1234!" },
);

const logSignUpExistingUser = new ScenarioOutput(
  "logSignUpExistingUser",
  (/** @type {State} */ state) => `Signing up user "${state.selectedUser}".`,
  { skipWhen: skipWhenErrors },
);

const signUpExistingUser = new ScenarioAction(
  "signUpExistingUser",
  async (/** @type {State} */ state) => {
    const signUp = (password) =>
      signUpUser({
        region: state.stackRegion,
        userPoolClientId: state.UserPoolClientId,
        username: state.selectedUser,
        email: state.users.find((u) => u.UserName === state.selectedUser)
          .UserEmail,
        password,
      });

    let [_, err] = await signUp(state.password);

    while (err?.name === "InvalidPasswordException") {
      console.warn("The password you entered was invalid.");
      await createPassword.handle(state);
      [_, err] = await signUp(state.password);
    }

    if (err) {
      state.errors.push(err);
    }
  },
  { skipWhen: skipWhenErrors },
);

const logSignUpExistingUserComplete = new ScenarioOutput(
  "logSignUpExistingUserComplete",
  (/** @type {State} */ state) =>
    `"${state.selectedUser} was signed up successfully.`,
  { skipWhen: skipWhenErrors },
```

```
);

const logLambdaLogs = new ScenarioAction(
  "logLambdaLogs",
  async (/** @type {State} */ state) => {
    console.log(
      "Waiting a few seconds to let Lambda write to CloudWatch Logs...\n",
    );
    await wait(10);

    const [logStream, logStreamErr] = await getLatestLogStreamForLambda({
      functionName: state.AutoConfirmHandlerName,
      region: state.stackRegion,
    });
    if (logStreamErr) {
      state.errors.push(logStreamErr);
      return;
    }

    console.log(
      `Getting some recent events from log stream "${logStream.logStreamName}"`,
    );
    const [logEvents, logEventsErr] = await getLogEvents({
      functionName: state.AutoConfirmHandlerName,
      region: state.stackRegion,
      eventCount: 10,
      logStreamName: logStream.logStreamName,
    });
    if (logEventsErr) {
      state.errors.push(logEventsErr);
      return;
    }

    console.log(logEvents.map((ev) => `\t${ev.message}`).join(""));
  },
  { skipWhen: skipWhenErrors },
);

const logSignInUser = new ScenarioOutput(
  "logSignInUser",
  (/** @type {State} */ state) => `Let's sign in as ${state.selectedUser}`,
  { skipWhen: skipWhenErrors },
);
```

```
const signInUser = new ScenarioAction(
  "signInUser",
  async (/** @type {State} */ state) => {
    const [response, err] = await signIn({
      region: state.stackRegion,
      clientId: state.UserPoolClientId,
      username: state.selectedUser,
      password: state.password,
    });

    if (err?.name === "PasswordResetRequiredException") {
      state.errors.push(new Error("Please reset your password."));
      return;
    }

    if (err) {
      state.errors.push(err);
      return;
    }

    state.token = response?.AuthenticationResult?.AccessToken;
  },
  { skipWhen: skipWhenErrors },
);

const logSignInUserComplete = new ScenarioOutput(
  "logSignInUserComplete",
  (/** @type {State} */ state) =>
    `Successfully signed in. Your access token starts with:
 ${state.token.slice(0, 11)}`,
  { skipWhen: skipWhenErrors },
);

const confirmDeleteSignedInUser = new ScenarioInput(
  "confirmDeleteSignedInUser",
  "Do you want to delete the currently signed in user?",
  { type: "confirm", skipWhen: skipWhenErrors },
);

const deleteSignedInUser = new ScenarioAction(
  "deleteSignedInUser",
  async (/** @type {State} */ state) => {
    const [_, err] = await deleteUser({
      region: state.stackRegion,
```

```
      accessToken: state.token,
    });

    if (err) {
      state.errors.push(err);
    }
  },
  {
    skipWhen: (/** @type {State} */ state) =>
      skipWhenErrors(state) || !state.confirmDeleteSignedInUser,
  },
);

const logErrors = new ScenarioOutput(
  "logErrors",
  (/** @type {State}*/ state) => {
    const errorList = state.errors
      .map((err) => ` - ${err.name}: ${err.message}`)
      .join("\n");
    return `Scenario errors found:\n${errorList}`;
  },
  {
    // Don't log errors when there aren't any!
    skipWhen: (/** @type {State} */ state) => state.errors.length === 0,
  },
);

export const AutoConfirm = (context) =>
  new Scenario(
    "AutoConfirm",
    [
      promptForStackName,
      promptForStackRegion,
      getStackOutputs,
      greeting,
      logPopulatingUsers,
      populateUsers,
      logPopulatingUsersComplete,
      logSetupSignUpTrigger,
      setupSignUpTrigger,
      logSetupSignUpTriggerComplete,
      selectUser,
      checkIfUserAlreadyExists,
      createPassword,
```

```
      logSignUpExistingUser,
      signUpExistingUser,
      logSignUpExistingUserComplete,
      logLambdaLogs,
      logSignInUser,
      signInUser,
      logSignInUserComplete,
      confirmDeleteSignedInUser,
      deleteSignedInUser,
      logCleanUpReminder,
      logErrors,
    ],
    context,
  );
```

Essas são etapas compartilhadas com outros cenários.

```
import {
  ScenarioAction,
  ScenarioInput,
  ScenarioOutput,
} from "@aws-doc-sdk-examples/lib/scenario/scenario.js";
import { getCfnOutputs } from "@aws-doc-sdk-examples/lib/sdk/cfn-outputs.js";

export const skipWhenErrors = (state) => state.errors.length > 0;

export const getStackOutputs = new ScenarioAction(
  "getStackOutputs",
  async (state) => {
    if (!state.stackName || !state.stackRegion) {
      state.errors.push(
        new Error(
          "No stack name or region provided. The stack name and \
region are required to fetch CFN outputs relevant to this example.",
        ),
      );
      return;
    }

    const outputs = await getCfnOutputs(state.stackName, state.stackRegion);
    Object.assign(state, outputs);
  },
```

```
);

export const promptForStackName = new ScenarioInput(
  "stackName",
  "Enter the name of the stack you deployed earlier.",
  { type: "input", default: "PoolsAndTriggersStack" },
);

export const promptForStackRegion = new ScenarioInput(
  "stackRegion",
  "Enter the region of the stack you deployed earlier.",
  { type: "input", default: "us-east-1" },
);

export const logCleanUpReminder = new ScenarioOutput(
  "logCleanUpReminder",
  "All done. Remember to run 'cdk destroy' to teardown the stack.",
  { skipWhen: skipWhenErrors },
);
```

Um manipulador do gatilho `PreSignUp` com uma função do Lambda.

```
import type { PreSignUpTriggerEvent, Handler } from "aws-lambda";
import type { UserRepository } from "./user-repository";
import { DynamoDBUserRepository } from "./user-repository";

export class PreSignUpHandler {
  private userRepository: UserRepository;

  constructor(userRepository: UserRepository) {
    this.userRepository = userRepository;
  }

  private isPreSignUpTriggerSource(event: PreSignUpTriggerEvent): boolean {
    return event.triggerSource === "PreSignUp_SignUp";
  }

  private getEventUserEmail(event: PreSignUpTriggerEvent): string {
    return event.request.userAttributes.email;
  }

  async handlePreSignUpTriggerEvent(
```

```
    event: PreSignUpTriggerEvent,
  ): Promise<PreSignUpTriggerEvent> {
    console.log(
      `Received presignup from ${event.triggerSource} for user
 '${event.userName}'`,
    );

    if (!this.isPreSignUpTriggerSource(event)) {
      return event;
    }

    const eventEmail = this.getEventUserEmail(event);
    console.log(`Looking up email ${eventEmail}.`);
    const storedUserInfo =
      await this.userRepository.getUserInfoByEmail(eventEmail);

    if (!storedUserInfo) {
      console.log(
        `Email ${eventEmail} not found. Email verification is required.`,
      );
      return event;
    }

    if (storedUserInfo.UserName !== event.userName) {
      console.log(
        `UserEmail ${eventEmail} found, but stored UserName
 '${storedUserInfo.UserName}' does not match supplied UserName
 '${event.userName}'. Verification is required.`,
      );
    } else {
      console.log(
        `UserEmail ${eventEmail} found with matching UserName
 ${storedUserInfo.UserName}. User is confirmed.`,
      );
      event.response.autoConfirmUser = true;
      event.response.autoVerifyEmail = true;
    }
    return event;
  }
}

const createPreSignUpHandler = (): PreSignUpHandler => {
  const tableName = process.env.TABLE_NAME;
  if (!tableName) {
```

```
    throw new Error("TABLE_NAME environment variable is not set");
  }

  const userRepository = new DynamoDBUserRepository(tableName);
  return new PreSignUpHandler(userRepository);
};

export const handler: Handler = async (event: PreSignUpTriggerEvent) => {
  const preSignUpHandler = createPreSignUpHandler();
  return preSignUpHandler.handlePreSignUpTriggerEvent(event);
};
```

Módulo de ações de CloudWatch registros.

```
import {
  CloudWatchLogsClient,
  GetLogEventsCommand,
  OrderBy,
  paginateDescribeLogStreams,
} from "@aws-sdk/client-cloudwatch-logs";

/**
 * Get the latest log stream for a Lambda function.
 * @param {{ functionName: string, region: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cloudwatch-logs").LogStream | null,
 unknown]>}
 */
export const getLatestLogStreamForLambda = async ({ functionName, region }) => {
  try {
    const logGroupName = `/aws/lambda/${functionName}`;
    const cwlClient = new CloudWatchLogsClient({ region });
    const paginator = paginateDescribeLogStreams(
      { client: cwlClient },
      {
        descending: true,
        limit: 1,
        orderBy: OrderBy.LastEventTime,
        logGroupName,
      },
    );
```

```
    for await (const page of paginator) {
      return [page.logStreams[0], null];
    }
  } catch (err) {
    return [null, err];
  }
};

/**
 * Get the log events for a Lambda function's log stream.
 * @param {{
 *   functionName: string,
 *   logStreamName: string,
 *   eventCount: number,
 *   region: string
 * }} config
 * @returns {Promise<[import("@aws-sdk/client-cloudwatch-logs").OutputLogEvent[]
 | null, unknown]>}
 */
export const getLogEvents = async ({
  functionName,
  logStreamName,
  eventCount,
  region,
}) => {
  try {
    const cwlClient = new CloudWatchLogsClient({ region });
    const logGroupName = `/aws/lambda/${functionName}`;
    const response = await cwlClient.send(
      new GetLogEventsCommand({
        logStreamName: logStreamName,
        limit: eventCount,
        logGroupName: logGroupName,
      }),
    );

    return [response.events, null];
  } catch (err) {
    return [null, err];
  }
};
```

Módulo de ações do Amazon Cognito.

```
import {
  AdminGetUserCommand,
  CognitoIdentityProviderClient,
  DeleteUserCommand,
  InitiateAuthCommand,
  SignUpCommand,
  UpdateUserPoolCommand,
} from "@aws-sdk/client-cognito-identity-provider";

/**
 * Connect a Lambda function to the PreSignUp trigger for a Cognito user pool
 * @param {{ region: string, userPoolId: string, handlerArn: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").UpdateUserPoolCommandOutput | null, unknown]>}
 */
export const addPreSignUpHandler = async ({
  region,
  userPoolId,
  handlerArn,
}) => {
  try {
    const cognitoClient = new CognitoIdentityProviderClient({
      region,
    });

    const command = new UpdateUserPoolCommand({
      UserPoolId: userPoolId,
      LambdaConfig: {
        PreSignUp: handlerArn,
      },
    });

    const response = await cognitoClient.send(command);
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};

/**
 * Attempt to register a user to a user pool with a given username and password.
```

```
 * @param {{
 *   region: string,
 *   userPoolClientId: string,
 *   username: string,
 *   email: string,
 *   password: string
 * }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").SignUpCommandOutput | null, unknown]>}
 */
export const signUpUser = async ({
  region,
  userPoolClientId,
  username,
  email,
  password,
}) => {
  try {
    const cognitoClient = new CognitoIdentityProviderClient({
      region,
    });

    const response = await cognitoClient.send(
      new SignUpCommand({
        ClientId: userPoolClientId,
        Username: username,
        Password: password,
        UserAttributes: [{ Name: "email", Value: email }],
      }),
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};

/**
 * Sign in a user to Amazon Cognito using a username and password authentication
 flow.
 * @param {{ region: string, clientId: string, username: string, password:
 string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").InitiateAuthCommandOutput | null, unknown]>}
 */
```

```
export const signIn = async ({ region, clientId, username, password }) => {
  try {
    const cognitoClient = new CognitoIdentityProviderClient({ region });
    const response = await cognitoClient.send(
      new InitiateAuthCommand({
        AuthFlow: "USER_PASSWORD_AUTH",
        ClientId: clientId,
        AuthParameters: { USERNAME: username, PASSWORD: password },
      }),
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};

/**
 * Retrieve an existing user from a user pool.
 * @param {{ region: string, userPoolId: string, username: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").AdminGetUserCommandOutput | null, unknown]>}
 */
export const getUser = async ({ region, userPoolId, username }) => {
  try {
    const cognitoClient = new CognitoIdentityProviderClient({ region });
    const response = await cognitoClient.send(
      new AdminGetUserCommand({
        UserPoolId: userPoolId,
        Username: username,
      }),
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};

/**
 * Delete the signed-in user. Useful for allowing a user to delete their
 * own profile.
 * @param {{ region: string, accessToken: string }} config
 * @returns {Promise<[import("@aws-sdk/client-cognito-identity-
provider").DeleteUserCommandOutput | null, unknown]>}
 */
```

```
export const deleteUser = async ({ region, accessToken }) => {
  try {
    const client = new CognitoIdentityProviderClient({ region });
    const response = await client.send(
      new DeleteUserCommand({ AccessToken: accessToken }),
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};
```

Módulo de ações do DynamoDB.

```
import { DynamoDBClient } from "@aws-sdk/client-dynamodb";
import {
  BatchWriteCommand,
  DynamoDBDocumentClient,
} from "@aws-sdk/lib-dynamodb";

/**
 * Populate a DynamoDB table with provide items.
 * @param {{ region: string, tableName: string, items: Record<string,
 unknown>[] }} config
 * @returns {Promise<[import("@aws-sdk/lib-dynamodb").BatchWriteCommandOutput |
 null, unknown]>}
 */
export const populateTable = async ({ region, tableName, items }) => {
  try {
    const ddbClient = new DynamoDBClient({ region });
    const docClient = DynamoDBDocumentClient.from(ddbClient);
    const response = await docClient.send(
      new BatchWriteCommand({
        RequestItems: {
          [tableName]: items.map((item) => ({
            PutRequest: {
              Item: item,
            },
          })),
        },
      }),
```

```
    );
    return [response, null];
  } catch (err) {
    return [null, err];
  }
};
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para JavaScript .
  - DeleteUser
  - InitiateAuth
  - SignUp
  - UpdateUserPool

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Migre automaticamente usuários conhecidos do Amazon Cognito com uma função Lambda usando um SDK AWS

O exemplo de código a seguir mostra como migrar automaticamente usuários conhecidas do Amazon Cognito com uma função do Lambda.

- Configure um grupo de usuários para chamar uma função do Lambda para o acionador `MigrateUser`.
- Faça login no Amazon Cognito com um nome de usuário e e-mail que não estejam no grupo de usuários.
- A função do Lambda verifica uma tabela do DynamoDB e migra automaticamente os usuários conhecidos para o grupo de usuários.
- Realize um fluxo de senha esquecida para redefinir a senha para o usuário migrado.
- Faça login como o novo usuário e, em seguida, limpe os recursos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Execute um cenário interativo em um prompt de comando.

```
import (
 "context"
 "errors"
 "fmt"
 "log"
 "strings"
 "user_pools_and_lambda_triggers/actions"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// MigrateUser separates the steps of this scenario into individual functions so
 that
// they are simpler to read and understand.
type MigrateUser struct {
 helper       IScenarioHelper
 questioner   demotools.IQuestioner
 resources    Resources
 cognitoActor *actions.CognitoActions
}

// NewMigrateUser constructs a new migrate user runner.
func NewMigrateUser(sdkConfig aws.Config, questioner demotools.IQuestioner,
 helper IScenarioHelper) MigrateUser {
 scenario := MigrateUser{
  helper:       helper,
  questioner:   questioner,
```

```
  resources:    Resources{},
  cognitoActor: &actions.CognitoActions{CognitoClient:
 cognitoidentityprovider.NewFromConfig(sdkConfig)},
 }
 scenario.resources.init(scenario.cognitoActor, questioner)
 return scenario
}

// AddMigrateUserTrigger adds a Lambda handler as an invocation target for the
 MigrateUser trigger.
func (runner *MigrateUser) AddMigrateUserTrigger(ctx context.Context, userPoolId
 string, functionArn string) {
 log.Printf("Let's add a Lambda function to handle the MigrateUser trigger from
 Cognito.\n" +
  "This trigger happens when an unknown user signs in, and lets your function
 take action before Cognito\n" +
  "rejects the user.\n\n")
 err := runner.cognitoActor.UpdateTriggers(
  ctx, userPoolId,
  actions.TriggerInfo{Trigger: actions.UserMigration, HandlerArn:
 aws.String(functionArn)})
 if err != nil {
  panic(err)
 }
 log.Printf("Lambda function %v added to user pool %v to handle the MigrateUser
 trigger.\n",
  functionArn, userPoolId)

 log.Println(strings.Repeat("-", 88))
}

// SignInUser adds a new user to the known users table and signs that user in to
 Amazon Cognito.
func (runner *MigrateUser) SignInUser(ctx context.Context, usersTable string,
 clientId string) (bool, actions.User) {
 log.Println("Let's sign in a user to your Cognito user pool. When the username
 and email matches an entry in the\n" +
  "DynamoDB known users table, the email is automatically verified and the user
 is migrated to the Cognito user pool.")

 user := actions.User{}
 user.UserName = runner.questioner.Ask("\nEnter a username:")
 user.UserEmail = runner.questioner.Ask("\nEnter an email that you own. This
 email will be used to confirm user migration\n" +
```

```
  "during this example:")

 runner.helper.AddKnownUser(ctx, usersTable, user)

 var err error
 var resetRequired *types.PasswordResetRequiredException
 var authResult *types.AuthenticationResultType
 signedIn := false
 for !signedIn && resetRequired == nil {
  log.Printf("Signing in to Cognito as user '%v'. The expected result is a
 PasswordResetRequiredException.\n\n", user.UserName)
  authResult, err = runner.cognitoActor.SignIn(ctx, clientId, user.UserName, "_")
  if err != nil {
   if errors.As(err, &resetRequired) {
    log.Printf("\nUser '%v' is not in the Cognito user pool but was found in the
 DynamoDB known users table.\n"+
     "User migration is started and a password reset is required.",
 user.UserName)
   } else {
    panic(err)
   }
  } else {
   log.Printf("User '%v' successfully signed in. This is unexpected and probably
 means you have not\n"+
    "cleaned up a previous run of this scenario, so the user exist in the Cognito
 user pool.\n"+
    "You can continue this example and select to clean up resources, or manually
 remove\n"+
    "the user from your user pool and try again.", user.UserName)
   runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
 *authResult.AccessToken)
   signedIn = true
  }
 }

 log.Println(strings.Repeat("-", 88))
 return resetRequired != nil, user
}

// ResetPassword starts a password recovery flow.
func (runner *MigrateUser) ResetPassword(ctx context.Context, clientId string,
 user actions.User) {
 wantCode := runner.questioner.AskBool(fmt.Sprintf("In order to migrate the user
 to Cognito, you must be able to receive a confirmation\n"+
```

```
 "code by email at %v. Do you want to send a code (y/n)?", user.UserEmail), "y")
if !wantCode {
 log.Println("To complete this example and successfully migrate a user to
Cognito, you must enter an email\n" +
  "you own that can receive a confirmation code.")
 return
}
codeDelivery, err := runner.cognitoActor.ForgotPassword(ctx, clientId,
user.UserName)
if err != nil {
 panic(err)
}
log.Printf("\nA confirmation code has been sent to %v.",
*codeDelivery.Destination)
code := runner.questioner.Ask("Check your email and enter it here:")

confirmed := false
password := runner.questioner.AskPassword("\nEnter a password that has at least
eight characters, uppercase, lowercase, numbers and symbols.\n"+
 "(the password will not display as you type):", 8)
for !confirmed {
 log.Printf("\nConfirming password reset for user '%v'.\n", user.UserName)
 err = runner.cognitoActor.ConfirmForgotPassword(ctx, clientId, code,
user.UserName, password)
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   password = runner.questioner.AskPassword("\nEnter another password:", 8)
  } else {
   panic(err)
  }
 } else {
  confirmed = true
 }
}
log.Printf("User '%v' successfully confirmed and migrated.\n", user.UserName)
log.Println("Signing in with your username and password...")
authResult, err := runner.cognitoActor.SignIn(ctx, clientId, user.UserName,
password)
if err != nil {
 panic(err)
}
log.Printf("Successfully signed in. Your access token starts with: %v...\n",
(*authResult.AccessToken)[:10])
```

```
 runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
 *authResult.AccessToken)

 log.Println(strings.Repeat("-", 88))
}

// Run runs the scenario.
func (runner *MigrateUser) Run(ctx context.Context, stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup(ctx)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(ctx, stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]

 runner.AddMigrateUserTrigger(ctx, stackOutputs["UserPoolId"],
 stackOutputs["MigrateUserFunctionArn"])
 runner.resources.triggers = append(runner.resources.triggers,
 actions.UserMigration)
 resetNeeded, user := runner.SignInUser(ctx, stackOutputs["TableName"],
 stackOutputs["UserPoolClientId"])
 if resetNeeded {
  runner.helper.ListRecentLogEvents(ctx, stackOutputs["MigrateUserFunction"])
  runner.ResetPassword(ctx, stackOutputs["UserPoolClientId"], user)
 }

 runner.resources.Cleanup(ctx)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Aborde o acionador `MigrateUser` com uma função do Lambda.

```
import (
 "context"
 "log"
 "os"

 "github.com/aws/aws-lambda-go/events"
 "github.com/aws/aws-lambda-go/lambda"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/expression"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
)

const TABLE_NAME = "TABLE_NAME"

// UserInfo defines structured user data that can be marshalled to a DynamoDB
 format.
type UserInfo struct {
 UserName  string `dynamodbav:"UserName"`
 UserEmail string `dynamodbav:"UserEmail"`
}

type handler struct {
 dynamoClient *dynamodb.Client
}

// HandleRequest handles the MigrateUser event by looking up a user in an Amazon
 DynamoDB table and
// specifying whether they should be migrated to the user pool.
func (h *handler) HandleRequest(ctx context.Context, event
 events.CognitoEventUserPoolsMigrateUser)
 (events.CognitoEventUserPoolsMigrateUser, error) {
 log.Printf("Received migrate trigger from %v for user '%v'",
 event.TriggerSource, event.UserName)
 if event.TriggerSource != "UserMigration_Authentication" {
  return event, nil
 }
```

```
tableName := os.Getenv(TABLE_NAME)
user := UserInfo{
 UserName: event.UserName,
}
log.Printf("Looking up user '%v' in table %v.\n", user.UserName, tableName)
filterEx := expression.Name("UserName").Equal(expression.Value(user.UserName))
expr, err := expression.NewBuilder().WithFilter(filterEx).Build()
if err != nil {
 log.Printf("Error building expression to query for user '%v'.\n",
user.UserName)
 return event, err
}
output, err := h.dynamoClient.Scan(ctx, &dynamodb.ScanInput{
 TableName:                 aws.String(tableName),
 FilterExpression:          expr.Filter(),
 ExpressionAttributeNames:  expr.Names(),
 ExpressionAttributeValues: expr.Values(),
})
if err != nil {
 log.Printf("Error looking up user '%v'.\n", user.UserName)
 return event, err
}
if len(output.Items) == 0 {
 log.Printf("User '%v' not found, not migrating user.\n", user.UserName)
 return event, err
}

var users []UserInfo
err = attributevalue.UnmarshalListOfMaps(output.Items, &users)
if err != nil {
 log.Printf("Couldn't unmarshal DynamoDB items. Here's why: %v\n", err)
 return event, err
}

user = users[0]
log.Printf("UserName '%v' found with email %v. User is migrated and must reset
password.\n", user.UserName, user.UserEmail)
event.CognitoEventUserPoolsMigrateUserResponse.UserAttributes =
map[string]string{
 "email":          user.UserEmail,
 "email_verified": "true", // email_verified is required for the forgot password
flow.
}
```

```
 event.CognitoEventUserPoolsMigrateUserResponse.FinalUserStatus =
 "RESET_REQUIRED"
 event.CognitoEventUserPoolsMigrateUserResponse.MessageAction = "SUPPRESS"

 return event, err
}

func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Crie uma struct que realize tarefas comuns.

```
import (
 "context"
 "log"
 "strings"
 "time"
 "user_pools_and_lambda_triggers/actions"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudformation"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
```

```
 GetStackOutputs(ctx context.Context, stackName string) (actions.StackOutputs,
 error)
 PopulateUserTable(ctx context.Context, tableName string)
 GetKnownUsers(ctx context.Context, tableName string) (actions.UserList, error)
 AddKnownUser(ctx context.Context, tableName string, user actions.User)
 ListRecentLogEvents(ctx context.Context, functionName string)
}

// ScenarioHelper contains AWS wrapper structs used by the workflows in this
 example.
type ScenarioHelper struct {
 questioner  demotools.IQuestioner
 dynamoActor *actions.DynamoActions
 cfnActor    *actions.CloudFormationActions
 cwlActor    *actions.CloudWatchLogsActions
 isTestRun   bool
}

// NewScenarioHelper constructs a new scenario helper.
func NewScenarioHelper(sdkConfig aws.Config, questioner demotools.IQuestioner)
 ScenarioHelper {
 scenario := ScenarioHelper{
  questioner:  questioner,
  dynamoActor: &actions.DynamoActions{DynamoClient:
 dynamodb.NewFromConfig(sdkConfig)},
  cfnActor:    &actions.CloudFormationActions{CfnClient:
 cloudformation.NewFromConfig(sdkConfig)},
  cwlActor:    &actions.CloudWatchLogsActions{CwlClient:
 cloudwatchlogs.NewFromConfig(sdkConfig)},
 }
 return scenario
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 if !helper.isTestRun {
  time.Sleep(time.Duration(secs) * time.Second)
 }
}

// GetStackOutputs gets the outputs from the specified CloudFormation stack in a
 structured format.
func (helper ScenarioHelper) GetStackOutputs(ctx context.Context, stackName
 string) (actions.StackOutputs, error) {
```

```
 return helper.cfnActor.GetOutputs(ctx, stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(ctx context.Context, tableName
 string) {
 log.Printf("First, let's add some users to the DynamoDB %v table we'll use for
 this example.\n", tableName)
 err := helper.dynamoActor.PopulateTable(ctx, tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(ctx context.Context, tableName string)
 (actions.UserList, error) {
 knownUsers, err := helper.dynamoActor.Scan(ctx, tableName)
 if err != nil {
  log.Printf("Couldn't get known users from table %v. Here's why: %v\n",
 tableName, err)
 }
 return knownUsers, err
}

// AddKnownUser adds a user to the known users table.
func (helper ScenarioHelper) AddKnownUser(ctx context.Context, tableName string,
 user actions.User) {
 log.Printf("Adding user '%v' with email '%v' to the DynamoDB known users
 table...\n",
  user.UserName, user.UserEmail)
 err := helper.dynamoActor.AddUser(ctx, tableName, user)
 if err != nil {
  panic(err)
 }
}

// ListRecentLogEvents gets the most recent log stream and events for the
 specified Lambda function and displays them.
func (helper ScenarioHelper) ListRecentLogEvents(ctx context.Context,
 functionName string) {
 log.Println("Waiting a few seconds to let Lambda write to CloudWatch Logs...")
 helper.Pause(10)
```

```
 log.Println("Okay, let's check the logs to find what's happened recently with
 your Lambda function.")
 logStream, err := helper.cwlActor.GetLatestLogStream(ctx, functionName)
 if err != nil {
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
 events, err := helper.cwlActor.GetLogEvents(ctx, functionName,
 *logStream.LogStreamName, 10)
 if err != nil {
  panic(err)
 }
 for _, event := range events {
  log.Printf("\t%v", *event.Message)
 }
 log.Println(strings.Repeat("-", 88))
}
```

Crie uma struct que encapsule ações do Amazon Cognito.

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
 trigger.
type Trigger int
```

```
const (
 PreSignUp Trigger = iota
 UserMigration
 PostAuthentication
)

type TriggerInfo struct {
 Trigger    Trigger
 HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
 is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(ctx context.Context, userPoolId
 string, triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(ctx,
 &cognitoidentityprovider.DescribeUserPoolInput{
  UserPoolId: aws.String(userPoolId),
 })
 if err != nil {
  log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
 userPoolId, err)
  return err
 }
 lambdaConfig := output.UserPool.LambdaConfig
 for _, trigger := range triggers {
  switch trigger.Trigger {
  case PreSignUp:
   lambdaConfig.PreSignUp = trigger.HandlerArn
  case UserMigration:
   lambdaConfig.UserMigration = trigger.HandlerArn
  case PostAuthentication:
   lambdaConfig.PostAuthentication = trigger.HandlerArn
  }
 }
 _, err = actor.CognitoClient.UpdateUserPool(ctx,
 &cognitoidentityprovider.UpdateUserPoolInput{
  UserPoolId:   aws.String(userPoolId),
  LambdaConfig: lambdaConfig,
 })
 if err != nil {
  log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
```

```
 }
 return err
}



// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(ctx context.Context, clientId string, userName
 string, password string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(ctx,
 &cognitoidentityprovider.SignUpInput{
  ClientId: aws.String(clientId),
  Password: aws.String(password),
  Username: aws.String(userName),
  UserAttributes: []types.AttributeType{
   {Name: aws.String("email"), Value: aws.String(userEmail)},
  },
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
  }
 } else {
  confirmed = output.UserConfirmed
 }
 return confirmed, err
}



// SignIn signs in a user to Amazon Cognito using a username and password
 authentication flow.
func (actor CognitoActions) SignIn(ctx context.Context, clientId string, userName
 string, password string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(ctx,
 &cognitoidentityprovider.InitiateAuthInput{
  AuthFlow:       "USER_PASSWORD_AUTH",
  ClientId:       aws.String(clientId),
  AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
```

```
 })
 if err != nil {
  var resetRequired *types.PasswordResetRequiredException
  if errors.As(err, &resetRequired) {
   log.Println(*resetRequired.Message)
  } else {
   log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
  }
 } else {
  authResult = output.AuthenticationResult
 }
 return authResult, err
}



// ForgotPassword starts a password recovery flow for a user. This flow typically
 sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(ctx context.Context, clientId string,
 userName string) (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(ctx,
 &cognitoidentityprovider.ForgotPasswordInput{
  ClientId: aws.String(clientId),
  Username: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't start password reset for user '%v'. Here;s why: %v\n",
 userName, err)
 }
 return output.CodeDeliveryDetails, err
}



// ConfirmForgotPassword confirms a user with a confirmation code and a new
 password.
func (actor CognitoActions) ConfirmForgotPassword(ctx context.Context, clientId
 string, code string, userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(ctx,
 &cognitoidentityprovider.ConfirmForgotPasswordInput{
  ClientId:         aws.String(clientId),
  ConfirmationCode: aws.String(code),
  Password:         aws.String(password),
```

```
  Username:         aws.String(userName),
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
  }
 }
 return err
}



// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(ctx context.Context, userAccessToken
 string) error {
 _, err := actor.CognitoClient.DeleteUser(ctx,
 &cognitoidentityprovider.DeleteUserInput{
  AccessToken: aws.String(userAccessToken),
 })
 if err != nil {
  log.Printf("Couldn't delete user. Here's why: %v\n", err)
 }
 return err
}



// AdminCreateUser uses administrator credentials to add a user to a user pool.
 This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(ctx context.Context, userPoolId
 string, userName string, userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(ctx,
 &cognitoidentityprovider.AdminCreateUserInput{
  UserPoolId:     aws.String(userPoolId),
  Username:       aws.String(userName),
  MessageAction:  types.MessageActionTypeSuppress,
  UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
 aws.String(userEmail)}},
 })
 if err != nil {
```

```
  var userExists *types.UsernameExistsException
  if errors.As(err, &userExists) {
   log.Printf("User %v already exists in the user pool.", userName)
   err = nil
  } else {
   log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
  }
 }
 return err
}



// AdminSetUserPassword uses administrator credentials to set a password for a
 user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(ctx context.Context, userPoolId
 string, userName string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(ctx,
 &cognitoidentityprovider.AdminSetUserPasswordInput{
  Password:   aws.String(password),
  UserPoolId: aws.String(userPoolId),
  Username:   aws.String(userName),
  Permanent:  true,
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
 err)
  }
 }
 return err
}
```

Crie uma struct que encapsule ações do DynamoDB.

```
import (
```

```
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb/types"
)

// DynamoActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
 actions
// used in the examples.
type DynamoActions struct {
 DynamoClient *dynamodb.Client
}

// User defines structured user data.
type User struct {
 UserName  string
 UserEmail string
 LastLogin *LoginInfo `dynamodbav:",omitempty"`
}

// LoginInfo defines structured custom login data.
type LoginInfo struct {
 UserPoolId string
 ClientId   string
 Time       string
}

// UserList defines a list of users.
type UserList struct {
 Users []User
}

// UserNameList returns the usernames contained in a UserList as a list of
 strings.
func (users *UserList) UserNameList() []string {
 names := make([]string, len(users.Users))
 for i := 0; i < len(users.Users); i++ {
  names[i] = users.Users[i].UserName
 }
 return names
```

```
}

// PopulateTable adds a set of test users to the table.
func (actor DynamoActions) PopulateTable(ctx context.Context, tableName string)
 error {
 var err error
 var item map[string]types.AttributeValue
 var writeReqs []types.WriteRequest
 for i := 1; i < 4; i++ {
  item, err = attributevalue.MarshalMap(User{UserName: fmt.Sprintf("test_user_
%v", i), UserEmail: fmt.Sprintf("test_email_%v@example.com", i)})
  if err != nil {
   log.Printf("Couldn't marshall user into DynamoDB format. Here's why: %v\n",
 err)
   return err
  }
  writeReqs = append(writeReqs, types.WriteRequest{PutRequest:
 &types.PutRequest{Item: item}})
 }
 _, err = actor.DynamoClient.BatchWriteItem(ctx, &dynamodb.BatchWriteItemInput{
  RequestItems: map[string][]types.WriteRequest{tableName: writeReqs},
 })
 if err != nil {
  log.Printf("Couldn't populate table %v with users. Here's why: %v\n",
 tableName, err)
 }
 return err
}

// Scan scans the table for all items.
func (actor DynamoActions) Scan(ctx context.Context, tableName string) (UserList,
 error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(ctx, &dynamodb.ScanInput{
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't scan table %v for items. Here's why: %v\n", tableName,
 err)
 } else {
  err = attributevalue.UnmarshalListOfMaps(output.Items, &userList.Users)
  if err != nil {
   log.Printf("Couldn't unmarshal items into users. Here's why: %v\n", err)
  }
```

```
 }
 return userList, err
}

// AddUser adds a user item to a table.
func (actor DynamoActions) AddUser(ctx context.Context, tableName string, user
 User) error {
 userItem, err := attributevalue.MarshalMap(user)
 if err != nil {
  log.Printf("Couldn't marshall user to item. Here's why: %v\n", err)
 }
 _, err = actor.DynamoClient.PutItem(ctx, &dynamodb.PutItemInput{
  Item:      userItem,
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't put item in table %v. Here's why: %v", tableName, err)
 }
 return err
}
```

Crie uma estrutura que envolva as ações do CloudWatch Logs.

```
import (
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs/types"
)

type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(ctx context.Context,
 functionName string) (types.LogStream, error) {
```

```
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(ctx,
 &cloudwatchlogs.DescribeLogStreamsInput{
  Descending:   aws.Bool(true),
  Limit:        aws.Int32(1),
  LogGroupName: aws.String(logGroupName),
  OrderBy:      types.OrderByLastEventTime,
 })
 if err != nil {
  log.Printf("Couldn't get log streams for log group %v. Here's why: %v\n",
 logGroupName, err)
 } else {
  logStream = output.LogStreams[0]
 }
 return logStream, err
}

// GetLogEvents gets the most recent eventCount events from the specified log
 stream.
func (actor CloudWatchLogsActions) GetLogEvents(ctx context.Context, functionName
 string, logStreamName string, eventCount int32) (
 []types.OutputLogEvent, error) {
 var events []types.OutputLogEvent
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.GetLogEvents(ctx,
 &cloudwatchlogs.GetLogEventsInput{
  LogStreamName: aws.String(logStreamName),
  Limit:         aws.Int32(eventCount),
  LogGroupName:  aws.String(logGroupName),
 })
 if err != nil {
  log.Printf("Couldn't get log event for log stream %v. Here's why: %v\n",
 logStreamName, err)
 } else {
  events = output.Events
 }
 return events, err
}
```

Crie uma estrutura que envolva as ações. CloudFormation

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudformation"
)

// StackOutputs defines a map of outputs from a specific stack.
type StackOutputs map[string]string

type CloudFormationActions struct {
 CfnClient *cloudformation.Client
}

// GetOutputs gets the outputs from a CloudFormation stack and puts them into a
 structured format.
func (actor CloudFormationActions) GetOutputs(ctx context.Context, stackName
 string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(ctx,
 &cloudformation.DescribeStacksInput{
  StackName: aws.String(stackName),
 })
 if err != nil || len(output.Stacks) == 0 {
  log.Panicf("Couldn't find a CloudFormation stack named %v. Here's why: %v\n",
 stackName, err)
 }
 stackOutputs := StackOutputs{}
 for _, out := range output.Stacks[0].Outputs {
  stackOutputs[*out.OutputKey] = *out.OutputValue
 }
 return stackOutputs
}
```

Limpe recursos.

```
import (
 "context"
 "log"
```

```
 "user_pools_and_lambda_triggers/actions"

 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// Resources keeps track of AWS resources created during an example and handles
// cleanup when the example finishes.
type Resources struct {
 userPoolId       string
 userAccessTokens []string
 triggers         []actions.Trigger

 cognitoActor *actions.CognitoActions
 questioner   demotools.IQuestioner
}

func (resources *Resources) init(cognitoActor *actions.CognitoActions, questioner
 demotools.IQuestioner) {
 resources.userAccessTokens = []string{}
 resources.triggers = []actions.Trigger{}
 resources.cognitoActor = cognitoActor
 resources.questioner = questioner
}

// Cleanup deletes all AWS resources created during an example.
func (resources *Resources) Cleanup(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong during cleanup.\n%v\n", r)
   log.Println("Use the AWS Management Console to remove any remaining resources
 \n" +
    "that were created for this scenario.")
  }
 }()

 wantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS
 resources that were created "+
  "during this demo (y/n)?", "y")
 if wantDelete {
  for _, accessToken := range resources.userAccessTokens {
   err := resources.cognitoActor.DeleteUser(ctx, accessToken)
   if err != nil {
    log.Println("Couldn't delete user during cleanup.")
    panic(err)
```

```
    }
    log.Println("Deleted user.")
   }
   triggerList := make([]actions.TriggerInfo, len(resources.triggers))
   for i := 0; i < len(resources.triggers); i++ {
    triggerList[i] = actions.TriggerInfo{Trigger: resources.triggers[i],
  HandlerArn: nil}
   }
   err := resources.cognitoActor.UpdateTriggers(ctx, resources.userPoolId,
  triggerList...)
   if err != nil {
    log.Println("Couldn't update Cognito triggers during cleanup.")
    panic(err)
   }
   log.Println("Removed Cognito triggers from user pool.")
  } else {
   log.Println("Be sure to remove resources when you're done with them to avoid
  unexpected charges!")
  }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Go .
  - ConfirmForgotPassword
  - DeleteUser
  - ForgotPassword
  - InitiateAuth
  - SignUp
  - UpdateUserPool

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Cadastrar um usuário com um grupo de usuários do Amazon Cognito que exija MFA usando um SDK AWS

Os exemplos de código a seguir mostram como:

- Inscrever e confirmar um usuário com nome de usuário, senha e endereço de e-mail.
- Configurar a autenticação multifator associando uma aplicação de MFA ao usuário.
- Faça login usando uma senha e um código de MFA.

.NET

SDK para .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
namespace CognitoBasics;

public class CognitoBasics
{
    private static ILogger logger = null!;

    static async Task Main(string[] args)
    {
        // Set up dependency injection for Amazon Cognito.
        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<IAmazonCognitoIdentityProvider>()
            .AddTransient<CognitoWrapper>()
            )
            .Build();
```

```
        logger = LoggerFactory.Create(builder => { builder.AddConsole(); })
            .CreateLogger<CognitoBasics>();

        var configuration = new ConfigurationBuilder()
            .SetBasePath(Directory.GetCurrentDirectory())
            .AddJsonFile("settings.json") // Load settings from .json file.
            .AddJsonFile("settings.local.json",
                true) // Optionally load local settings.
            .Build();

        var cognitoWrapper = host.Services.GetRequiredService<CognitoWrapper>();

        Console.WriteLine(new string('-', 80));
        UiMethods.DisplayOverview();
        Console.WriteLine(new string('-', 80));

        // clientId - The app client Id value that you get from the AWS CDK
script.
        var clientId = configuration["ClientId"]; // "*** REPLACE WITH CLIENT ID
VALUE FROM CDK SCRIPT";

        // poolId - The pool Id that you get from the AWS CDK script.
        var poolId = configuration["PoolId"]!; // "*** REPLACE WITH POOL ID VALUE
FROM CDK SCRIPT";
        var userName = configuration["UserName"];
        var password = configuration["Password"];
        var email = configuration["Email"];

        // If the username wasn't set in the configuration file,
        // get it from the user now.
        if (userName is null)
        {
            do
            {
                Console.Write("Username: ");
                userName = Console.ReadLine();
            }
            while (string.IsNullOrEmpty(userName));
        }
        Console.WriteLine($"\nUsername: {userName}");

        // If the password wasn't set in the configuration file,
        // get it from the user now.
```

```
        if (password is null)
        {
            do
            {
                Console.Write("Password: ");
                password = Console.ReadLine();
            }
            while (string.IsNullOrEmpty(password));
        }

        // If the email address wasn't set in the configuration file,
        // get it from the user now.
        if (email is null)
        {
            do
            {
                Console.Write("Email: ");
                email = Console.ReadLine();
            } while (string.IsNullOrEmpty(email));
        }

        // Now sign up the user.
        Console.WriteLine($"\nSigning up {userName} with email address:
{email}");
        await cognitoWrapper.SignUpAsync(clientId, userName, password, email);

        // Add the user to the user pool.
        Console.WriteLine($"Adding {userName} to the user pool");
        await cognitoWrapper.GetAdminUserAsync(userName, poolId);

        UiMethods.DisplayTitle("Get confirmation code");
        Console.WriteLine($"Conformation code sent to {userName}.");
        Console.Write("Would you like to send a new code? (Y/N) ");
        var answer = Console.ReadLine();

        if (answer!.ToLower() == "y")
        {
            await cognitoWrapper.ResendConfirmationCodeAsync(clientId, userName);
            Console.WriteLine("Sending a new confirmation code");
        }

        Console.Write("Enter confirmation code (from Email): ");
        var code = Console.ReadLine();
```

```
        await cognitoWrapper.ConfirmSignupAsync(clientId, code, userName);

        UiMethods.DisplayTitle("Checking status");
        Console.WriteLine($"Rechecking the status of {userName} in the user
 pool");
        await cognitoWrapper.GetAdminUserAsync(userName, poolId);

        Console.WriteLine($"Setting up authenticator for {userName} in the user
 pool");
        var setupResponse = await cognitoWrapper.InitiateAuthAsync(clientId,
 userName, password);

        var setupSession = await
 cognitoWrapper.AssociateSoftwareTokenAsync(setupResponse.Session);
        Console.Write("Enter the 6-digit code displayed in Google Authenticator:
 ");
        var setupCode = Console.ReadLine();

        var setupResult = await
 cognitoWrapper.VerifySoftwareTokenAsync(setupSession, setupCode);
        Console.WriteLine($"Setup status: {setupResult}");

        Console.WriteLine($"Now logging in {userName} in the user pool");
        var authSession = await cognitoWrapper.AdminInitiateAuthAsync(clientId,
 poolId, userName, password);

        Console.Write("Enter a new 6-digit code displayed in Google
 Authenticator: ");
        var authCode = Console.ReadLine();

        var authResult = await
 cognitoWrapper.AdminRespondToAuthChallengeAsync(userName, clientId, authCode,
 authSession, poolId);
        Console.WriteLine($"Authenticated and received access token:
 {authResult.AccessToken}");

        Console.WriteLine(new string('-', 80));
        Console.WriteLine("Cognito scenario is complete.");
        Console.WriteLine(new string('-', 80));
    }
}


using System.Net;
```

```
namespace CognitoActions;

/// <summary>
/// Methods to perform Amazon Cognito Identity Provider actions.
/// </summary>
public class CognitoWrapper
{
    private readonly IAmazonCognitoIdentityProvider _cognitoService;

    /// <summary>
    /// Constructor for the wrapper class containing Amazon Cognito actions.
    /// </summary>
    /// <param name="cognitoService">The Amazon Cognito client object.</param>
    public CognitoWrapper(IAmazonCognitoIdentityProvider cognitoService)
    {
        _cognitoService = cognitoService;
    }

    /// <summary>
    /// List the Amazon Cognito user pools for an account.
    /// </summary>
    /// <returns>A list of UserPoolDescriptionType objects.</returns>
    public async Task<List<UserPoolDescriptionType>> ListUserPoolsAsync()
    {
        var userPools = new List<UserPoolDescriptionType>();

        var userPoolsPaginator = _cognitoService.Paginators.ListUserPools(new
 ListUserPoolsRequest());

        await foreach (var response in userPoolsPaginator.Responses)
        {
            userPools.AddRange(response.UserPools);
        }

        return userPools;
    }


    /// <summary>
    /// Get a list of users for the Amazon Cognito user pool.
    /// </summary>
    /// <param name="userPoolId">The user pool ID.</param>
    /// <returns>A list of users.</returns>
```

```
    public async Task<List<UserType>> ListUsersAsync(string userPoolId)
    {
        var request = new ListUsersRequest
        {
            UserPoolId = userPoolId
        };

        var users = new List<UserType>();

        var usersPaginator = _cognitoService.Paginators.ListUsers(request);
        await foreach (var response in usersPaginator.Responses)
        {
            users.AddRange(response.Users);
        }

        return users;
    }


    /// <summary>
    /// Respond to an admin authentication challenge.
    /// </summary>
    /// <param name="userName">The name of the user.</param>
    /// <param name="clientId">The client ID.</param>
    /// <param name="mfaCode">The multi-factor authentication code.</param>
    /// <param name="session">The current application session.</param>
    /// <param name="clientId">The user pool ID.</param>
    /// <returns>The result of the authentication response.</returns>
    public async Task<AuthenticationResultType> AdminRespondToAuthChallengeAsync(
        string userName,
        string clientId,
        string mfaCode,
        string session,
        string userPoolId)
    {
        Console.WriteLine("SOFTWARE_TOKEN_MFA challenge is generated");

        var challengeResponses = new Dictionary<string, string>();
        challengeResponses.Add("USERNAME", userName);
        challengeResponses.Add("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

        var respondToAuthChallengeRequest = new
AdminRespondToAuthChallengeRequest
        {
```

```
            ChallengeName = ChallengeNameType.SOFTWARE_TOKEN_MFA,
            ClientId = clientId,
            ChallengeResponses = challengeResponses,
            Session = session,
            UserPoolId = userPoolId,
        };

        var response = await
_cognitoService.AdminRespondToAuthChallengeAsync(respondToAuthChallengeRequest);
        Console.WriteLine($"Response to Authentication
{response.AuthenticationResult.TokenType}");
        return response.AuthenticationResult;
    }


    /// <summary>
    /// Verify the TOTP and register for MFA.
    /// </summary>
    /// <param name="session">The name of the session.</param>
    /// <param name="code">The MFA code.</param>
    /// <returns>The status of the software token.</returns>
    public async Task<VerifySoftwareTokenResponseType>
VerifySoftwareTokenAsync(string session, string code)
    {
        var tokenRequest = new VerifySoftwareTokenRequest
        {
            UserCode = code,
            Session = session,
        };

        var verifyResponse = await
_cognitoService.VerifySoftwareTokenAsync(tokenRequest);

        return verifyResponse.Status;
    }


    /// <summary>
    /// Get an MFA token to authenticate the user with the authenticator.
    /// </summary>
    /// <param name="session">The session name.</param>
    /// <returns>The session name.</returns>
    public async Task<string> AssociateSoftwareTokenAsync(string session)
    {
```

```
        var softwareTokenRequest = new AssociateSoftwareTokenRequest
        {
            Session = session,
        };

        var tokenResponse = await
_cognitoService.AssociateSoftwareTokenAsync(softwareTokenRequest);
        var secretCode = tokenResponse.SecretCode;

        Console.WriteLine($"Use the following secret code to set up the
authenticator: {secretCode}");

        return tokenResponse.Session;
    }


    /// <summary>
    /// Initiate an admin auth request.
    /// </summary>
    /// <param name="clientId">The client ID to use.</param>
    /// <param name="userPoolId">The ID of the user pool.</param>
    /// <param name="userName">The username to authenticate.</param>
    /// <param name="password">The user's password.</param>
    /// <returns>The session to use in challenge-response.</returns>
    public async Task<string> AdminInitiateAuthAsync(string clientId, string
userPoolId, string userName, string password)
    {
        var authParameters = new Dictionary<string, string>();
        authParameters.Add("USERNAME", userName);
        authParameters.Add("PASSWORD", password);

        var request = new AdminInitiateAuthRequest
        {
            ClientId = clientId,
            UserPoolId = userPoolId,
            AuthParameters = authParameters,
            AuthFlow = AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
        };

        var response = await _cognitoService.AdminInitiateAuthAsync(request);
        return response.Session;
    }

    /// <summary>
```

```
    /// Initiate authorization.
    /// </summary>
    /// <param name="clientId">The client Id of the application.</param>
    /// <param name="userName">The name of the user who is authenticating.</
param>
    /// <param name="password">The password for the user who is authenticating.</
param>
    /// <returns>The response from the initiate auth request.</returns>
    public async Task<InitiateAuthResponse> InitiateAuthAsync(string clientId,
 string userName, string password)
    {
        var authParameters = new Dictionary<string, string>();
        authParameters.Add("USERNAME", userName);
        authParameters.Add("PASSWORD", password);

        var authRequest = new InitiateAuthRequest

        {
            ClientId = clientId,
            AuthParameters = authParameters,
            AuthFlow = AuthFlowType.USER_PASSWORD_AUTH,
        };

        var response = await _cognitoService.InitiateAuthAsync(authRequest);
        Console.WriteLine($"Result Challenge is : {response.ChallengeName}");

        return response;
    }

    /// <summary>
    /// Confirm that the user has signed up.
    /// </summary>
    /// <param name="clientId">The Id of this application.</param>
    /// <param name="code">The confirmation code sent to the user.</param>
    /// <param name="userName">The username.</param>
    /// <returns>True if successful.</returns>
    public async Task<bool> ConfirmSignupAsync(string clientId, string code,
 string userName)
    {
        var signUpRequest = new ConfirmSignUpRequest
        {
            ClientId = clientId,
            ConfirmationCode = code,
            Username = userName,
```

```
        };

        var response = await _cognitoService.ConfirmSignUpAsync(signUpRequest);
        if (response.HttpStatusCode == HttpStatusCode.OK)
        {
            Console.WriteLine($"{userName} was confirmed");
            return true;
        }
        return false;
    }


    /// <summary>
    /// Initiates and confirms tracking of the device.
    /// </summary>
    /// <param name="accessToken">The user's access token.</param>
    /// <param name="deviceKey">The key of the device from Amazon Cognito.</
param>
    /// <param name="deviceName">The device name.</param>
    /// <returns></returns>
    public async Task<bool> ConfirmDeviceAsync(string accessToken, string
 deviceKey, string deviceName)
    {
        var request = new ConfirmDeviceRequest
        {
            AccessToken = accessToken,
            DeviceKey = deviceKey,
            DeviceName = deviceName
        };

        var response = await _cognitoService.ConfirmDeviceAsync(request);
        return response.UserConfirmationNecessary;
    }


    /// <summary>
    /// Send a new confirmation code to a user.
    /// </summary>
    /// <param name="clientId">The Id of the client application.</param>
    /// <param name="userName">The username of user who will receive the code.</
param>
    /// <returns>The delivery details.</returns>
    public async Task<CodeDeliveryDetailsType> ResendConfirmationCodeAsync(string
 clientId, string userName)
```

```
    {
        var codeRequest = new ResendConfirmationCodeRequest
        {
            ClientId = clientId,
            Username = userName,
        };

        var response = await
_cognitoService.ResendConfirmationCodeAsync(codeRequest);

        Console.WriteLine($"Method of delivery is
{response.CodeDeliveryDetails.DeliveryMedium}");

        return response.CodeDeliveryDetails;
    }


    /// <summary>
    /// Get the specified user from an Amazon Cognito user pool with
administrator access.
    /// </summary>
    /// <param name="userName">The name of the user.</param>
    /// <param name="poolId">The Id of the Amazon Cognito user pool.</param>
    /// <returns>Async task.</returns>
    public async Task<UserStatusType> GetAdminUserAsync(string userName, string
poolId)
    {
        AdminGetUserRequest userRequest = new AdminGetUserRequest
        {
            Username = userName,
            UserPoolId = poolId,
        };

        var response = await _cognitoService.AdminGetUserAsync(userRequest);

        Console.WriteLine($"User status {response.UserStatus}");
        return response.UserStatus;
    }


    /// <summary>
    /// Sign up a new user.
    /// </summary>
    /// <param name="clientId">The client Id of the application.</param>
```

```
    /// <param name="userName">The username to use.</param>
    /// <param name="password">The user's password.</param>
    /// <param name="email">The email address of the user.</param>
    /// <returns>A Boolean value indicating whether the user was confirmed.</
returns>
    public async Task<bool> SignUpAsync(string clientId, string userName, string
 password, string email)
    {
        var userAttrs = new AttributeType
        {
            Name = "email",
            Value = email,
        };

        var userAttrsList = new List<AttributeType>();

        userAttrsList.Add(userAttrs);

        var signUpRequest = new SignUpRequest
        {
            UserAttributes = userAttrsList,
            Username = userName,
            ClientId = clientId,
            Password = password
        };

        var response = await _cognitoService.SignUpAsync(signUpRequest);
        return response.HttpStatusCode == HttpStatusCode.OK;
    }

}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para .NET .
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice

- ConfirmSignUp
- InitiateAuth
- ListUsers
- ResendConfirmationCode
- RespondToAuthChallenge
- SignUp
- VerifySoftwareToken

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
        Aws::Client::ClientConfiguration clientConfig;
        // Optional: Set to the AWS Region (overrides config file).
        // clientConfig.region = "us-east-1";

//! Scenario that adds a user to an Amazon Cognito user pool.
/*!
  \sa gettingStartedWithUserPools()
  \param clientID: Client ID associated with an Amazon Cognito user pool.
  \param userPoolID: An Amazon Cognito user pool ID.
  \param clientConfig: Aws client configuration.
  \return bool: Successful completion.
 */
bool AwsDoc::Cognito::gettingStartedWithUserPools(const Aws::String &clientID,
                                                  const Aws::String &userPoolID,
                                                  const
 Aws::Client::ClientConfiguration &clientConfig) {
    printAsterisksLine();
    std::cout
            << "Welcome to the Amazon Cognito example scenario."
            << std::endl;
    printAsterisksLine();
```

```
   std::cout
           << "This scenario will add a user to an Amazon Cognito user pool."
           << std::endl;
   const Aws::String userName = askQuestion("Enter a new username: ");
   const Aws::String password = askQuestion("Enter a new password: ");
   const Aws::String email = askQuestion("Enter a valid email for the user: ");

   std::cout << "Signing up " << userName << std::endl;

   Aws::CognitoIdentityProvider::CognitoIdentityProviderClient
client(clientConfig);
   bool userExists = false;
   do {
       // 1. Add a user with a username, password, and email address.
       Aws::CognitoIdentityProvider::Model::SignUpRequest request;
       request.AddUserAttributes(
               Aws::CognitoIdentityProvider::Model::AttributeType().WithName(
                       "email").WithValue(email));
       request.SetUsername(userName);
       request.SetPassword(password);
       request.SetClientId(clientID);
       Aws::CognitoIdentityProvider::Model::SignUpOutcome outcome =
               client.SignUp(request);

       if (outcome.IsSuccess()) {
           std::cout << "The signup request for " << userName << " was
successful."
                     << std::endl;
       }
       else if (outcome.GetError().GetErrorType() ==

Aws::CognitoIdentityProvider::CognitoIdentityProviderErrors::USERNAME_EXISTS) {
           std::cout
                   << "The username already exists. Please enter a different
username."
                   << std::endl;
           userExists = true;
       }
       else {
           std::cerr << "Error with CognitoIdentityProvider::SignUpRequest. "
                     << outcome.GetError().GetMessage()
                     << std::endl;
           return false;
```

```
        }
    } while (userExists);

    printAsterisksLine();
    std::cout << "Retrieving status of " << userName << " in the user pool."
              << std::endl;
    // 2. Confirm that the user was added to the user pool.
    if (!checkAdminUserStatus(userName, userPoolID, client)) {
        return false;
    }

    std::cout << "A confirmation code was sent to " << email << "." << std::endl;

    bool resend = askYesNoQuestion("Would you like to send a new code? (y/n) ");
    if (resend) {
        // Request a resend of the confirmation code to the email address.
(ResendConfirmationCode)
        Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeRequest
request;
        request.SetUsername(userName);
        request.SetClientId(clientID);

        Aws::CognitoIdentityProvider::Model::ResendConfirmationCodeOutcome
outcome =
                client.ResendConfirmationCode(request);

        if (outcome.IsSuccess()) {
            std::cout
                    << "CognitoIdentityProvider::ResendConfirmationCode was
successful."
                    << std::endl;
        }
        else {
            std::cerr << "Error with
CognitoIdentityProvider::ResendConfirmationCode. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }

    printAsterisksLine();

    {
```

```
        // 4. Send the confirmation code that's received in the email.
(ConfirmSignUp)
        const Aws::String confirmationCode = askQuestion(
                "Enter the confirmation code that was emailed: ");
        Aws::CognitoIdentityProvider::Model::ConfirmSignUpRequest request;
        request.SetClientId(clientID);
        request.SetConfirmationCode(confirmationCode);
        request.SetUsername(userName);

        Aws::CognitoIdentityProvider::Model::ConfirmSignUpOutcome outcome =
                client.ConfirmSignUp(request);

        if (outcome.IsSuccess()) {
            std::cout << "ConfirmSignup was Successful."
                      << std::endl;
        }
        else {
            std::cerr << "Error with CognitoIdentityProvider::ConfirmSignUp. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }

    std::cout << "Rechecking the status of " << userName << " in the user pool."
              << std::endl;
    if (!checkAdminUserStatus(userName, userPoolID, client)) {
        return false;
    }

    printAsterisksLine();

    std::cout << "Initiating authorization using the username and password."
              << std::endl;

    Aws::String session;
    // 5. Initiate authorization with username and password. (AdminInitiateAuth)
    if (!adminInitiateAuthorization(clientID, userPoolID,  userName, password,
session, client)) {
        return false;
    }

    printAsterisksLine();
```

```
    std::cout
            << "Starting setup of time-based one-time password (TOTP) multi-
factor authentication (MFA)."
            << std::endl;

    {
        // 6. Request a setup key for one-time password (TOTP)
        //    multi-factor authentication (MFA). (AssociateSoftwareToken)
        Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenRequest
 request;
        request.SetSession(session);

        Aws::CognitoIdentityProvider::Model::AssociateSoftwareTokenOutcome
 outcome =
                client.AssociateSoftwareToken(request);

        if (outcome.IsSuccess()) {
            std::cout
                    << "Enter this setup key into an authenticator app, for
 example Google Authenticator."
                    << std::endl;
            std::cout << "Setup key: " << outcome.GetResult().GetSecretCode()
                      << std::endl;
#ifdef USING_QR
            printAsterisksLine();
            std::cout << "\nOr scan the QR code in the file '" << QR_CODE_PATH <<
 "."
                      << std::endl;

            saveQRCode(std::string("otpauth://totp/") + userName + "?secret=" +
                       outcome.GetResult().GetSecretCode());
#endif // USING_QR
            session = outcome.GetResult().GetSession();
        }
        else {
            std::cerr << "Error with
 CognitoIdentityProvider::AssociateSoftwareToken. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }
    askQuestion("Type enter to continue...", alwaysTrueTest);
```

```
    printAsterisksLine();

    {
        Aws::String userCode = askQuestion(
                "Enter the 6 digit code displayed in the authenticator app: ");

        //  7. Send the MFA code copied from an authenticator app.
(VerifySoftwareToken)
        Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenRequest request;
        request.SetUserCode(userCode);
        request.SetSession(session);

        Aws::CognitoIdentityProvider::Model::VerifySoftwareTokenOutcome outcome =
                client.VerifySoftwareToken(request);

        if (outcome.IsSuccess()) {
            std::cout << "Verification of the code was successful."
                      << std::endl;
            session = outcome.GetResult().GetSession();
        }
        else {
            std::cerr << "Error with
CognitoIdentityProvider::VerifySoftwareToken. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
            return false;
        }
    }

    printAsterisksLine();
    std::cout << "You have completed the MFA authentication setup." << std::endl;
    std::cout << "Now, sign in." << std::endl;

    // 8. Initiate authorization again with username and password.
(AdminInitiateAuth)
    if (!adminInitiateAuthorization(clientID, userPoolID, userName, password,
session, client)) {
        return false;
    }

    Aws::String accessToken;
    {
        Aws::String mfaCode = askQuestion(
```

```
                "Re-enter the 6 digit code displayed in the authenticator app:
");

       // 9. Send a new MFA code copied from an authenticator app.
(AdminRespondToAuthChallenge)
       Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeRequest
request;
       request.AddChallengeResponses("USERNAME", userName);
       request.AddChallengeResponses("SOFTWARE_TOKEN_MFA_CODE", mfaCode);
       request.SetChallengeName(

Aws::CognitoIdentityProvider::Model::ChallengeNameType::SOFTWARE_TOKEN_MFA);
       request.SetClientId(clientID);
       request.SetUserPoolId(userPoolID);
       request.SetSession(session);

       Aws::CognitoIdentityProvider::Model::AdminRespondToAuthChallengeOutcome
outcome =
               client.AdminRespondToAuthChallenge(request);

       if (outcome.IsSuccess()) {
           std::cout << "Here is the response to the challenge.\n" <<

outcome.GetResult().GetAuthenticationResult().Jsonize().View().WriteReadable()
                     << std::endl;

           accessToken =
outcome.GetResult().GetAuthenticationResult().GetAccessToken();
       }
       else {
           std::cerr << "Error with
CognitoIdentityProvider::AdminRespondToAuthChallenge. "
                     << outcome.GetError().GetMessage()
                     << std::endl;
           return false;
       }

       std::cout << "You have successfully added a user to Amazon Cognito."
                 << std::endl;
   }

   if (askYesNoQuestion("Would you like to delete the user that you just added?
(y/n) ")) {
       // 10. Delete the user that you just added. (DeleteUser)
```

```
        Aws::CognitoIdentityProvider::Model::DeleteUserRequest request;
        request.SetAccessToken(accessToken);

        Aws::CognitoIdentityProvider::Model::DeleteUserOutcome outcome =
                client.DeleteUser(request);

        if (outcome.IsSuccess()) {
            std::cout << "The user " << userName << " was deleted."
                      << std::endl;
        }
        else {
            std::cerr << "Error with CognitoIdentityProvider::DeleteUser. "
                      << outcome.GetError().GetMessage()
                      << std::endl;
        }
    }

    return true;
}

//! Routine which checks the user status in an Amazon Cognito user pool.
/*!
 \sa checkAdminUserStatus()
 \param userName: A username.
 \param userPoolID: An Amazon Cognito user pool ID.
 \return bool: Successful completion.
 */
bool AwsDoc::Cognito::checkAdminUserStatus(const Aws::String &userName,
                                           const Aws::String &userPoolID,
                                           const
 Aws::CognitoIdentityProvider::CognitoIdentityProviderClient &client) {
    Aws::CognitoIdentityProvider::Model::AdminGetUserRequest request;
    request.SetUsername(userName);
    request.SetUserPoolId(userPoolID);

    Aws::CognitoIdentityProvider::Model::AdminGetUserOutcome outcome =
            client.AdminGetUser(request);

    if (outcome.IsSuccess()) {
        std::cout << "The status for " << userName << " is " <<

 Aws::CognitoIdentityProvider::Model::UserStatusTypeMapper::GetNameForUserStatusType(
                         outcome.GetResult().GetUserStatus()) << std::endl;
```

```
        std::cout << "Enabled is " << outcome.GetResult().GetEnabled() <<
 std::endl;
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::AdminGetUser. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}

//! Routine which starts authorization of an Amazon Cognito user.
//! This routine requires administrator credentials.
/*!
 \sa adminInitiateAuthorization()
 \param clientID: Client ID of tracked device.
 \param userPoolID: An Amazon Cognito user pool ID.
 \param userName: A username.
 \param password: A password.
 \param sessionResult: String to receive a session token.
 \return bool: Successful completion.
 */
bool AwsDoc::Cognito::adminInitiateAuthorization(const Aws::String &clientID,
                                                 const Aws::String &userPoolID,
                                                 const Aws::String &userName,
                                                 const Aws::String &password,
                                                 Aws::String &sessionResult,
                                                 const
 Aws::CognitoIdentityProvider::CognitoIdentityProviderClient &client) {
    Aws::CognitoIdentityProvider::Model::AdminInitiateAuthRequest request;
    request.SetClientId(clientID);
    request.SetUserPoolId(userPoolID);
    request.AddAuthParameters("USERNAME", userName);
    request.AddAuthParameters("PASSWORD", password);
    request.SetAuthFlow(

 Aws::CognitoIdentityProvider::Model::AuthFlowType::ADMIN_USER_PASSWORD_AUTH);


    Aws::CognitoIdentityProvider::Model::AdminInitiateAuthOutcome outcome =
            client.AdminInitiateAuth(request);

    if (outcome.IsSuccess()) {
```

```
        std::cout << "Call to AdminInitiateAuth was successful." << std::endl;
        sessionResult = outcome.GetResult().GetSession();
    }
    else {
        std::cerr << "Error with CognitoIdentityProvider::AdminInitiateAuth. "
                  << outcome.GetError().GetMessage()
                  << std::endl;
    }

    return outcome.IsSuccess();
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para C++ .
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice
  - ConfirmSignUp
  - InitiateAuth
  - ListUsers
  - ResendConfirmationCode
  - RespondToAuthChallenge
  - SignUp
  - VerifySoftwareToken

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.cognitoidentityprovider.CognitoIdentityProviderClient;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminGetUserRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminGetUserResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminInitiateAuthRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminInitiateAuthResponse;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminRespondToAuthChalleng
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AdminRespondToAuthChalleng
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AssociateSoftwareTokenRequ
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AssociateSoftwareTokenResp
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AttributeType;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.AuthFlowType;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ChallengeNameType;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.CognitoIdentityProviderExc
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ConfirmSignUpRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ResendConfirmationCodeRequ
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.ResendConfirmationCodeResp
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.SignUpRequest;
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.VerifySoftwareTokenRequest
import
 software.amazon.awssdk.services.cognitoidentityprovider.model.VerifySoftwareTokenRespons
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.util.ArrayList;
import java.util.HashMap;
```

```
import java.util.List;
import java.util.Map;
import java.util.Scanner;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 *
 * TIP: To set up the required user pool, run the AWS Cloud Development Kit (AWS
 * CDK) script provided in this GitHub repo at
 * resources/cdk/cognito_scenario_user_pool_with_mfa.
 *
 * This code example performs the following operations:
 *
 * 1. Invokes the signUp method to sign up a user.
 * 2. Invokes the adminGetUser method to get the user's confirmation status.
 * 3. Invokes the ResendConfirmationCode method if the user requested another
 * code.
 * 4. Invokes the confirmSignUp method.
 * 5. Invokes the AdminInitiateAuth to sign in. This results in being prompted
 * to set up TOTP (time-based one-time password). (The response is
 * “ChallengeName”: “MFA_SETUP”).
 * 6. Invokes the AssociateSoftwareToken method to generate a TOTP MFA private
 * key. This can be used with Google Authenticator.
 * 7. Invokes the VerifySoftwareToken method to verify the TOTP and register for
 * MFA.
 * 8. Invokes the AdminInitiateAuth to sign in again. This results in being
 * prompted to submit a TOTP (Response: “ChallengeName”: “SOFTWARE_TOKEN_MFA”).
 * 9. Invokes the AdminRespondToAuthChallenge to get back a token.
 */

public class CognitoMVP {
    public static final String DASHES = new String(new char[80]).replace("\0",
 "-");

    public static void main(String[] args) throws NoSuchAlgorithmException,
 InvalidKeyException {
        final String usage = """
```

```
                Usage:
                    <clientId> <poolId>

                Where:
                    clientId - The app client Id value that you can get from the
AWS CDK script.
                    poolId - The pool Id that you can get from the AWS CDK
script.\s
                """;

        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        String clientId = args[0];
        String poolId = args[1];
        CognitoIdentityProviderClient identityProviderClient =
CognitoIdentityProviderClient.builder()
                .region(Region.US_EAST_1)
                .build();

        System.out.println(DASHES);
        System.out.println("Welcome to the Amazon Cognito example scenario.");
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Enter your user name");
        Scanner in = new Scanner(System.in);
        String userName = in.nextLine();

        System.out.println("*** Enter your password");
        String password = in.nextLine();

        System.out.println("*** Enter your email");
        String email = in.nextLine();

        System.out.println("1. Signing up " + userName);
        signUp(identityProviderClient, clientId, userName, password, email);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("2. Getting " + userName + " in the user pool");
        getAdminUser(identityProviderClient, userName, poolId);
```

```
        System.out
                .println("*** Conformation code sent to " + userName + ". Would
you like to send a new code? (Yes/No)");
        System.out.println(DASHES);

        System.out.println(DASHES);
        String ans = in.nextLine();

        if (ans.compareTo("Yes") == 0) {
            resendConfirmationCode(identityProviderClient, clientId, userName);
            System.out.println("3. Sending a new confirmation code");
        }
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("4. Enter confirmation code that was emailed");
        String code = in.nextLine();
        confirmSignUp(identityProviderClient, clientId, code, userName);
        System.out.println("Rechecking the status of " + userName + " in the user
pool");
        getAdminUser(identityProviderClient, userName, poolId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("5. Invokes the initiateAuth to sign in");
        AdminInitiateAuthResponse authResponse =
initiateAuth(identityProviderClient, clientId, userName, password,
                poolId);
        String mySession = authResponse.session();
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("6. Invokes the AssociateSoftwareToken method to
generate a TOTP key");
        String newSession = getSecretForAppMFA(identityProviderClient,
mySession);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("*** Enter the 6-digit code displayed in Google
Authenticator");
        String myCode = in.nextLine();
        System.out.println(DASHES);
```

```
        System.out.println(DASHES);
        System.out.println("7. Verify the TOTP and register for MFA");
        verifyTOTP(identityProviderClient, newSession, myCode);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("8. Re-enter a 6-digit code displayed in Google
Authenticator");
        String mfaCode = in.nextLine();
        AdminInitiateAuthResponse authResponse1 =
initiateAuth(identityProviderClient, clientId, userName, password,
                poolId);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("9.  Invokes the AdminRespondToAuthChallenge");
        String session2 = authResponse1.session();
        adminRespondToAuthChallenge(identityProviderClient, userName, clientId,
mfaCode, session2);
        System.out.println(DASHES);

        System.out.println(DASHES);
        System.out.println("All Amazon Cognito operations were successfully
performed");
        System.out.println(DASHES);
    }

    // Respond to an authentication challenge.
    public static void adminRespondToAuthChallenge(CognitoIdentityProviderClient
identityProviderClient,
            String userName, String clientId, String mfaCode, String session) {
        System.out.println("SOFTWARE_TOKEN_MFA challenge is generated");
        Map<String, String> challengeResponses = new HashMap<>();

        challengeResponses.put("USERNAME", userName);
        challengeResponses.put("SOFTWARE_TOKEN_MFA_CODE", mfaCode);

        AdminRespondToAuthChallengeRequest respondToAuthChallengeRequest =
AdminRespondToAuthChallengeRequest.builder()
                .challengeName(ChallengeNameType.SOFTWARE_TOKEN_MFA)
                .clientId(clientId)
                .challengeResponses(challengeResponses)
                .session(session)
```

```
                .build();

        AdminRespondToAuthChallengeResponse respondToAuthChallengeResult =
identityProviderClient
                .adminRespondToAuthChallenge(respondToAuthChallengeRequest);

System.out.println("respondToAuthChallengeResult.getAuthenticationResult()"
                + respondToAuthChallengeResult.authenticationResult());
    }

    // Verify the TOTP and register for MFA.
    public static void verifyTOTP(CognitoIdentityProviderClient
identityProviderClient, String session, String code) {
        try {
            VerifySoftwareTokenRequest tokenRequest =
VerifySoftwareTokenRequest.builder()
                    .userCode(code)
                    .session(session)
                    .build();

            VerifySoftwareTokenResponse verifyResponse =
identityProviderClient.verifySoftwareToken(tokenRequest);
            System.out.println("The status of the token is " +
verifyResponse.statusAsString());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static AdminInitiateAuthResponse
initiateAuth(CognitoIdentityProviderClient identityProviderClient,
            String clientId, String userName, String password, String userPoolId)
{
        try {
            Map<String, String> authParameters = new HashMap<>();
            authParameters.put("USERNAME", userName);
            authParameters.put("PASSWORD", password);

            AdminInitiateAuthRequest authRequest =
AdminInitiateAuthRequest.builder()
                    .clientId(clientId)
                    .userPoolId(userPoolId)
```

```
                    .authParameters(authParameters)
                    .authFlow(AuthFlowType.ADMIN_USER_PASSWORD_AUTH)
                    .build();

            AdminInitiateAuthResponse response =
identityProviderClient.adminInitiateAuth(authRequest);
            System.out.println("Result Challenge is : " +
response.challengeName());
            return response;

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }

        return null;
    }

    public static String getSecretForAppMFA(CognitoIdentityProviderClient
identityProviderClient, String session) {
        AssociateSoftwareTokenRequest softwareTokenRequest =
AssociateSoftwareTokenRequest.builder()
                .session(session)
                .build();

        AssociateSoftwareTokenResponse tokenResponse = identityProviderClient
                .associateSoftwareToken(softwareTokenRequest);
        String secretCode = tokenResponse.secretCode();
        System.out.println("Enter this token into Google Authenticator");
        System.out.println(secretCode);
        return tokenResponse.session();
    }

    public static void confirmSignUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String code,
            String userName) {
        try {
            ConfirmSignUpRequest signUpRequest = ConfirmSignUpRequest.builder()
                    .clientId(clientId)
                    .confirmationCode(code)
                    .username(userName)
                    .build();

            identityProviderClient.confirmSignUp(signUpRequest);
```

```
            System.out.println(userName + " was confirmed");

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void resendConfirmationCode(CognitoIdentityProviderClient
identityProviderClient, String clientId,
            String userName) {
        try {
            ResendConfirmationCodeRequest codeRequest =
ResendConfirmationCodeRequest.builder()
                    .clientId(clientId)
                    .username(userName)
                    .build();

            ResendConfirmationCodeResponse response =
identityProviderClient.resendConfirmationCode(codeRequest);
            System.out.println("Method of delivery is " +
response.codeDeliveryDetails().deliveryMediumAsString());

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void signUp(CognitoIdentityProviderClient
identityProviderClient, String clientId, String userName,
            String password, String email) {
        AttributeType userAttrs = AttributeType.builder()
                .name("email")
                .value(email)
                .build();

        List<AttributeType> userAttrsList = new ArrayList<>();
        userAttrsList.add(userAttrs);
        try {
            SignUpRequest signUpRequest = SignUpRequest.builder()
                    .userAttributes(userAttrsList)
                    .username(userName)
                    .clientId(clientId)
```

```
                    .password(password)
                    .build();

            identityProviderClient.signUp(signUpRequest);
            System.out.println("User has been signed up ");

        } catch (CognitoIdentityProviderException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
            System.exit(1);
        }
    }

    public static void getAdminUser(CognitoIdentityProviderClient
 identityProviderClient, String userName,
            String poolId) {
        try {
            AdminGetUserRequest userRequest = AdminGetUserRequest.builder()
                    .username(userName)
                    .userPoolId(poolId)
                    .build();

            AdminGetUserResponse response =
 identityProviderClient.adminGetUser(userRequest);
            System.out.println("User status " + response.userStatusAsString());

        } catch (CognitoIdentityProviderException 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 .
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice
  - ConfirmSignUp

- InitiateAuth
- ListUsers
- ResendConfirmationCode
- RespondToAuthChallenge
- SignUp
- VerifySoftwareToken

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Para obter a melhor experiência, clone o GitHub repositório e execute este exemplo. O código a seguir representa uma amostra da aplicação de exemplo completa.

```
import { logger } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { signUp } from "../../../actions/sign-up.js";
import { FILE_USER_POOLS } from "./constants.js";
import { getSecondValuesFromEntries } from "@aws-doc-sdk-examples/lib/utils/util-
csv.js";

const validateClient = (clientId) => {
  if (!clientId) {
    throw new Error(
      `App client id is missing. Did you run 'create-user-pool'?`,
    );
  }
};

const validateUser = (username, password, email) => {
  if (!(username && password && email)) {
    throw new Error(
      `Username, password, and email must be provided as arguments to the 'sign-
up' command.`,
```

```
    );
  }
};

const signUpHandler = async (commands) => {
  const [_, username, password, email] = commands;

  try {
    validateUser(username, password, email);
    /**
     * @type {string[]}
     */
    const values = getSecondValuesFromEntries(FILE_USER_POOLS);
    const clientId = values[0];
    validateClient(clientId);
    logger.log("Signing up.");
    await signUp({ clientId, username, password, email });
    logger.log(`Signed up. A confirmation email has been sent to: ${email}.`);
    logger.log(
      `Run 'confirm-sign-up ${username} <code>' to confirm your account.`,
    );
  } catch (err) {
    logger.error(err);
  }
};

export { signUpHandler };

const signUp = ({ clientId, username, password, email }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new SignUpCommand({
    ClientId: clientId,
    Username: username,
    Password: password,
    UserAttributes: [{ Name: "email", Value: email }],
  });

  return client.send(command);
};

import { logger } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { confirmSignUp } from "../../../actions/confirm-sign-up.js";
import { FILE_USER_POOLS } from "./constants.js";
```

```
import { getSecondValuesFromEntries } from "@aws-doc-sdk-examples/lib/utils/util-
csv.js";

const validateClient = (clientId) => {
  if (!clientId) {
    throw new Error(
      `App client id is missing. Did you run 'create-user-pool'?`,
    );
  }
};

const validateUser = (username) => {
  if (!username) {
    throw new Error(
      `Username name is missing. It must be provided as an argument to the
 'confirm-sign-up' command.`,
    );
  }
};

const validateCode = (code) => {
  if (!code) {
    throw new Error(
      `Verification code is missing. It must be provided as an argument to the
 'confirm-sign-up' command.`,
    );
  }
};

const confirmSignUpHandler = async (commands) => {
  const [_, username, code] = commands;

  try {
    validateUser(username);
    validateCode(code);
    /**
     * @type {string[]}
     */
    const values = getSecondValuesFromEntries(FILE_USER_POOLS);
    const clientId = values[0];
    validateClient(clientId);
    logger.log("Confirming user.");
    await confirmSignUp({ clientId, username, code });
    logger.log(
```

```
      `User confirmed. Run 'admin-initiate-auth ${username} <password>' to sign
 in.`,
    );
  } catch (err) {
    logger.error(err);
  }
};

export { confirmSignUpHandler };

const confirmSignUp = ({ clientId, username, code }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new ConfirmSignUpCommand({
    ClientId: clientId,
    Username: username,
    ConfirmationCode: code,
  });

  return client.send(command);
};

import qrcode from "qrcode-terminal";
import { logger } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { adminInitiateAuth } from "../../../actions/admin-initiate-auth.js";
import { associateSoftwareToken } from "../../../actions/associate-software-
token.js";
import { FILE_USER_POOLS } from "./constants.js";
import { getFirstEntry } from "@aws-doc-sdk-examples/lib/utils/util-csv.js";

const handleMfaSetup = async (session, username) => {
  const { SecretCode, Session } = await associateSoftwareToken(session);

  // Store the Session for use with 'VerifySoftwareToken'.
  process.env.SESSION = Session;

  console.log(
    "Scan this code in your preferred authenticator app, then run 'verify-
software-token' to finish the setup.",
  );
  qrcode.generate(
    `otpauth://totp/${username}?secret=${SecretCode}`,
    { small: true },
    console.log,
```

```
  );
};

const handleSoftwareTokenMfa = (session) => {
  // Store the Session for use with 'AdminRespondToAuthChallenge'.
  process.env.SESSION = session;
};

const validateClient = (id) => {
  if (!id) {
    throw new Error(
      `User pool client id is missing. Did you run 'create-user-pool'?`,
    );
  }
};

const validateId = (id) => {
  if (!id) {
    throw new Error(`User pool id is missing. Did you run 'create-user-pool'?`);
  }
};

const validateUser = (username, password) => {
  if (!(username && password)) {
    throw new Error(
      `Username and password must be provided as arguments to the 'admin-
initiate-auth' command.`,
    );
  }
};

const adminInitiateAuthHandler = async (commands) => {
  const [_, username, password] = commands;

  try {
    validateUser(username, password);

    const [userPoolId, clientId] = getFirstEntry(FILE_USER_POOLS);
    validateId(userPoolId);
    validateClient(clientId);

    logger.log("Signing in.");
    const { ChallengeName, Session } = await adminInitiateAuth({
      clientId,
```

```
      userPoolId,
      username,
      password,
    });

    if (ChallengeName === "MFA_SETUP") {
      logger.log("MFA setup is required.");
      return handleMfaSetup(Session, username);
    }

    if (ChallengeName === "SOFTWARE_TOKEN_MFA") {
      handleSoftwareTokenMfa(Session);
      logger.log(`Run 'admin-respond-to-auth-challenge ${username} <totp>'`);
    }
  } catch (err) {
    logger.error(err);
  }
};

export { adminInitiateAuthHandler };

const adminInitiateAuth = ({ clientId, userPoolId, username, password }) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new AdminInitiateAuthCommand({
    ClientId: clientId,
    UserPoolId: userPoolId,
    AuthFlow: AuthFlowType.ADMIN_USER_PASSWORD_AUTH,
    AuthParameters: { USERNAME: username, PASSWORD: password },
  });

  return client.send(command);
};

import { logger } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { adminRespondToAuthChallenge } from "../../../actions/admin-respond-to-
auth-challenge.js";
import { getFirstEntry } from "@aws-doc-sdk-examples/lib/utils/util-csv.js";
import { FILE_USER_POOLS } from "./constants.js";

const verifyUsername = (username) => {
  if (!username) {
    throw new Error(
```

```
      `Username is missing. It must be provided as an argument to the 'admin-
respond-to-auth-challenge' command.`,
    );
  }
};

const verifyTotp = (totp) => {
  if (!totp) {
    throw new Error(
      `Time-based one-time password (TOTP) is missing. It must be provided as an
 argument to the 'admin-respond-to-auth-challenge' command.`,
    );
  }
};

const storeAccessToken = (token) => {
  process.env.AccessToken = token;
};

const adminRespondToAuthChallengeHandler = async (commands) => {
  const [_, username, totp] = commands;

  try {
    verifyUsername(username);
    verifyTotp(totp);

    const [userPoolId, clientId] = getFirstEntry(FILE_USER_POOLS);
    const session = process.env.SESSION;

    const { AuthenticationResult } = await adminRespondToAuthChallenge({
      clientId,
      userPoolId,
      username,
      totp,
      session,
    });

    storeAccessToken(AuthenticationResult.AccessToken);

    logger.log("Successfully authenticated.");
  } catch (err) {
    logger.error(err);
  }
};
```

```
export { adminRespondToAuthChallengeHandler };

const respondToAuthChallenge = ({
  clientId,
  username,
  session,
  userPoolId,
  code,
}) => {
  const client = new CognitoIdentityProviderClient({});

  const command = new RespondToAuthChallengeCommand({
    ChallengeName: ChallengeNameType.SOFTWARE_TOKEN_MFA,
    ChallengeResponses: {
      SOFTWARE_TOKEN_MFA_CODE: code,
      USERNAME: username,
    },
    ClientId: clientId,
    UserPoolId: userPoolId,
    Session: session,
  });

  return client.send(command);
};

import { logger } from "@aws-doc-sdk-examples/lib/utils/util-log.js";
import { verifySoftwareToken } from "../../../actions/verify-software-token.js";

const validateTotp = (totp) => {
  if (!totp) {
    throw new Error(
      `Time-based one-time password (TOTP) must be provided to the 'validate-
software-token' command.`,
    );
  }
};
const verifySoftwareTokenHandler = async (commands) => {
  const [_, totp] = commands;

  try {
    validateTotp(totp);

    logger.log("Verifying TOTP.");
```

```
    await verifySoftwareToken(totp);
    logger.log("TOTP Verified. Run 'admin-initiate-auth' again to sign-in.");
  } catch (err) {
    logger.error(err);
  }
};

export { verifySoftwareTokenHandler };

const verifySoftwareToken = (totp) => {
  const client = new CognitoIdentityProviderClient({});

  // The 'Session' is provided in the response to 'AssociateSoftwareToken'.
  const session = process.env.SESSION;

  if (!session) {
    throw new Error(
      "Missing a valid Session. Did you run 'admin-initiate-auth'?",
    );
  }

  const command = new VerifySoftwareTokenCommand({
    Session: session,
    UserCode: totp,
  });

  return client.send(command);
};
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para JavaScript .
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice
  - ConfirmSignUp
  - InitiateAuth
  - ListUsers

- ResendConfirmationCode
- RespondToAuthChallenge
- SignUp
- VerifySoftwareToken

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
/**
 Before running this Kotlin code example, set up your development environment,
 including your credentials.

 For more information, see the following documentation:
 https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html

 TIP: To set up the required user pool, run the AWS Cloud Development
 Kit (AWS CDK) script provided in this GitHub repo at resources/cdk/
cognito_scenario_user_pool_with_mfa.

 This code example performs the following operations:

 1. Invokes the signUp method to sign up a user.
 2. Invokes the adminGetUser method to get the user's confirmation status.
 3. Invokes the ResendConfirmationCode method if the user requested another code.
 4. Invokes the confirmSignUp method.
 5. Invokes the initiateAuth to sign in. This results in being prompted to
 set up TOTP (time-based one-time password). (The response is “ChallengeName”:
 “MFA_SETUP”).
 6. Invokes the AssociateSoftwareToken method to generate a TOTP MFA private key.
 This can be used with Google Authenticator.
 7. Invokes the VerifySoftwareToken method to verify the TOTP and register for
 MFA.
```

```
 8. Invokes the AdminInitiateAuth to sign in again. This results in being
 prompted to submit a TOTP (Response: “ChallengeName”: “SOFTWARE_TOKEN_MFA”).
 9. Invokes the AdminRespondToAuthChallenge to get back a token.
 */

suspend fun main(args: Array<String>) {
    val usage = """
        Usage:
            <clientId> <poolId>
        Where:
            clientId - The app client Id value that you can get from the AWS CDK
 script.
            poolId - The pool Id that you can get from the AWS CDK script.
    """

    if (args.size != 2) {
        println(usage)
        exitProcess(1)
    }

    val clientId = args[0]
    val poolId = args[1]

    // Use the console to get data from the user.
    println("*** Enter your use name")
    val inOb = Scanner(System.`in`)
    val userName = inOb.nextLine()
    println(userName)

    println("*** Enter your password")
    val password: String = inOb.nextLine()

    println("*** Enter your email")
    val email = inOb.nextLine()

    println("*** Signing up $userName")
    signUp(clientId, userName, password, email)

    println("*** Getting $userName in the user pool")
    getAdminUser(userName, poolId)

    println("*** Conformation code sent to $userName. Would you like to send a
 new code? (Yes/No)")
    val ans = inOb.nextLine()
```

```
    if (ans.compareTo("Yes") == 0) {
        println("*** Sending a new confirmation code")
        resendConfirmationCode(clientId, userName)
    }
    println("*** Enter the confirmation code that was emailed")
    val code = inOb.nextLine()
    confirmSignUp(clientId, code, userName)

    println("*** Rechecking the status of $userName in the user pool")
    getAdminUser(userName, poolId)

    val authResponse = checkAuthMethod(clientId, userName, password, poolId)
    val mySession = authResponse.session
    val newSession = getSecretForAppMFA(mySession)
    println("*** Enter the 6-digit code displayed in Google Authenticator")
    val myCode = inOb.nextLine()

    // Verify the TOTP and register for MFA.
    verifyTOTP(newSession, myCode)
    println("*** Re-enter a 6-digit code displayed in Google Authenticator")
    val mfaCode: String = inOb.nextLine()
    val authResponse1 = checkAuthMethod(clientId, userName, password, poolId)
    val session2 = authResponse1.session
    adminRespondToAuthChallenge(userName, clientId, mfaCode, session2)
}

suspend fun checkAuthMethod(
    clientIdVal: String,
    userNameVal: String,
    passwordVal: String,
    userPoolIdVal: String,
): AdminInitiateAuthResponse {
    val authParas = mutableMapOf<String, String>()
    authParas["USERNAME"] = userNameVal
    authParas["PASSWORD"] = passwordVal

    val authRequest =
        AdminInitiateAuthRequest {
            clientId = clientIdVal
            userPoolId = userPoolIdVal
            authParameters = authParas
            authFlow = AuthFlowType.AdminUserPasswordAuth
        }
```

```
    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.adminInitiateAuth(authRequest)
        println("Result Challenge is ${response.challengeName}")
        return response
    }
}

suspend fun resendConfirmationCode(
    clientIdVal: String?,
    userNameVal: String?,
) {
    val codeRequest =
        ResendConfirmationCodeRequest {
            clientId = clientIdVal
            username = userNameVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.resendConfirmationCode(codeRequest)
        println("Method of delivery is " +
 (response.codeDeliveryDetails?.deliveryMedium))
    }
}

// Respond to an authentication challenge.
suspend fun adminRespondToAuthChallenge(
    userName: String,
    clientIdVal: String?,
    mfaCode: String,
    sessionVal: String?,
) {
    println("SOFTWARE_TOKEN_MFA challenge is generated")
    val challengeResponsesOb = mutableMapOf<String, String>()
    challengeResponsesOb["USERNAME"] = userName
    challengeResponsesOb["SOFTWARE_TOKEN_MFA_CODE"] = mfaCode

    val adminRespondToAuthChallengeRequest =
        AdminRespondToAuthChallengeRequest {
            challengeName = ChallengeNameType.SoftwareTokenMfa
            clientId = clientIdVal
            challengeResponses = challengeResponsesOb
```

```
            session = sessionVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val respondToAuthChallengeResult =
 identityProviderClient.adminRespondToAuthChallenge(adminRespondToAuthChallengeRequest)
        println("respondToAuthChallengeResult.getAuthenticationResult()
 ${respondToAuthChallengeResult.authenticationResult}")
    }
}

// Verify the TOTP and register for MFA.
suspend fun verifyTOTP(
    sessionVal: String?,
    codeVal: String?,
) {
    val tokenRequest =
        VerifySoftwareTokenRequest {
            userCode = codeVal
            session = sessionVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val verifyResponse =
 identityProviderClient.verifySoftwareToken(tokenRequest)
        println("The status of the token is ${verifyResponse.status}")
    }
}

suspend fun getSecretForAppMFA(sessionVal: String?): String? {
    val softwareTokenRequest =
        AssociateSoftwareTokenRequest {
            session = sessionVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val tokenResponse =
 identityProviderClient.associateSoftwareToken(softwareTokenRequest)
        val secretCode = tokenResponse.secretCode
        println("Enter this token into Google Authenticator")
        println(secretCode)
```

```
        return tokenResponse.session
    }
}

suspend fun confirmSignUp(
    clientIdVal: String?,
    codeVal: String?,
    userNameVal: String?,
) {
    val signUpRequest =
        ConfirmSignUpRequest {
            clientId = clientIdVal
            confirmationCode = codeVal
            username = userNameVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.confirmSignUp(signUpRequest)
        println("$userNameVal  was confirmed")
    }
}

suspend fun getAdminUser(
    userNameVal: String?,
    poolIdVal: String?,
) {
    val userRequest =
        AdminGetUserRequest {
            username = userNameVal
            userPoolId = poolIdVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        val response = identityProviderClient.adminGetUser(userRequest)
        println("User status ${response.userStatus}")
    }
}

suspend fun signUp(
    clientIdVal: String?,
    userNameVal: String?,
    passwordVal: String?,
```

```
    emailVal: String?,
) {
    val userAttrs =
        AttributeType {
            name = "email"
            value = emailVal
        }

    val userAttrsList = mutableListOf<AttributeType>()
    userAttrsList.add(userAttrs)
    val signUpRequest =
        SignUpRequest {
            userAttributes = userAttrsList
            username = userNameVal
            clientId = clientIdVal
            password = passwordVal
        }

    CognitoIdentityProviderClient.fromEnvironment { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.signUp(signUpRequest)
        println("User has been signed up")
    }
}
```

- Consulte detalhes da API nos tópicos a seguir na Referência de API do AWS SDK para Kotlin.
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice
  - ConfirmSignUp
  - InitiateAuth
  - ListUsers
  - ResendConfirmationCode
  - RespondToAuthChallenge
  - SignUp

- VerifySoftwareToken

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Crie uma classe que englobe as funções do Amazon Cognito usadas no cenário.

```
class CognitoIdentityProviderWrapper:
    """Encapsulates Amazon Cognito actions"""

    def __init__(self, cognito_idp_client, user_pool_id, client_id,
 client_secret=None):
        """
        :param cognito_idp_client: A Boto3 Amazon Cognito Identity Provider
 client.
        :param user_pool_id: The ID of an existing Amazon Cognito user pool.
        :param client_id: The ID of a client application registered with the user
 pool.
        :param client_secret: The client secret, if the client has a secret.
        """
        self.cognito_idp_client = cognito_idp_client
        self.user_pool_id = user_pool_id
        self.client_id = client_id
        self.client_secret = client_secret


    def _secret_hash(self, user_name):
        """
        Calculates a secret hash from a user name and a client secret.

        :param user_name: The user name to use when calculating the hash.
        :return: The secret hash.
        """
        key = self.client_secret.encode()
        msg = bytes(user_name + self.client_id, "utf-8")
```

```
        secret_hash = base64.b64encode(
            hmac.new(key, msg, digestmod=hashlib.sha256).digest()
        ).decode()
        logger.info("Made secret hash for %s: %s.", user_name, secret_hash)
        return secret_hash

    def sign_up_user(self, user_name, password, user_email):
        """
        Signs up a new user with Amazon Cognito. This action prompts Amazon
Cognito
        to send an email to the specified email address. The email contains a
code that
        can be used to confirm the user.

        When the user already exists, the user status is checked to determine
whether
        the user has been confirmed.

        :param user_name: The user name that identifies the new user.
        :param password: The password for the new user.
        :param user_email: The email address for the new user.
        :return: True when the user is already confirmed with Amazon Cognito.
                 Otherwise, false.
        """
        try:
            kwargs = {
                "ClientId": self.client_id,
                "Username": user_name,
                "Password": password,
                "UserAttributes": [{"Name": "email", "Value": user_email}],
            }
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            response = self.cognito_idp_client.sign_up(**kwargs)
            confirmed = response["UserConfirmed"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "UsernameExistsException":
                response = self.cognito_idp_client.admin_get_user(
                    UserPoolId=self.user_pool_id, Username=user_name
                )
                logger.warning(
                    "User %s exists and is %s.", user_name,
response["UserStatus"]
                )
```

```
                confirmed = response["UserStatus"] == "CONFIRMED"
            else:
                logger.error(
                    "Couldn't sign up %s. Here's why: %s: %s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        return confirmed


    def resend_confirmation(self, user_name):
        """
        Prompts Amazon Cognito to resend an email with a new confirmation code.

        :param user_name: The name of the user who will receive the email.
        :return: Delivery information about where the email is sent.
        """
        try:
            kwargs = {"ClientId": self.client_id, "Username": user_name}
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            response = self.cognito_idp_client.resend_confirmation_code(**kwargs)
            delivery = response["CodeDeliveryDetails"]
        except ClientError as err:
            logger.error(
                "Couldn't resend confirmation to %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return delivery


    def confirm_user_sign_up(self, user_name, confirmation_code):
        """
        Confirms a previously created user. A user must be confirmed before they
        can sign in to Amazon Cognito.

        :param user_name: The name of the user to confirm.
```

```
        :param confirmation_code: The confirmation code sent to the user's
registered
                                  email address.
        :return: True when the confirmation succeeds.
        """
        try:
            kwargs = {
                "ClientId": self.client_id,
                "Username": user_name,
                "ConfirmationCode": confirmation_code,
            }
            if self.client_secret is not None:
                kwargs["SecretHash"] = self._secret_hash(user_name)
            self.cognito_idp_client.confirm_sign_up(**kwargs)
        except ClientError as err:
            logger.error(
                "Couldn't confirm sign up for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return True


    def list_users(self):
        """
        Returns a list of the users in the current user pool.

        :return: The list of users.
        """
        try:
            response =
self.cognito_idp_client.list_users(UserPoolId=self.user_pool_id)
            users = response["Users"]
        except ClientError as err:
            logger.error(
                "Couldn't list users for %s. Here's why: %s: %s",
                self.user_pool_id,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
```

```
        else:
            return users


    def start_sign_in(self, user_name, password):
        """
        Starts the sign-in process for a user by using administrator credentials.
        This method of signing in is appropriate for code running on a secure
 server.

        If the user pool is configured to require MFA and this is the first sign-
in
        for the user, Amazon Cognito returns a challenge response to set up an
        MFA application. When this occurs, this function gets an MFA secret from
        Amazon Cognito and returns it to the caller.

        :param user_name: The name of the user to sign in.
        :param password: The user's password.
        :return: The result of the sign-in attempt. When sign-in is successful,
 this
                 returns an access token that can be used to get AWS credentials.
 Otherwise,
                 Amazon Cognito returns a challenge to set up an MFA application,
                 or a challenge to enter an MFA code from a registered MFA
 application.
        """
        try:
            kwargs = {
                "UserPoolId": self.user_pool_id,
                "ClientId": self.client_id,
                "AuthFlow": "ADMIN_USER_PASSWORD_AUTH",
                "AuthParameters": {"USERNAME": user_name, "PASSWORD": password},
            }
            if self.client_secret is not None:
                kwargs["AuthParameters"]["SECRET_HASH"] =
 self._secret_hash(user_name)
            response = self.cognito_idp_client.admin_initiate_auth(**kwargs)
            challenge_name = response.get("ChallengeName", None)
            if challenge_name == "MFA_SETUP":
                if (
                    "SOFTWARE_TOKEN_MFA"
                    in response["ChallengeParameters"]["MFAS_CAN_SETUP"]
                ):
                    response.update(self.get_mfa_secret(response["Session"]))
```

```
                else:
                    raise RuntimeError(
                        "The user pool requires MFA setup, but the user pool is
not "
                        "configured for TOTP MFA. This example requires TOTP
MFA."
                    )
        except ClientError as err:
            logger.error(
                "Couldn't start sign in for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response


    def get_mfa_secret(self, session):
        """
        Gets a token that can be used to associate an MFA application with the
user.

        :param session: Session information returned from a previous call to
initiate
                        authentication.
        :return: An MFA token that can be used to set up an MFA application.
        """
        try:
            response =
self.cognito_idp_client.associate_software_token(Session=session)
        except ClientError as err:
            logger.error(
                "Couldn't get MFA secret. Here's why: %s: %s",
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            response.pop("ResponseMetadata", None)
            return response
```

```
   def verify_mfa(self, session, user_code):
       """
       Verify a new MFA application that is associated with a user.

       :param session: Session information returned from a previous call to
initiate
                       authentication.
       :param user_code: A code generated by the associated MFA application.
       :return: Status that indicates whether the MFA application is verified.
       """
       try:
           response = self.cognito_idp_client.verify_software_token(
               Session=session, UserCode=user_code
           )
       except ClientError as err:
           logger.error(
               "Couldn't verify MFA. Here's why: %s: %s",
               err.response["Error"]["Code"],
               err.response["Error"]["Message"],
           )
           raise
       else:
           response.pop("ResponseMetadata", None)
           return response


   def respond_to_mfa_challenge(self, user_name, session, mfa_code):
       """
       Responds to a challenge for an MFA code. This completes the second step
of
       a two-factor sign-in. When sign-in is successful, it returns an access
token
       that can be used to get AWS credentials from Amazon Cognito.

       :param user_name: The name of the user who is signing in.
       :param session: Session information returned from a previous call to
initiate
                       authentication.
       :param mfa_code: A code generated by the associated MFA application.
       :return: The result of the authentication. When successful, this contains
an
                access token for the user.
       """
```

```
        try:
            kwargs = {
                "UserPoolId": self.user_pool_id,
                "ClientId": self.client_id,
                "ChallengeName": "SOFTWARE_TOKEN_MFA",
                "Session": session,
                "ChallengeResponses": {
                    "USERNAME": user_name,
                    "SOFTWARE_TOKEN_MFA_CODE": mfa_code,
                },
            }
            if self.client_secret is not None:
                kwargs["ChallengeResponses"]["SECRET_HASH"] = self._secret_hash(
                    user_name
                )
            response =
self.cognito_idp_client.admin_respond_to_auth_challenge(**kwargs)
            auth_result = response["AuthenticationResult"]
        except ClientError as err:
            if err.response["Error"]["Code"] == "ExpiredCodeException":
                logger.warning(
                    "Your MFA code has expired or has been used already. You
might have "
                    "to wait a few seconds until your app shows you a new code."
                )
            else:
                logger.error(
                    "Couldn't respond to mfa challenge for %s. Here's why: %s:
%s",
                    user_name,
                    err.response["Error"]["Code"],
                    err.response["Error"]["Message"],
                )
                raise
        else:
            return auth_result


    def confirm_mfa_device(
        self,
        user_name,
        device_key,
        device_group_key,
        device_password,
```

```
        access_token,
        aws_srp,
    ):
        """
        Confirms an MFA device to be tracked by Amazon Cognito. When a device is
        tracked, its key and password can be used to sign in without requiring a
new
        MFA code from the MFA application.

        :param user_name: The user that is associated with the device.
        :param device_key: The key of the device, returned by Amazon Cognito.
        :param device_group_key: The group key of the device, returned by Amazon
Cognito.
        :param device_password: The password that is associated with the device.
        :param access_token: The user's access token.
        :param aws_srp: A class that helps with Secure Remote Password (SRP)
                        calculations. The scenario associated with this example
uses
                        the warrant package.
        :return: True when the user must confirm the device. Otherwise, False.
When
                 False, the device is automatically confirmed and tracked.
        """
        srp_helper = aws_srp.AWSSRP(
            username=user_name,
            password=device_password,
            pool_id="_",
            client_id=self.client_id,
            client_secret=None,
            client=self.cognito_idp_client,
        )
        device_and_pw = f"{device_group_key}{device_key}:{device_password}"
        device_and_pw_hash = aws_srp.hash_sha256(device_and_pw.encode("utf-8"))
        salt = aws_srp.pad_hex(aws_srp.get_random(16))
        x_value = aws_srp.hex_to_long(aws_srp.hex_hash(salt +
device_and_pw_hash))
        verifier = aws_srp.pad_hex(pow(srp_helper.val_g, x_value,
srp_helper.big_n))
        device_secret_verifier_config = {
            "PasswordVerifier": base64.standard_b64encode(
                bytearray.fromhex(verifier)
            ).decode("utf-8"),
            "Salt":
base64.standard_b64encode(bytearray.fromhex(salt)).decode("utf-8"),
```

```
        }
        try:
            response = self.cognito_idp_client.confirm_device(
                AccessToken=access_token,
                DeviceKey=device_key,
                DeviceSecretVerifierConfig=device_secret_verifier_config,
            )
            user_confirm = response["UserConfirmationNecessary"]
        except ClientError as err:
            logger.error(
                "Couldn't confirm mfa device %s. Here's why: %s: %s",
                device_key,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return user_confirm


    def sign_in_with_tracked_device(
        self,
        user_name,
        password,
        device_key,
        device_group_key,
        device_password,
        aws_srp,
    ):
        """
        Signs in to Amazon Cognito as a user who has a tracked device. Signing in
        with a tracked device lets a user sign in without entering a new MFA
 code.

        Signing in with a tracked device requires that the client respond to the
 SRP
        protocol. The scenario associated with this example uses the warrant
 package
        to help with SRP calculations.

        For more information on SRP, see https://en.wikipedia.org/wiki/
Secure_Remote_Password_protocol.

        :param user_name: The user that is associated with the device.
```

```
        :param password: The user's password.
        :param device_key: The key of a tracked device.
        :param device_group_key: The group key of a tracked device.
        :param device_password: The password that is associated with the device.
        :param aws_srp: A class that helps with SRP calculations. The scenario
                        associated with this example uses the warrant package.
        :return: The result of the authentication. When successful, this contains
an
                 access token for the user.
        """
        try:
            srp_helper = aws_srp.AWSSRP(
                username=user_name,
                password=device_password,
                pool_id="_",
                client_id=self.client_id,
                client_secret=None,
                client=self.cognito_idp_client,
            )

            response_init = self.cognito_idp_client.initiate_auth(
                ClientId=self.client_id,
                AuthFlow="USER_PASSWORD_AUTH",
                AuthParameters={
                    "USERNAME": user_name,
                    "PASSWORD": password,
                    "DEVICE_KEY": device_key,
                },
            )
            if response_init["ChallengeName"] != "DEVICE_SRP_AUTH":
                raise RuntimeError(
                    f"Expected DEVICE_SRP_AUTH challenge but got
{response_init['ChallengeName']}."
                )

            auth_params = srp_helper.get_auth_params()
            auth_params["DEVICE_KEY"] = device_key
            response_auth = self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_SRP_AUTH",
                ChallengeResponses=auth_params,
            )
            if response_auth["ChallengeName"] != "DEVICE_PASSWORD_VERIFIER":
                raise RuntimeError(
```

```
                    f"Expected DEVICE_PASSWORD_VERIFIER challenge but got "
                    f"{response_init['ChallengeName']}."
                )

            challenge_params = response_auth["ChallengeParameters"]
            challenge_params["USER_ID_FOR_SRP"] = device_group_key + device_key
            cr = srp_helper.process_challenge(challenge_params, {"USERNAME":
 user_name})
            cr["USERNAME"] = user_name
            cr["DEVICE_KEY"] = device_key
            response_verifier =
 self.cognito_idp_client.respond_to_auth_challenge(
                ClientId=self.client_id,
                ChallengeName="DEVICE_PASSWORD_VERIFIER",
                ChallengeResponses=cr,
            )
            auth_tokens = response_verifier["AuthenticationResult"]
        except ClientError as err:
            logger.error(
                "Couldn't start client sign in for %s. Here's why: %s: %s",
                user_name,
                err.response["Error"]["Code"],
                err.response["Error"]["Message"],
            )
            raise
        else:
            return auth_tokens
```

Crie uma classe que execute o cenário. Este exemplo também registra um dispositivo de MFA a ser rastreado pelo Amazon Cognito e mostra como fazer login usando uma senha e informações do dispositivo rastreado. Isso evita a necessidade de inserir um novo código de MFA.

```
def run_scenario(cognito_idp_client, user_pool_id, client_id):
    logging.basicConfig(level=logging.INFO, format="%(levelname)s: %(message)s")

    print("-" * 88)
    print("Welcome to the Amazon Cognito user signup with MFA demo.")
    print("-" * 88)
```

```
    cog_wrapper = CognitoIdentityProviderWrapper(
        cognito_idp_client, user_pool_id, client_id
    )

    user_name = q.ask("Let's sign up a new user. Enter a user name: ",
q.non_empty)
    password = q.ask("Enter a password for the user: ", q.non_empty)
    email = q.ask("Enter a valid email address that you own: ", q.non_empty)
    confirmed = cog_wrapper.sign_up_user(user_name, password, email)
    while not confirmed:
        print(
            f"User {user_name} requires confirmation. Check {email} for "
            f"a verification code."
        )
        confirmation_code = q.ask("Enter the confirmation code from the email: ")
        if not confirmation_code:
            if q.ask("Do you need another confirmation code (y/n)? ",
q.is_yesno):
                delivery = cog_wrapper.resend_confirmation(user_name)
                print(
                    f"Confirmation code sent by {delivery['DeliveryMedium']} "
                    f"to {delivery['Destination']}."
                )
        else:
            confirmed = cog_wrapper.confirm_user_sign_up(user_name,
confirmation_code)
    print(f"User {user_name} is confirmed and ready to use.")
    print("-" * 88)

    print("Let's get a list of users in the user pool.")
    q.ask("Press Enter when you're ready.")
    users = cog_wrapper.list_users()
    if users:
        print(f"Found {len(users)} users:")
        pp(users)
    else:
        print("No users found.")
    print("-" * 88)

    print("Let's sign in and get an access token.")
    auth_tokens = None
    challenge = "ADMIN_USER_PASSWORD_AUTH"
    response = {}
```

```
    while challenge is not None:
        if challenge == "ADMIN_USER_PASSWORD_AUTH":
            response = cog_wrapper.start_sign_in(user_name, password)
            challenge = response["ChallengeName"]
        elif response["ChallengeName"] == "MFA_SETUP":
            print("First, we need to set up an MFA application.")
            qr_img = qrcode.make(
                f"otpauth://totp/{user_name}?secret={response['SecretCode']}"
            )
            qr_img.save("qr.png")
            q.ask(
                "Press Enter to see a QR code on your screen. Scan it into an MFA
"
                "application, such as Google Authenticator."
            )
            webbrowser.open("qr.png")
            mfa_code = q.ask(
                "Enter the verification code from your MFA application: ",
q.non_empty
            )
            response = cog_wrapper.verify_mfa(response["Session"], mfa_code)
            print(f"MFA device setup {response['Status']}")
            print("Now that an MFA application is set up, let's sign in again.")
            print(
                "You might have to wait a few seconds for a new MFA code to
appear in "
                "your MFA application."
            )
            challenge = "ADMIN_USER_PASSWORD_AUTH"
        elif response["ChallengeName"] == "SOFTWARE_TOKEN_MFA":
            auth_tokens = None
            while auth_tokens is None:
                mfa_code = q.ask(
                    "Enter a verification code from your MFA application: ",
q.non_empty
                )
                auth_tokens = cog_wrapper.respond_to_mfa_challenge(
                    user_name, response["Session"], mfa_code
                )
            print(f"You're signed in as {user_name}.")
            print("Here's your access token:")
            pp(auth_tokens["AccessToken"])
            print("And your device information:")
            pp(auth_tokens["NewDeviceMetadata"])
```

```
            challenge = None
        else:
            raise Exception(f"Got unexpected challenge
 {response['ChallengeName']}")
    print("-" * 88)

    device_group_key = auth_tokens["NewDeviceMetadata"]["DeviceGroupKey"]
    device_key = auth_tokens["NewDeviceMetadata"]["DeviceKey"]
    device_password = base64.standard_b64encode(os.urandom(40)).decode("utf-8")

    print("Let's confirm your MFA device so you don't have re-enter MFA tokens
 for it.")
    q.ask("Press Enter when you're ready.")
    cog_wrapper.confirm_mfa_device(
        user_name,
        device_key,
        device_group_key,
        device_password,
        auth_tokens["AccessToken"],
        aws_srp,
    )
    print(f"Your device {device_key} is confirmed.")
    print("-" * 88)

    print(
        f"Now let's sign in as {user_name} from your confirmed device
 {device_key}.\n"
        f"Because this device is tracked by Amazon Cognito, you won't have to re-
enter an MFA code."
    )
    q.ask("Press Enter when ready.")
    auth_tokens = cog_wrapper.sign_in_with_tracked_device(
        user_name, password, device_key, device_group_key, device_password,
 aws_srp
    )
    print("You're signed in. Your access token is:")
    pp(auth_tokens["AccessToken"])
    print("-" * 88)

    print("Don't forget to delete your user pool when you're done with this
 example.")
    print("\nThanks for watching!")
    print("-" * 88)
```

```
def main():
    parser = argparse.ArgumentParser(
        description="Shows how to sign up a new user with Amazon Cognito and
 associate "
        "the user with an MFA application for multi-factor authentication."
    )
    parser.add_argument(
        "user_pool_id", help="The ID of the user pool to use for the example."
    )
    parser.add_argument(
        "client_id", help="The ID of the client application to use for the
 example."
    )
    args = parser.parse_args()
    try:
        run_scenario(boto3.client("cognito-idp"), args.user_pool_id,
 args.client_id)
    except Exception:
        logging.exception("Something went wrong with the demo.")


if __name__ == "__main__":
    main()
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Python (Boto3).
  - AdminGetUser
  - AdminInitiateAuth
  - AdminRespondToAuthChallenge
  - AssociateSoftwareToken
  - ConfirmDevice
  - ConfirmSignUp
  - InitiateAuth
  - ListUsers
  - ResendConfirmationCode
  - RespondToAuthChallenge
  - SignUp

- VerifySoftwareToken

Swift

SDK para Swift

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

O arquivo `Package.swift`.

```
// swift-tools-version: 5.9
//
// The swift-tools-version declares the minimum version of Swift required to
// build this package.

import PackageDescription

let package = Package(
    name: "cognito-scenario",
    // Let Xcode know the minimum Apple platforms supported.
    platforms: [
        .macOS(.v13),
        .iOS(.v15)
    ],
    dependencies: [
        // Dependencies declare other packages that this package depends on.
        .package(
            url: "https://github.com/awslabs/aws-sdk-swift",
            from: "1.0.0"),
        .package(
            url: "https://github.com/apple/swift-argument-parser.git",
            branch: "main"
        )
    ],
    targets: [
        // Targets are the basic building blocks of a package, defining a module
 or a test suite.
        // Targets can depend on other targets in this package and products
```

```
        // from dependencies.
        .executableTarget(
            name: "cognito-scenario",
            dependencies: [
                .product(name: "AWSCognitoIdentityProvider", package: "aws-sdk-
swift"),
                .product(name: "ArgumentParser", package: "swift-argument-
parser")
            ],
            path: "Sources")

    ]
)
```

O arquivo de código do Swift.

```
// An example demonstrating various features of Amazon Cognito. Before running
// this Swift code example, set up your development environment, including
// your credentials.
//
// For more information, see the following documentation:
// https://docs.aws.amazon.com/sdk-for-kotlin/latest/developer-guide/setup.html
//
// TIP: To set up the required user pool, run the AWS Cloud Development Kit
// (AWS CDK) script provided in this GitHub repo at
// resources/cdk/cognito_scenario_user_pool_with_mfa.
//
// This example performs the following functions:
//
// 1. Invokes the signUp method to sign up a user.
// 2. Invokes the adminGetUser method to get the user's confirmation status.
// 3. Invokes the ResendConfirmationCode method if the user requested another
//    code.
// 4. Invokes the confirmSignUp method.
// 5. Invokes the initiateAuth to sign in. This results in being prompted to
//    set up TOTP (time-based one-time password). (The response is
//    “ChallengeName”: “MFA_SETUP”).
// 6. Invokes the AssociateSoftwareToken method to generate a TOTP MFA private
//    key. This can be used with Google Authenticator.
// 7. Invokes the VerifySoftwareToken method to verify the TOTP and register
//    for MFA.
// 8. Invokes the AdminInitiateAuth to sign in again. This results in being
```

```
//    prompted to submit a TOTP (Response: “ChallengeName”:
//    “SOFTWARE_TOKEN_MFA”).
// 9. Invokes the AdminRespondToAuthChallenge to get back a token.

import ArgumentParser
import Foundation

import AWSClientRuntime
import AWSCognitoIdentityProvider

struct ExampleCommand: ParsableCommand {
    @Argument(help: "The application clientId.")
    var clientId: String
    @Argument(help: "The user pool ID to use.")
    var poolId: String
    @Option(help: "Name of the Amazon Region to use")
    var region = "us-east-1"

    static var configuration = CommandConfiguration(
        commandName: "cognito-scenario",
        abstract: """
        Demonstrates various features of Amazon Cognito.
        """,
        discussion: """
        """
    )

    /// Prompt for an input string of at least a minimum length.
    ///
    /// - Parameters:
    ///   - prompt: The prompt string to display.
    ///   - minLength: The minimum number of characters to allow in the
    ///     response. Default value is 0.
    ///
    /// - Returns: The entered string.
    func stringRequest(_ prompt: String, minLength: Int = 1) -> String {
        while true {
            print(prompt, terminator: "")
            let str = readLine()

            guard let str else {
                continue
            }
            if str.count >= minLength {
```

```
                return str
            } else {
                print("*** Response must be at least \(minLength) character(s)
long.")
            }
        }
    }

    /// Ask a yes/no question.
    ///
    /// - Parameter prompt: A prompt string to print.
    ///
    /// - Returns: `true` if the user answered "Y", otherwise `false`.
    func yesNoRequest(_ prompt: String) -> Bool {
        while true {
            let answer = stringRequest(prompt).lowercased()
            if answer == "y" || answer == "n" {
                return answer == "y"
            }
        }
    }

    /// Get information about a specific user in a user pool.
    ///
    /// - Parameters:
    ///   - cipClient: The Amazon Cognito Identity Provider client to use.
    ///   - userName: The user to retrieve information about.
    ///   - userPoolId: The user pool to search for the specified user.
    ///
    /// - Returns: `true` if the user's information was successfully
    ///   retrieved. Otherwise returns `false`.
    func adminGetUser(cipClient: CognitoIdentityProviderClient, userName: String,
                      userPoolId: String) async -> Bool {
        do {
            let output = try await cipClient.adminGetUser(
                input: AdminGetUserInput(
                    userPoolId: userPoolId,
                    username: userName
                )
            )

            guard let userStatus = output.userStatus else {
                print("*** Unable to get the user's status.")
                return false
```

```
            }

            print("User status: \(userStatus)")
            return true
        } catch {
            return false
        }
    }

    /// Create a new user in a user pool.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The ID of the app client to create a user for.
    ///   - userName: The username for the new user.
    ///   - password: The new user's password.
    ///   - email: The new user's email address.
    ///
    /// - Returns: `true` if successful; otherwise `false`.
    func signUp(cipClient: CognitoIdentityProviderClient, clientId: String,
userName: String, password: String, email: String) async -> Bool {
        let emailAttr = CognitoIdentityProviderClientTypes.AttributeType(
            name: "email",
            value: email
        )

        let userAttrsList = [emailAttr]

        do {
            _ = try await cipClient.signUp(
                input: SignUpInput(
                    clientId: clientId,
                    password: password,
                    userAttributes: userAttrsList,
                    username: userName
                )

            )

            print("=====> User \(userName) signed up.")
        } catch _ as AWSCognitoIdentityProvider.UsernameExistsException {
            print("*** The username \(userName) already exists. Please use a
different one.")
            return false
```

```
        } catch let error as AWSCognitoIdentityProvider.InvalidPasswordException
{
            print("*** Error: The specified password is invalid. Reason:
\(error.properties.message ?? "<none available>").")
            return false
        } catch _ as AWSCognitoIdentityProvider.ResourceNotFoundException {
            print("*** Error: The specified client ID (\(clientId)) doesn't
exist.")
            return false
        } catch {
            print("*** Unexpected error: \(error)")
            return false
        }

        return true
    }

    /// Requests a new confirmation code be sent to the given user's contact
    /// method.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The application client ID.
    ///   - userName: The user to resend a code for.
    ///
    /// - Returns: `true` if a new code was sent successfully, otherwise
    ///   `false`.
    func resendConfirmationCode(cipClient: CognitoIdentityProviderClient,
clientId: String,
                                userName: String) async -> Bool {
        do {
            let output = try await cipClient.resendConfirmationCode(
                input: ResendConfirmationCodeInput(
                    clientId: clientId,
                    username: userName
                )
            )

            guard let deliveryMedium = output.codeDeliveryDetails?.deliveryMedium
else {
                print("*** Unable to get the delivery method for the resent
code.")
                return false
            }
```

```
            print("=====> A new code has been sent by \(deliveryMedium)")
            return true
        } catch {
            print("*** Unable to resend the confirmation code to user
\(userName).")
            return false
        }
    }

    /// Submit a confirmation code for the specified user. This is the code as
    /// entered by the user after they've received it by email or text
    /// message.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - clientId: The app client ID the user is signing up for.
    ///   - userName: The username of the user whose code is being sent.
    ///   - code: The user's confirmation code.
    ///
    /// - Returns: `true` if the code was successfully confirmed; otherwise
`false`.
    func confirmSignUp(cipClient: CognitoIdentityProviderClient, clientId:
String,
                       userName: String, code: String) async -> Bool {
        do {
            _ = try await cipClient.confirmSignUp(
                input: ConfirmSignUpInput(
                    clientId: clientId,
                    confirmationCode: code,
                    username: userName
                )
            )

            print("=====> \(userName) has been confirmed.")
            return true
        } catch {
            print("=====> \(userName)'s code was entered incorrectly.")
            return false
        }
    }

    /// Begin an authentication session.
    ///
```

```
    /// - Parameters:
    ///   - cipClient: The `CongitoIdentityProviderClient` to use.
    ///   - clientId: The app client ID to use.
    ///   - userName: The username to check.
    ///   - password: The user's password.
    ///   - userPoolId: The user pool to use.
    ///
    /// - Returns: The session token associated with this authentication
    ///   session.
    func initiateAuth(cipClient: CognitoIdentityProviderClient, clientId: String,
                      userName: String, password: String,
                      userPoolId: String) async -> String? {
        var authParams: [String: String] = [:]

        authParams["USERNAME"] = userName
        authParams["PASSWORD"] = password

        do {
            let output = try await cipClient.adminInitiateAuth(
                input: AdminInitiateAuthInput(
                    authFlow:
CognitoIdentityProviderClientTypes.AuthFlowType.adminUserPasswordAuth,
                    authParameters: authParams,
                    clientId: clientId,
                    userPoolId: userPoolId
                )
            )

            guard let challengeName = output.challengeName else {
                print("*** Invalid response from the auth service.")
                return nil
            }

            print("=====> Response challenge is \(challengeName)")

            return output.session
        } catch _ as UserNotFoundException {
            print("*** The specified username, \(userName), doesn't exist.")
            return nil
        } catch _ as UserNotConfirmedException {
            print("*** The user \(userName) has not been confirmed.")
            return nil
        } catch {
            print("*** An unexpected error occurred.")
```

```
            return nil
        }
    }

    /// Request and display an MFA secret token that the user should enter
    /// into their authenticator to set it up for the user account.
    ///
    /// - Parameters:
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - authSession: The authentication session to request an MFA secret
    ///     for.
    ///
    /// - Returns: A string containing the MFA secret token that should be
    ///   entered into the authenticator software.
    func getSecretForAppMFA(cipClient: CognitoIdentityProviderClient,
authSession: String?) async -> String? {
        do {
            let output = try await cipClient.associateSoftwareToken(
                input: AssociateSoftwareTokenInput(
                    session: authSession
                )
            )

            guard let secretCode = output.secretCode else {
                print("*** Unable to get the secret code")
                return nil
            }

            print("=====> Enter this token into Google Authenticator:
\(secretCode)")
            return output.session
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
            return nil
        } catch {
            print("*** An unexpected error occurred getting the secret for the
app's MFA.")
            return nil
        }
    }

    /// Confirm that the user's TOTP authenticator is configured correctly by
    /// sending a code to it to check that it matches successfully.
    ///
```

```
    /// - Parameters:
    ///   - cipClient: The `CongnitoIdentityProviderClient` to use.
    ///   - session: An authentication session previously returned by an
    ///     `associateSoftwareToken()` call.
    ///   - mfaCode: The 6-digit code currently displayed by the user's
    ///     authenticator, as provided by the user.
    func verifyTOTP(cipClient: CognitoIdentityProviderClient, session: String?,
mfaCode: String?) async {
        do {
            let output = try await cipClient.verifySoftwareToken(
                input: VerifySoftwareTokenInput(
                    session: session,
                    userCode: mfaCode
                )
            )

            guard let tokenStatus = output.status else {
                print("*** Unable to get the token's status.")
                return
            }
            print("=====> The token's status is: \(tokenStatus)")
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
            return
        } catch _ as CodeMismatchException {
            print("*** The specified MFA code doesn't match the expected value.")
            return
        } catch _ as UserNotFoundException {
            print("*** The specified username doesn't exist.")
            return
        } catch _ as UserNotConfirmedException {
            print("*** The user has not been confirmed.")
            return
        } catch {
            print("*** Error verifying the MFA token!")
            return
        }
    }

    /// Respond to the authentication challenge received from Cognito after
    /// initiating an authentication session. This involves sending a current
    /// MFA code to the service.
    ///
    /// - Parameters:
```

```
    ///   - cipClient: The `CognitoIdentityProviderClient` to use.
    ///   - userName: The user's username.
    ///   - clientId: The app client ID.
    ///   - userPoolId: The user pool to sign into.
    ///   - mfaCode: The 6-digit MFA code currently displayed by the user's
    ///     authenticator.
    ///   - session: The authentication session to continue processing.
    func adminRespondToAuthChallenge(cipClient: CognitoIdentityProviderClient,
userName: String,
                                     clientId: String, userPoolId: String,
mfaCode: String,
                                     session: String) async {
        print("=====> SOFTWARE_TOKEN_MFA challenge is generated...")

        var challengeResponsesOb: [String: String] = [:]
        challengeResponsesOb["USERNAME"] = userName
        challengeResponsesOb["SOFTWARE_TOKEN_MFA_CODE"] = mfaCode

        do {
            let output = try await cipClient.adminRespondToAuthChallenge(
                input: AdminRespondToAuthChallengeInput(
                    challengeName:
CognitoIdentityProviderClientTypes.ChallengeNameType.softwareTokenMfa,
                    challengeResponses: challengeResponsesOb,
                    clientId: clientId,
                    session: session,
                    userPoolId: userPoolId
                )
            )

            guard let authenticationResult = output.authenticationResult else {
                print("*** Unable to get authentication result.")
                return
            }

            print("=====> Authentication result (JWTs are redacted):")
            print(authenticationResult)
        } catch _ as SoftwareTokenMFANotFoundException {
            print("*** The specified user pool isn't configured for MFA.")
            return
        } catch _ as CodeMismatchException {
            print("*** The specified MFA code doesn't match the expected value.")
            return
        } catch _ as UserNotFoundException {
```

```
            print("*** The specified username, \(userName), doesn't exist.")
            return
        } catch _ as UserNotConfirmedException {
            print("*** The user \(userName) has not been confirmed.")
            return
        } catch let error as NotAuthorizedException {
            print("*** Unauthorized access. Reason: \(error.properties.message ??
"<unknown>")")
        } catch {
            print("*** Error responding to the MFA challenge.")
            return
        }
    }

    /// Called by ``main()`` to run the bulk of the example.
    func runAsync() async throws {
        let config = try await
CognitoIdentityProviderClient.CognitoIdentityProviderClientConfiguration(region:
region)
        let cipClient = CognitoIdentityProviderClient(config: config)

        print("""
              This example collects information about a user, then creates that
user in the
              specified user pool. Then, it enables Multi-Factor Authentication
(MFA) for that
              user by associating an authenticator application (such as Google
Authenticator
              or a password manager that supports TOTP). Then, the user uses a
code from their
              authenticator application to sign in.

              """)

        let userName = stringRequest("Please enter a new username: ")
        let password = stringRequest("Enter a password: ")
        let email = stringRequest("Enter your email address: ", minLength: 5)

        // Submit the sign-up request to AWS.

        print("==> Signing up user \(userName)...")
        if await signUp(cipClient: cipClient, clientId: clientId,
                        userName: userName, password: password,
                        email: email) == false {
```

```
            return
        }

        // Check the user's status. This time, it should come back "unconfirmed".

        print("==> Getting the status of user \(userName) from the user pool
(should be 'unconfirmed')...")
        if await adminGetUser(cipClient: cipClient, userName: userName,
userPoolId: poolId) == false {
            return
        }

        // Ask the user if they want a replacement code sent, such as if the
        // code hasn't arrived yet. If the user responds with a "yes," send a
        // new code.

        if yesNoRequest("==> A confirmation code was sent to \(userName). Would
you like to send a new code (Y/N)? ") {
            print("==> Sending a new confirmation code...")
            if await resendConfirmationCode(cipClient: cipClient, clientId:
clientId, userName: userName) == false {
                return
            }
        }

        // Ask the user to enter the confirmation code, then send it to Amazon
        // Cognito to verify it.

        let code = stringRequest("==> Enter the confirmation code sent to
\(userName): ")
        if await confirmSignUp(cipClient: cipClient, clientId: clientId,
userName: userName, code: code) == false {
            // The code didn't match. Your application may wish to offer to
            // re-send the confirmation code here and try again.
            return
        }

        // Check the user's status again. This time it should come back
        // "confirmed".

        print("==> Rechecking status of user \(userName) in the user pool (should
be 'confirmed')...")
        if await adminGetUser(cipClient: cipClient, userName: userName,
userPoolId: poolId) == false {
```

```
            return
        }
        // Check the challenge mode. Here, it should be "mfaSetup", indicating
        // that the user needs to add MFA before using it. This returns a
        // session that can be used to register MFA, or nil if an error occurs.

        let authSession = await initiateAuth(cipClient: cipClient, clientId:
clientId,
                                             userName: userName, password:
password,
                                             userPoolId: poolId)
        if authSession == nil {
            return
        }

        // Ask Cognito for an MFA secret token that the user should enter into
        // their authenticator software (such as Google Authenticator) or
        // password manager to configure it for this user account. This
        // returns a new session that should be used for the new stage of the
        // authentication process.

        let newSession = await getSecretForAppMFA(cipClient: cipClient,
authSession: authSession)
        if newSession == nil {
            return
        }

        // Ask the user to enter the current 6-digit code displayed by their
        // authenticator. Then verify that it matches the value expected for
        // the session.

        let mfaCode1 = stringRequest("==> Enter the 6-digit code displayed in
your authenticator: ",
                                     minLength: 6)
        await verifyTOTP(cipClient: cipClient, session: newSession, mfaCode:
mfaCode1)

        // Ask the user to authenticate now that the authenticator has been
        // configured. This creates a new session using the user's username
        // and password as already entered.

        print("\nNow starting the sign-in process for user \(userName)...\n")
```

```
        let session2 = await initiateAuth(cipClient: cipClient, clientId:
 clientId,
                                          userName: userName, password: password,
 userPoolId: poolId)
        guard let session2 else {
            return
        }

        // Now that we have a new auth session, `session2`, ask the user for a
        // new 6-digit code from their authenticator, and send it to the auth
        // session.

        let mfaCode2 = stringRequest("==> Wait for your authenticator to show a
 new 6-digit code, then enter it: ",
                                     minLength: 6)
        await adminRespondToAuthChallenge(cipClient: cipClient, userName:
 userName,
                                          clientId: clientId, userPoolId: poolId,
                                          mfaCode: mfaCode2, session: session2)
    }
}

/// The program's asynchronous entry point.
@main
struct Main {
    static func main() async {
        let args = Array(CommandLine.arguments.dropFirst())

        do {
            let command = try ExampleCommand.parse(args)
            try await command.runAsync()
        } catch {
            ExampleCommand.exit(withError: error)
        }
    }
}
```

- Consulte detalhes da API nos tópicos a seguir na Referência de API do AWS SDK para Swift.
  - AdminGetUser
  - AdminInitiateAuth

- AdminRespondToAuthChallenge
- AssociateSoftwareToken
- ConfirmDevice
- ConfirmSignUp
- InitiateAuth
- ListUsers
- ResendConfirmationCode
- RespondToAuthChallenge
- SignUp
- VerifySoftwareToken

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Usar bancos de identidades e fluxos de identidades do Amazon Cognito

O exemplo de código a seguir mostra como criar uma aplicação de demonstração baseada na web que demonstra fluxos de identidades de bancos de identidades.

Python

SDK para Python (Boto3)

Mostra uma aplicação de demonstração baseada na web que demonstra os fluxos de autenticação dos bancos de identidades do Amazon Cognito, permitindo que os usuários explorem interativamente os fluxos de autenticação aprimorada e básica com vários provedores de identidade.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em GitHub.

Serviços usados neste exemplo

- Provedor de identidade do Amazon Cognito

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Grave dados de atividades personalizados com uma função Lambda após a autenticação do usuário do Amazon Cognito usando um SDK AWS

O exemplo de código a seguir mostra como gravar dados de atividade personalizados com uma função do Lambda depois da autenticação do usuário do Amazon Cognito.

- Use as funções de administrador para adicionar um usuário a um grupo de usuários.
- Configure um grupo de usuários para chamar uma função do Lambda para o acionador `PostAuthentication`.
- Faça login do novo usuário no Amazon Cognito.
- A função Lambda grava informações personalizadas em CloudWatch Logs e em uma tabela do DynamoDB.
- Obtenha e veja dados personalizados da tabela do DynamoDB e, em seguida, limpe os recursos.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

Execute um cenário interativo em um prompt de comando.

```
import (
 "context"
 "errors"
 "log"
 "strings"
 "user_pools_and_lambda_triggers/actions"

 "github.com/aws/aws-sdk-go-v2/aws"
```

```
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// ActivityLog separates the steps of this scenario into individual functions so
 that
// they are simpler to read and understand.
type ActivityLog struct {
 helper       IScenarioHelper
 questioner   demotools.IQuestioner
 resources    Resources
 cognitoActor *actions.CognitoActions
}

// NewActivityLog constructs a new activity log runner.
func NewActivityLog(sdkConfig aws.Config, questioner demotools.IQuestioner,
 helper IScenarioHelper) ActivityLog {
 scenario := ActivityLog{
  helper:       helper,
  questioner:   questioner,
  resources:    Resources{},
  cognitoActor: &actions.CognitoActions{CognitoClient:
 cognitoidentityprovider.NewFromConfig(sdkConfig)},
 }
 scenario.resources.init(scenario.cognitoActor, questioner)
 return scenario
}

// AddUserToPool selects a user from the known users table and uses administrator
 credentials to add the user to the user pool.
func (runner *ActivityLog) AddUserToPool(ctx context.Context, userPoolId string,
 tableName string) (string, string) {
 log.Println("To facilitate this example, let's add a user to the user pool using
 administrator privileges.")
 users, err := runner.helper.GetKnownUsers(ctx, tableName)
 if err != nil {
  panic(err)
 }
 user := users.Users[0]
 log.Printf("Adding known user %v to the user pool.\n", user.UserName)
 err = runner.cognitoActor.AdminCreateUser(ctx, userPoolId, user.UserName,
 user.UserEmail)
 if err != nil {
```

```
  panic(err)
 }
 pwSet := false
 password := runner.questioner.AskPassword("\nEnter a password that has at least
 eight characters, uppercase, lowercase, numbers and symbols.\n"+
  "(the password will not display as you type):", 8)
 for !pwSet {
  log.Printf("\nSetting password for user '%v'.\n", user.UserName)
  err = runner.cognitoActor.AdminSetUserPassword(ctx, userPoolId, user.UserName,
 password)
  if err != nil {
   var invalidPassword *types.InvalidPasswordException
   if errors.As(err, &invalidPassword) {
    password = runner.questioner.AskPassword("\nEnter another password:", 8)
   } else {
    panic(err)
   }
  } else {
   pwSet = true
  }
 }

 log.Println(strings.Repeat("-", 88))

 return user.UserName, password
}

// AddActivityLogTrigger adds a Lambda handler as an invocation target for the
 PostAuthentication trigger.
func (runner *ActivityLog) AddActivityLogTrigger(ctx context.Context, userPoolId
 string, activityLogArn string) {
 log.Println("Let's add a Lambda function to handle the PostAuthentication
 trigger from Cognito.\n" +
  "This trigger happens after a user is authenticated, and lets your function
 take action, such as logging\n" +
  "the outcome.")
 err := runner.cognitoActor.UpdateTriggers(
  ctx, userPoolId,
  actions.TriggerInfo{Trigger: actions.PostAuthentication, HandlerArn:
 aws.String(activityLogArn)})
 if err != nil {
  panic(err)
 }
```

```
 runner.resources.triggers = append(runner.resources.triggers,
 actions.PostAuthentication)
 log.Printf("Lambda function %v added to user pool %v to handle
 PostAuthentication Cognito trigger.\n",
  activityLogArn, userPoolId)

 log.Println(strings.Repeat("-", 88))
}

// SignInUser signs in as the specified user.
func (runner *ActivityLog) SignInUser(ctx context.Context, clientId string,
 userName string, password string) {
 log.Printf("Now we'll sign in user %v and check the results in the logs and the
 DynamoDB table.", userName)
 runner.questioner.Ask("Press Enter when you're ready.")
 authResult, err := runner.cognitoActor.SignIn(ctx, clientId, userName, password)
 if err != nil {
  panic(err)
 }
 log.Println("Sign in successful.",
  "The PostAuthentication Lambda handler writes custom information to CloudWatch
 Logs.")

 runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
 *authResult.AccessToken)
}

// GetKnownUserLastLogin gets the login info for a user from the Amazon DynamoDB
 table and displays it.
func (runner *ActivityLog) GetKnownUserLastLogin(ctx context.Context, tableName
 string, userName string) {
 log.Println("The PostAuthentication handler also writes login data to the
 DynamoDB table.")
 runner.questioner.Ask("Press Enter when you're ready to continue.")
 users, err := runner.helper.GetKnownUsers(ctx, tableName)
 if err != nil {
  panic(err)
 }
 for _, user := range users.Users {
  if user.UserName == userName {
   log.Println("The last login info for the user in the known users table is:")
   log.Printf("\t%+v", *user.LastLogin)
  }
 }
```

```
 log.Println(strings.Repeat("-", 88))
}

// Run runs the scenario.
func (runner *ActivityLog) Run(ctx context.Context, stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup(ctx)
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(ctx, stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]
 runner.helper.PopulateUserTable(ctx, stackOutputs["TableName"])
 userName, password := runner.AddUserToPool(ctx, stackOutputs["UserPoolId"],
 stackOutputs["TableName"])

 runner.AddActivityLogTrigger(ctx, stackOutputs["UserPoolId"],
 stackOutputs["ActivityLogFunctionArn"])
 runner.SignInUser(ctx, stackOutputs["UserPoolClientId"], userName, password)
 runner.helper.ListRecentLogEvents(ctx, stackOutputs["ActivityLogFunction"])
 runner.GetKnownUserLastLogin(ctx, stackOutputs["TableName"], userName)

 runner.resources.Cleanup(ctx)

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Aborde o acionador `PostAuthentication` com uma função do Lambda.

```
import (
 "context"
 "fmt"
 "log"
 "os"
 "time"

 "github.com/aws/aws-lambda-go/events"
 "github.com/aws/aws-lambda-go/lambda"
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 dynamodbtypes "github.com/aws/aws-sdk-go-v2/service/dynamodb/types"
)

const TABLE_NAME = "TABLE_NAME"

// LoginInfo defines structured login data that can be marshalled to a DynamoDB
 format.
type LoginInfo struct {
 UserPoolId string `dynamodbav:"UserPoolId"`
 ClientId   string `dynamodbav:"ClientId"`
 Time       string `dynamodbav:"Time"`
}

// UserInfo defines structured user data that can be marshalled to a DynamoDB
 format.
type UserInfo struct {
 UserName  string    `dynamodbav:"UserName"`
 UserEmail string    `dynamodbav:"UserEmail"`
 LastLogin LoginInfo `dynamodbav:"LastLogin"`
}

// GetKey marshals the user email value to a DynamoDB key format.
func (user UserInfo) GetKey() map[string]dynamodbtypes.AttributeValue {
 userEmail, err := attributevalue.Marshal(user.UserEmail)
 if err != nil {
  panic(err)
 }
 return map[string]dynamodbtypes.AttributeValue{"UserEmail": userEmail}
}
```

```
type handler struct {
 dynamoClient *dynamodb.Client
}

// HandleRequest handles the PostAuthentication event by writing custom data to
 the logs and
// to an Amazon DynamoDB table.
func (h *handler) HandleRequest(ctx context.Context,
 event events.CognitoEventUserPoolsPostAuthentication)
 (events.CognitoEventUserPoolsPostAuthentication, error) {
 log.Printf("Received post authentication trigger from %v for user '%v'",
 event.TriggerSource, event.UserName)
 tableName := os.Getenv(TABLE_NAME)
 user := UserInfo{
  UserName:  event.UserName,
  UserEmail: event.Request.UserAttributes["email"],
  LastLogin: LoginInfo{
   UserPoolId: event.UserPoolID,
   ClientId:   event.CallerContext.ClientID,
   Time:       time.Now().Format(time.UnixDate),
  },
 }
 // Write to CloudWatch Logs.
 fmt.Printf("%#v", user)

 // Also write to an external system. This examples uses DynamoDB to demonstrate.
 userMap, err := attributevalue.MarshalMap(user)
 if err != nil {
  log.Printf("Couldn't marshal to DynamoDB map. Here's why: %v\n", err)
 } else if len(userMap) == 0 {
  log.Printf("User info marshaled to an empty map.")
 } else {
  _, err := h.dynamoClient.PutItem(ctx, &dynamodb.PutItemInput{
   Item:      userMap,
   TableName: aws.String(tableName),
  })
  if err != nil {
   log.Printf("Couldn't write to DynamoDB. Here's why: %v\n", err)
  } else {
   log.Printf("Wrote user info to DynamoDB table %v.\n", tableName)
  }
 }
```

```
 return event, nil
}

func main() {
 ctx := context.Background()
 sdkConfig, err := config.LoadDefaultConfig(ctx)
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Crie uma struct que realize tarefas comuns.

```
import (
 "context"
 "log"
 "strings"
 "time"
 "user_pools_and_lambda_triggers/actions"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudformation"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
 GetStackOutputs(ctx context.Context, stackName string) (actions.StackOutputs,
 error)
 PopulateUserTable(ctx context.Context, tableName string)
 GetKnownUsers(ctx context.Context, tableName string) (actions.UserList, error)
 AddKnownUser(ctx context.Context, tableName string, user actions.User)
```

```
 ListRecentLogEvents(ctx context.Context, functionName string)
}

// ScenarioHelper contains AWS wrapper structs used by the workflows in this
 example.
type ScenarioHelper struct {
 questioner  demotools.IQuestioner
 dynamoActor *actions.DynamoActions
 cfnActor    *actions.CloudFormationActions
 cwlActor    *actions.CloudWatchLogsActions
 isTestRun   bool
}

// NewScenarioHelper constructs a new scenario helper.
func NewScenarioHelper(sdkConfig aws.Config, questioner demotools.IQuestioner)
 ScenarioHelper {
 scenario := ScenarioHelper{
  questioner:  questioner,
  dynamoActor: &actions.DynamoActions{DynamoClient:
 dynamodb.NewFromConfig(sdkConfig)},
  cfnActor:    &actions.CloudFormationActions{CfnClient:
 cloudformation.NewFromConfig(sdkConfig)},
  cwlActor:    &actions.CloudWatchLogsActions{CwlClient:
 cloudwatchlogs.NewFromConfig(sdkConfig)},
 }
 return scenario
}

// Pause waits for the specified number of seconds.
func (helper ScenarioHelper) Pause(secs int) {
 if !helper.isTestRun {
  time.Sleep(time.Duration(secs) * time.Second)
 }
}

// GetStackOutputs gets the outputs from the specified CloudFormation stack in a
 structured format.
func (helper ScenarioHelper) GetStackOutputs(ctx context.Context, stackName
 string) (actions.StackOutputs, error) {
 return helper.cfnActor.GetOutputs(ctx, stackName), nil
}

// PopulateUserTable fills the known user table with example data.
```

```
func (helper ScenarioHelper) PopulateUserTable(ctx context.Context, tableName
 string) {
 log.Printf("First, let's add some users to the DynamoDB %v table we'll use for
 this example.\n", tableName)
 err := helper.dynamoActor.PopulateTable(ctx, tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(ctx context.Context, tableName string)
 (actions.UserList, error) {
 knownUsers, err := helper.dynamoActor.Scan(ctx, tableName)
 if err != nil {
  log.Printf("Couldn't get known users from table %v. Here's why: %v\n",
 tableName, err)
 }
 return knownUsers, err
}

// AddKnownUser adds a user to the known users table.
func (helper ScenarioHelper) AddKnownUser(ctx context.Context, tableName string,
 user actions.User) {
 log.Printf("Adding user '%v' with email '%v' to the DynamoDB known users
 table...\n",
  user.UserName, user.UserEmail)
 err := helper.dynamoActor.AddUser(ctx, tableName, user)
 if err != nil {
  panic(err)
 }
}

// ListRecentLogEvents gets the most recent log stream and events for the
 specified Lambda function and displays them.
func (helper ScenarioHelper) ListRecentLogEvents(ctx context.Context,
 functionName string) {
 log.Println("Waiting a few seconds to let Lambda write to CloudWatch Logs...")
 helper.Pause(10)
 log.Println("Okay, let's check the logs to find what's happened recently with
 your Lambda function.")
 logStream, err := helper.cwlActor.GetLatestLogStream(ctx, functionName)
 if err != nil {
```

```
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
 events, err := helper.cwlActor.GetLogEvents(ctx, functionName,
 *logStream.LogStreamName, 10)
 if err != nil {
  panic(err)
 }
 for _, event := range events {
  log.Printf("\t%v", *event.Message)
 }
 log.Println(strings.Repeat("-", 88))
}
```

Crie uma struct que encapsule ações do Amazon Cognito.

```
import (
 "context"
 "errors"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider"
 "github.com/aws/aws-sdk-go-v2/service/cognitoidentityprovider/types"
)

type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// Trigger and TriggerInfo define typed data for updating an Amazon Cognito
 trigger.
type Trigger int

const (
 PreSignUp Trigger = iota
 UserMigration
```

```
 PostAuthentication
)

type TriggerInfo struct {
 Trigger    Trigger
 HandlerArn *string
}

// UpdateTriggers adds or removes Lambda triggers for a user pool. When a trigger
 is specified with a `nil` value,
// it is removed from the user pool.
func (actor CognitoActions) UpdateTriggers(ctx context.Context, userPoolId
 string, triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(ctx,
 &cognitoidentityprovider.DescribeUserPoolInput{
  UserPoolId: aws.String(userPoolId),
 })
 if err != nil {
  log.Printf("Couldn't get info about user pool %v. Here's why: %v\n",
 userPoolId, err)
  return err
 }
 lambdaConfig := output.UserPool.LambdaConfig
 for _, trigger := range triggers {
  switch trigger.Trigger {
  case PreSignUp:
   lambdaConfig.PreSignUp = trigger.HandlerArn
  case UserMigration:
   lambdaConfig.UserMigration = trigger.HandlerArn
  case PostAuthentication:
   lambdaConfig.PostAuthentication = trigger.HandlerArn
  }
 }
 _, err = actor.CognitoClient.UpdateUserPool(ctx,
 &cognitoidentityprovider.UpdateUserPoolInput{
  UserPoolId:   aws.String(userPoolId),
  LambdaConfig: lambdaConfig,
 })
 if err != nil {
  log.Printf("Couldn't update user pool %v. Here's why: %v\n", userPoolId, err)
 }
 return err
}
```

```
// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(ctx context.Context, clientId string, userName
 string, password string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(ctx,
 &cognitoidentityprovider.SignUpInput{
  ClientId: aws.String(clientId),
  Password: aws.String(password),
  Username: aws.String(userName),
  UserAttributes: []types.AttributeType{
   {Name: aws.String("email"), Value: aws.String(userEmail)},
  },
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't sign up user %v. Here's why: %v\n", userName, err)
  }
 } else {
  confirmed = output.UserConfirmed
 }
 return confirmed, err
}



// SignIn signs in a user to Amazon Cognito using a username and password
 authentication flow.
func (actor CognitoActions) SignIn(ctx context.Context, clientId string, userName
 string, password string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(ctx,
 &cognitoidentityprovider.InitiateAuthInput{
  AuthFlow:       "USER_PASSWORD_AUTH",
  ClientId:       aws.String(clientId),
  AuthParameters: map[string]string{"USERNAME": userName, "PASSWORD": password},
 })
 if err != nil {
  var resetRequired *types.PasswordResetRequiredException
  if errors.As(err, &resetRequired) {
```

```
   log.Println(*resetRequired.Message)
  } else {
   log.Printf("Couldn't sign in user %v. Here's why: %v\n", userName, err)
  }
 } else {
  authResult = output.AuthenticationResult
 }
 return authResult, err
}



// ForgotPassword starts a password recovery flow for a user. This flow typically
 sends a confirmation code
// to the user's configured notification destination, such as email.
func (actor CognitoActions) ForgotPassword(ctx context.Context, clientId string,
 userName string) (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(ctx,
 &cognitoidentityprovider.ForgotPasswordInput{
  ClientId: aws.String(clientId),
  Username: aws.String(userName),
 })
 if err != nil {
  log.Printf("Couldn't start password reset for user '%v'. Here;s why: %v\n",
 userName, err)
 }
 return output.CodeDeliveryDetails, err
}



// ConfirmForgotPassword confirms a user with a confirmation code and a new
 password.
func (actor CognitoActions) ConfirmForgotPassword(ctx context.Context, clientId
 string, code string, userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(ctx,
 &cognitoidentityprovider.ConfirmForgotPasswordInput{
  ClientId:         aws.String(clientId),
  ConfirmationCode: aws.String(code),
  Password:         aws.String(password),
  Username:         aws.String(userName),
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
```

```
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't confirm user %v. Here's why: %v", userName, err)
  }
 }
 return err
}



// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(ctx context.Context, userAccessToken
 string) error {
 _, err := actor.CognitoClient.DeleteUser(ctx,
 &cognitoidentityprovider.DeleteUserInput{
  AccessToken: aws.String(userAccessToken),
 })
 if err != nil {
  log.Printf("Couldn't delete user. Here's why: %v\n", err)
 }
 return err
}



// AdminCreateUser uses administrator credentials to add a user to a user pool.
 This method leaves the user
// in a state that requires they enter a new password next time they sign in.
func (actor CognitoActions) AdminCreateUser(ctx context.Context, userPoolId
 string, userName string, userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(ctx,
 &cognitoidentityprovider.AdminCreateUserInput{
  UserPoolId:     aws.String(userPoolId),
  Username:       aws.String(userName),
  MessageAction:  types.MessageActionTypeSuppress,
  UserAttributes: []types.AttributeType{{Name: aws.String("email"), Value:
 aws.String(userEmail)}},
 })
 if err != nil {
  var userExists *types.UsernameExistsException
  if errors.As(err, &userExists) {
   log.Printf("User %v already exists in the user pool.", userName)
   err = nil
```

```
  } else {
   log.Printf("Couldn't create user %v. Here's why: %v\n", userName, err)
  }
 }
 return err
}



// AdminSetUserPassword uses administrator credentials to set a password for a
 user without requiring a
// temporary password.
func (actor CognitoActions) AdminSetUserPassword(ctx context.Context, userPoolId
 string, userName string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(ctx,
 &cognitoidentityprovider.AdminSetUserPasswordInput{
  Password:   aws.String(password),
  UserPoolId: aws.String(userPoolId),
  Username:   aws.String(userName),
  Permanent:  true,
 })
 if err != nil {
  var invalidPassword *types.InvalidPasswordException
  if errors.As(err, &invalidPassword) {
   log.Println(*invalidPassword.Message)
  } else {
   log.Printf("Couldn't set password for user %v. Here's why: %v\n", userName,
 err)
  }
 }
 return err
}
```

Crie uma struct que encapsule ações do DynamoDB.

```
import (
 "context"
 "fmt"
 "log"
```

```
 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/feature/dynamodb/attributevalue"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb"
 "github.com/aws/aws-sdk-go-v2/service/dynamodb/types"
)

// DynamoActions encapsulates the Amazon Simple Notification Service (Amazon SNS)
 actions
// used in the examples.
type DynamoActions struct {
 DynamoClient *dynamodb.Client
}

// User defines structured user data.
type User struct {
 UserName  string
 UserEmail string
 LastLogin *LoginInfo `dynamodbav:",omitempty"`
}

// LoginInfo defines structured custom login data.
type LoginInfo struct {
 UserPoolId string
 ClientId   string
 Time       string
}

// UserList defines a list of users.
type UserList struct {
 Users []User
}

// UserNameList returns the usernames contained in a UserList as a list of
 strings.
func (users *UserList) UserNameList() []string {
 names := make([]string, len(users.Users))
 for i := 0; i < len(users.Users); i++ {
  names[i] = users.Users[i].UserName
 }
 return names
}

// PopulateTable adds a set of test users to the table.
```

```
func (actor DynamoActions) PopulateTable(ctx context.Context, tableName string)
 error {
 var err error
 var item map[string]types.AttributeValue
 var writeReqs []types.WriteRequest
 for i := 1; i < 4; i++ {
  item, err = attributevalue.MarshalMap(User{UserName: fmt.Sprintf("test_user_
%v", i), UserEmail: fmt.Sprintf("test_email_%v@example.com", i)})
  if err != nil {
   log.Printf("Couldn't marshall user into DynamoDB format. Here's why: %v\n",
 err)
   return err
  }
  writeReqs = append(writeReqs, types.WriteRequest{PutRequest:
 &types.PutRequest{Item: item}})
 }
 _, err = actor.DynamoClient.BatchWriteItem(ctx, &dynamodb.BatchWriteItemInput{
  RequestItems: map[string][]types.WriteRequest{tableName: writeReqs},
 })
 if err != nil {
  log.Printf("Couldn't populate table %v with users. Here's why: %v\n",
 tableName, err)
 }
 return err
}

// Scan scans the table for all items.
func (actor DynamoActions) Scan(ctx context.Context, tableName string) (UserList,
 error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(ctx, &dynamodb.ScanInput{
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't scan table %v for items. Here's why: %v\n", tableName,
 err)
 } else {
  err = attributevalue.UnmarshalListOfMaps(output.Items, &userList.Users)
  if err != nil {
   log.Printf("Couldn't unmarshal items into users. Here's why: %v\n", err)
  }
 }
 return userList, err
}
```

```
// AddUser adds a user item to a table.
func (actor DynamoActions) AddUser(ctx context.Context, tableName string, user
 User) error {
 userItem, err := attributevalue.MarshalMap(user)
 if err != nil {
  log.Printf("Couldn't marshall user to item. Here's why: %v\n", err)
 }
 _, err = actor.DynamoClient.PutItem(ctx, &dynamodb.PutItemInput{
  Item:      userItem,
  TableName: aws.String(tableName),
 })
 if err != nil {
  log.Printf("Couldn't put item in table %v. Here's why: %v", tableName, err)
 }
 return err
}
```

Crie uma estrutura que envolva as ações do CloudWatch Logs.

```
import (
 "context"
 "fmt"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs"
 "github.com/aws/aws-sdk-go-v2/service/cloudwatchlogs/types"
)

type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(ctx context.Context,
 functionName string) (types.LogStream, error) {
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
```

```
 output, err := actor.CwlClient.DescribeLogStreams(ctx,
 &cloudwatchlogs.DescribeLogStreamsInput{
  Descending:   aws.Bool(true),
  Limit:        aws.Int32(1),
  LogGroupName: aws.String(logGroupName),
  OrderBy:      types.OrderByLastEventTime,
 })
 if err != nil {
  log.Printf("Couldn't get log streams for log group %v. Here's why: %v\n",
 logGroupName, err)
 } else {
  logStream = output.LogStreams[0]
 }
 return logStream, err
}

// GetLogEvents gets the most recent eventCount events from the specified log
 stream.
func (actor CloudWatchLogsActions) GetLogEvents(ctx context.Context, functionName
 string, logStreamName string, eventCount int32) (
 []types.OutputLogEvent, error) {
 var events []types.OutputLogEvent
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.GetLogEvents(ctx,
 &cloudwatchlogs.GetLogEventsInput{
  LogStreamName: aws.String(logStreamName),
  Limit:         aws.Int32(eventCount),
  LogGroupName:  aws.String(logGroupName),
 })
 if err != nil {
  log.Printf("Couldn't get log event for log stream %v. Here's why: %v\n",
 logStreamName, err)
 } else {
  events = output.Events
 }
 return events, err
}
```

Crie uma estrutura que envolva as ações. CloudFormation

```
import (
 "context"
 "log"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/service/cloudformation"
)

// StackOutputs defines a map of outputs from a specific stack.
type StackOutputs map[string]string

type CloudFormationActions struct {
 CfnClient *cloudformation.Client
}

// GetOutputs gets the outputs from a CloudFormation stack and puts them into a
 structured format.
func (actor CloudFormationActions) GetOutputs(ctx context.Context, stackName
 string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(ctx,
 &cloudformation.DescribeStacksInput{
  StackName: aws.String(stackName),
 })
 if err != nil || len(output.Stacks) == 0 {
  log.Panicf("Couldn't find a CloudFormation stack named %v. Here's why: %v\n",
 stackName, err)
 }
 stackOutputs := StackOutputs{}
 for _, out := range output.Stacks[0].Outputs {
  stackOutputs[*out.OutputKey] = *out.OutputValue
 }
 return stackOutputs
}
```

Limpe recursos.

```
import (
 "context"
 "log"
 "user_pools_and_lambda_triggers/actions"
```

```
 "github.com/awsdocs/aws-doc-sdk-examples/gov2/demotools"
)

// Resources keeps track of AWS resources created during an example and handles
// cleanup when the example finishes.
type Resources struct {
 userPoolId       string
 userAccessTokens []string
 triggers         []actions.Trigger

 cognitoActor *actions.CognitoActions
 questioner   demotools.IQuestioner
}

func (resources *Resources) init(cognitoActor *actions.CognitoActions, questioner
 demotools.IQuestioner) {
 resources.userAccessTokens = []string{}
 resources.triggers = []actions.Trigger{}
 resources.cognitoActor = cognitoActor
 resources.questioner = questioner
}

// Cleanup deletes all AWS resources created during an example.
func (resources *Resources) Cleanup(ctx context.Context) {
 defer func() {
  if r := recover(); r != nil {
   log.Printf("Something went wrong during cleanup.\n%v\n", r)
   log.Println("Use the AWS Management Console to remove any remaining resources
 \n" +
    "that were created for this scenario.")
  }
 }()

 wantDelete := resources.questioner.AskBool("Do you want to remove all of the AWS
 resources that were created "+
  "during this demo (y/n)?", "y")
 if wantDelete {
  for _, accessToken := range resources.userAccessTokens {
   err := resources.cognitoActor.DeleteUser(ctx, accessToken)
   if err != nil {
    log.Println("Couldn't delete user during cleanup.")
    panic(err)
   }
```

```
   log.Println("Deleted user.")
  }
  triggerList := make([]actions.TriggerInfo, len(resources.triggers))
  for i := 0; i < len(resources.triggers); i++ {
   triggerList[i] = actions.TriggerInfo{Trigger: resources.triggers[i],
 HandlerArn: nil}
  }
  err := resources.cognitoActor.UpdateTriggers(ctx, resources.userPoolId,
 triggerList...)
  if err != nil {
   log.Println("Couldn't update Cognito triggers during cleanup.")
   panic(err)
  }
  log.Println("Removed Cognito triggers from user pool.")
 } else {
  log.Println("Be sure to remove resources when you're done with them to avoid
 unexpected charges!")
 }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para Go .
  - AdminCreateUser
  - AdminSetUserPassword
  - DeleteUser
  - InitiateAuth
  - UpdateUserPool

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Exemplos de código para o Amazon Cognito Sync usando AWS SDKs

Os exemplos de código a seguir mostram como usar o Amazon Cognito Sync com um kit de desenvolvimento AWS de software (SDK).

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar perfis de serviço individuais, você pode ver as ações no contexto em seus cenários relacionados.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Amazon Cognito Sync

- Exemplos básicos do Amazon Cognito Sync usando AWS SDKs
  - Ações para o Amazon Cognito Sync usando AWS SDKs
    - Use ListIdentityPoolUsage com um AWS SDK

Exemplos básicos do Amazon Cognito Sync usando AWS SDKs

Os exemplos de código a seguir mostram como usar os conceitos básicos do Amazon Cognito Sync com. AWS SDKs

Exemplos

- Ações para o Amazon Cognito Sync usando AWS SDKs
  - Use ListIdentityPoolUsage com um AWS SDK

Ações para o Amazon Cognito Sync usando AWS SDKs

Os exemplos de código a seguir demonstram como realizar ações individuais do Amazon Cognito Sync com. AWS SDKs Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções para configurar e executar o código.

Os exemplos a seguir incluem apenas as ações mais utilizadas. Para ter uma lista completa, consulte a Referência de API do Amazon Cognito Sync.

Exemplos

- Use ListIdentityPoolUsage com um AWS SDK

Use **`ListIdentityPoolUsage`** com um AWS SDK

O código de exemplo a seguir mostra como usar `ListIdentityPoolUsage`.

Rust

SDK para Rust

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no AWS Code Examples Repository.

```
async fn show_pools(client: &Client) -> Result<(), Error> {
    let response = client
        .list_identity_pool_usage()
        .max_results(10)
        .send()
        .await?;

    let pools = response.identity_pool_usages();
    println!("Identity pools:");

    for pool in pools {
        println!(
            "  Identity pool ID:    {}",
            pool.identity_pool_id().unwrap_or_default()
        );
        println!(
            "  Data storage:        {}",
            pool.data_storage().unwrap_or_default()
        );
        println!(
            "  Sync sessions count: {}",
            pool.sync_sessions_count().unwrap_or_default()
        );
        println!(
```

```
            "  Last modified:       {}",
            pool.last_modified_date().unwrap().to_chrono_utc()?
        );
        println!();
    }

    println!("Next token: {}", response.next_token().unwrap_or_default());

    Ok(())
}
```

- Para obter detalhes da API, consulte a ListIdentityPoolUsagereferência da API AWS SDK for Rust.

Para obter uma lista completa dos guias do desenvolvedor do AWS SDK e exemplos de código, consulteUsando esse serviço com um AWS SDK. Este tópico também inclui informações sobre como começar e detalhes sobre versões anteriores do SDK.

Práticas recomendadas de aplicações de vários locatários

Os grupos de usuários do Amazon Cognito operam com aplicações multilocatários que geram um volume de solicitações que devem permanecer dentro das cotas do Amazon Cognito. Para aumentar essa capacidade à medida que a base de clientes cresce, você pode comprar capacidade de cota adicional.

> **Note**
>
> As cotas do Amazon Cognito são aplicadas de forma periódica. Conta da AWS Região da AWS Essas cotas são compartilhadas entre todos os locatários da aplicação. Revise as cotas de serviço do Amazon Cognito e verifique se a cota atende ao volume esperado e ao número esperado de locatários na sua aplicação.

Esta seção descreve métodos que você pode implementar para separar inquilinos entre os recursos do Amazon Cognito dentro da mesma região e. Conta da AWS Você também pode dividir seus inquilinos em mais de uma Conta da AWS região e dar a cada um deles sua própria cota. Outras vantagens da multilocação em várias regiões incluem o nível mais alto possível de isolamento, o menor tempo de trânsito da rede para usuários distribuídos globalmente e a adesão aos modelos de distribuição existentes na organização.

A multilocação em uma única região também pode trazer vantagens para clientes e administradores.

A lista a seguir aborda algumas das vantagens da multilocação com recursos compartilhados.

Vantagens da multilocação

Diretório de usuários comum

A multilocação permite o uso de modelos em que os clientes têm contas em mais de uma aplicação. Você pode vincular identidades de provedores de terceiros em um único perfil consistente de grupo de usuários. Nos casos em que os perfis de usuário são exclusivos do locatário, qualquer estratégia de multilocação com um único grupo de usuários tem um ponto de entrada para a administração do usuário.

Segurança comum

Em um grupo de usuários compartilhado, você pode criar um único padrão de segurança e aplicar os mesmos padrões de proteção contra ameaças, autenticação multifator (MFA) e

AWS WAF a todos os locatários. Como uma ACL AWS WAF da web deve ser Região da AWS igual ao recurso ao qual você a associa, a multilocação oferece acesso compartilhado a um recurso complexo. Para manter uma configuração de segurança consistente em aplicações multirregionais do Amazon Cognito, você deve aplicar padrões operacionais que repliquem sua configuração entre recursos.

Personalização comum

Você pode personalizar grupos de usuários e grupos de identidades com AWS Lambda. A configuração de acionadores do Lambda em grupos de usuários e eventos do Amazon Cognito em bancos de identidades pode se tornar complexa. As funções do Lambda devem estar no mesmo grupo Região da AWS de usuários ou grupo de identidades. As funções compartilhadas do Lambda podem impor padrões para fluxos de autenticação personalizados, migração de usuários, geração de tokens e outras funções em uma região.

Mensagens comuns

O Amazon Simple Notification Service (Amazon SNS) exige configuração adicional em uma região para que você possa enviar mensagens SMS para os usuários. Você pode enviar mensagens de e-mail com identidades e domínios verificados do Amazon Simple Email Service (Amazon SES) que estejam contidos em uma região.

Com a multilocação, você pode compartilhar essa sobrecarga de configuração e manutenção entre todos os seus locatários. Como o Amazon SNS e o Amazon SES não estão disponíveis em todas as Regiões da AWS, dividir seus recursos entre regiões exige consideração adicional.

Ao usar provedores de mensagens personalizados, você obtém a personalização comum de uma única função do Lambda para gerenciar a entrega de mensagens.

O login gerenciado define um cookie de sessão no navegador para que ele reconheça um usuário já autenticado. Quando você autentica usuários locais em um grupo de usuários, o cookie de sessão os autentica para todos os clientes da aplicação no mesmo grupo de usuários. Um usuário local existe exclusivamente em seu diretório de grupo de usuários sem federação por meio de um IdP externo. O cookie da sessão é válido por uma hora. Não é possível alterar a duração do cookie da sessão.

Há duas maneiras de impedir o login em clientes de aplicações com um cookie de sessão de interface do usuário hospedado.

- Separe seus usuários em grupos de usuários por locatário.

- Substitua a interface do usuário hospedada pelo login da API de grupos de usuários do Amazon Cognito.

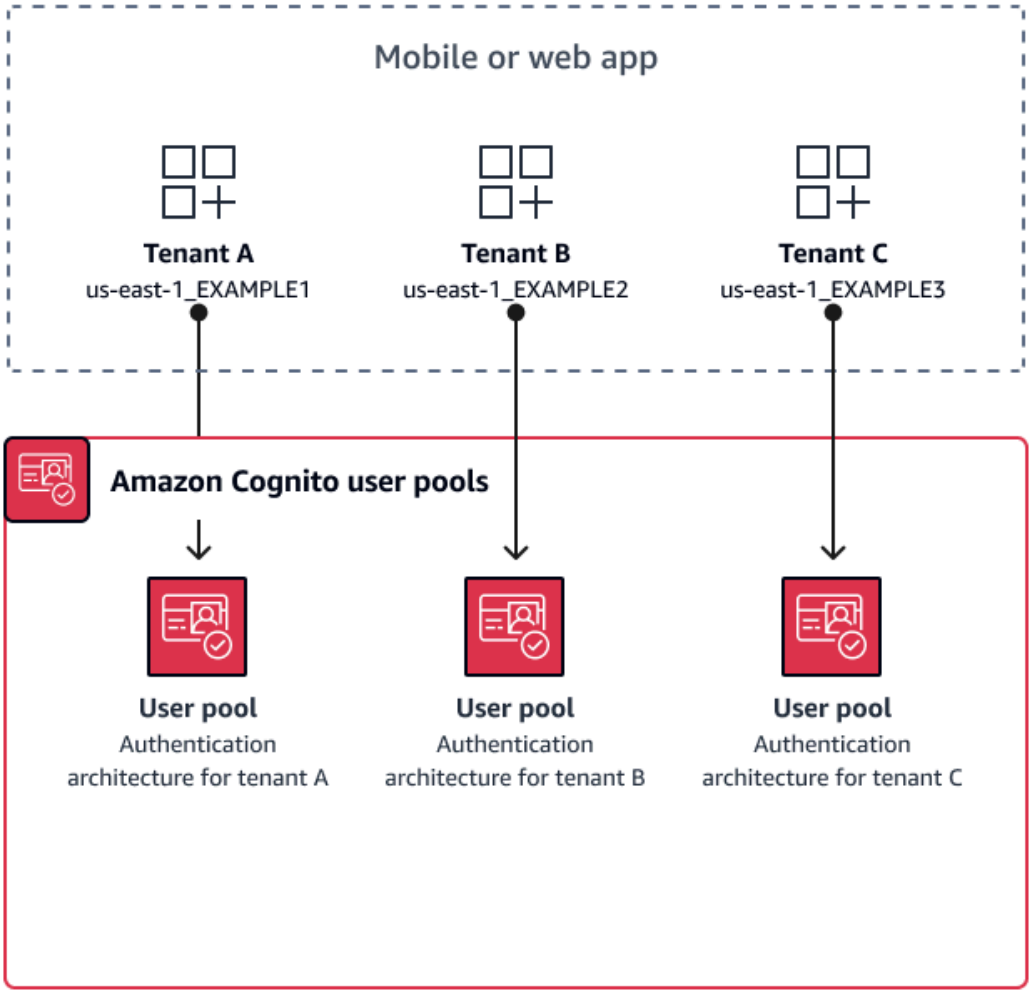
Tópicos

Práticas recomendadas de multilocação em grupos de usuários

Crie um grupo de usuários para cada locatário na sua aplicação. Essa abordagem fornece o máximo de isolamento para cada locatário. Você pode implementar configurações diferentes para cada locatário. O isolamento do inquilino por grupo de usuários oferece flexibilidade no user-to-tenant mapeamento. Você pode criar vários perfis para o mesmo usuário. No entanto, cada usuário deve se cadastrar individualmente para cada locatário ao qual tem acesso.

Com essa abordagem, você pode configurar a interface do usuário hospedada para cada locatário de forma independente e redirecionar os usuários para a instância específica do locatário de sua aplicação. Você também pode usar essa abordagem para facilitar a integração com serviços de backend, como o Amazon API Gateway.

O diagrama a seguir mostra cada locatário com um grupo de usuários dedicado.



Quando implementar a multilocação de grupos de usuários

Quando o isolamento e a personalização são suas principais preocupações. O relacionamento entre usuários e locatários pode ser complexo em uma arquitetura com vários grupos de usuários. Por exemplo, imagine que você tem dois locatários educacionais. O mesmo usuário pode ser um aluno com acesso limitado em uma aplicação e um professor com um alto nível de permissões em outra. Você pode precisar de MFA em uma aplicação, mas não em outra, ou ter uma política de senha diferente. Como os usuários locais podem fazer login em vários clientes da aplicação em grupos de usuários com login gerenciado, a multilocação do grupo de usuários também é ideal quando você deseja que mais de um locatário faça login com o login gerenciado.

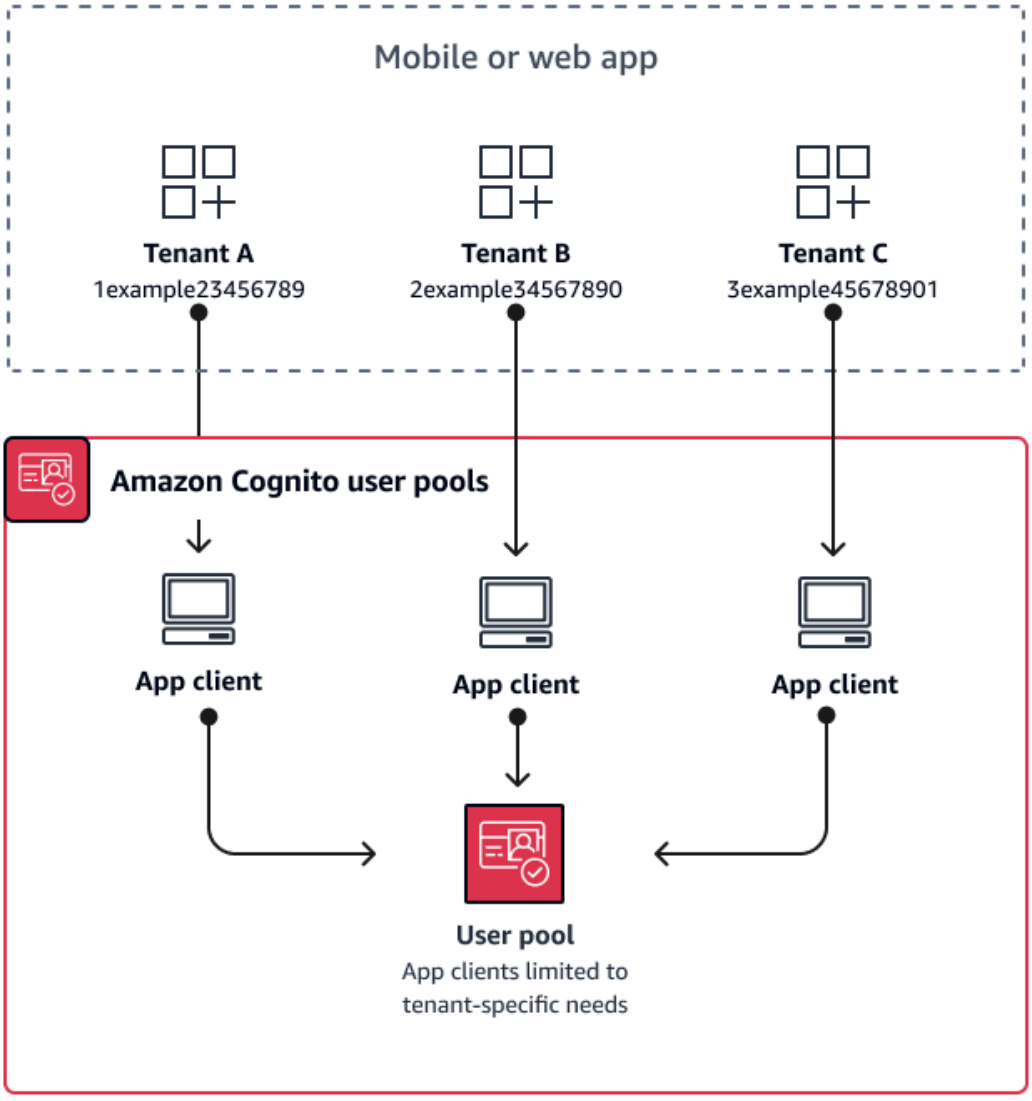
Nível de esforço

O esforço de desenvolvimento e operação para usar essa abordagem é alto. Para gerar resultados consistentes e previsíveis para sua família de aplicações, integre os recursos do Amazon Cognito às suas ferramentas de automação e mantenha as linhas de base à medida que sua arquitetura de autenticação se torna mais complexa. Quando quiser criar um único ponto de partida para as aplicações, crie os elementos da interface do usuário (UI) para capturar a decisão inicial que direciona os usuários para o recurso correto.

Práticas recomendadas de multilocatários em clientes de aplicações

Crie um cliente de aplicação para cada locatários na aplicação. Com a multilocação entre aplicações e clientes, você pode atribuir qualquer usuário a clientes de aplicações vinculados ao locatário e reter um único perfil de usuário. Como você pode atribuir qualquer um ou todos os provedores de identidade (IdPs) em seu grupo de usuários a um cliente de aplicativo, um cliente de aplicativo inquilino pode permitir o login com um IdP específico do inquilino. Quando os usuários existem em vários inquilinos, você pode vincular seus perfis a vários IdPs para uma experiência de usuário consistente.

O diagrama a seguir mostra cada locatário com um cliente de aplicação dedicado em um grupo de usuários compartilhado.



Quando implementar a multilocação entre aplicações e clientes

Quando você pode escolher uma configuração universal para configurações no nível do grupo de usuários, como acionadores do Lambda, política de senha e métodos de conteúdo e entrega de mensagens de e-mail e SMS. Como os usuários em um grupo de usuários compartilhado podem fazer login em qualquer cliente de aplicativo, a multilocação aplicativo-cliente é ideal para fazer login com ou com a API de grupos de usuários do app-client-specific IdPs Amazon Cognito. A multilocação entre aplicativos e clientes também é adequada para one-to-many ambientes em que você deseja permitir que os usuários façam a transição entre vários aplicativos.

Nível de esforço

A multilocação entre aplicações e clientes exige um esforço moderado. Um grande desafio da multilocação entre aplicações e clientes é a capacidade dos locatários de apresentar um cookie de interface do usuário hospedada e alternar entre aplicações. Em uma arquitetura de multilocação entre aplicações e clientes, evite fazer login na interface do usuário hospedada quando o isolamento for necessário. Você pode distribuir a aplicação móvel ou links para sua aplicação web com a lógica de cliente de aplicação incorporada, ou você pode criar elementos de interface de usuário iniciais que determinam a locação dos usuários. O nível de esforço é menor, porque você não precisa padronizar e manter a configuração em vários grupos de usuários e bancos de identidades.

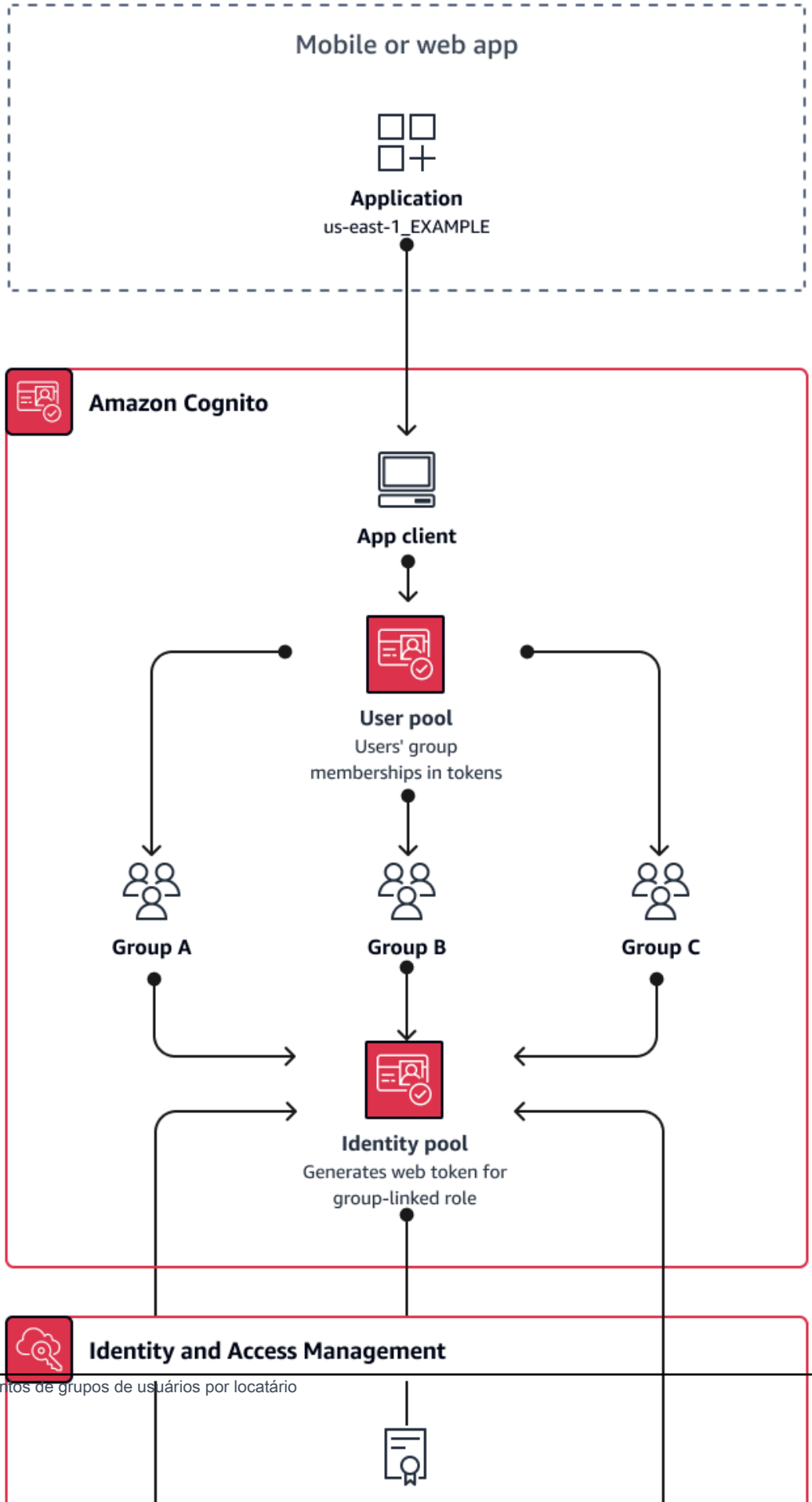
Práticas recomendadas de multilocação em conjuntos de grupos de usuários

A multilocação baseada em conjunto funciona melhor quando sua arquitetura exige grupos de usuários do Amazon Cognito com bancos de identidades.

Os tokens de ID e acesso do grupo de usuários contêm uma reivindicação `cognito:groups`. Além disso, os tokens de ID contêm reivindicações `cognito:roles` e `cognito:preferred_role`. Quando o resultado principal da autenticação na aplicação são credenciais da AWS temporárias de um banco de identidades, as associações de conjuntos de usuários podem determinar o perfil do IAM e as permissões que eles recebem.

Como exemplo, considere três locatários, em que cada um armazena ativos de aplicações em seu próprio bucket do Amazon S3. Atribua os usuários de cada locatário a um grupo associado, configure um perfil preferencial para o grupo e conceda a esse perfil acesso de leitura ao bucket.

O diagrama a seguir mostra inquilinos que compartilham um cliente de aplicação e um grupo de usuários, com grupos dedicados no grupo de usuários que determinam sua elegibilidade para um perfil do IAM.


Mobile or web app
Application
us-east-1_EXAMPLE
Amazon Cognito
App client
User pool
Users' group memberships in tokens
Group A
Group B
Group C
Identity pool
Generates web token for group-linked role
Identity and Access Management

Quando implementar a multilocação de conjuntos

Quando o acesso aos AWS recursos é sua principal preocupação. Os grupos de usuários do Amazon Cognito são um mecanismo de controle de acesso baseado em função (RBAC). Você pode configurar vários conjuntos em um grupo de usuários e tomar decisões complexas de RBAC com prioridade de conjunto. Os bancos de identidades podem atribuir credenciais para a função com a maior prioridade, qualquer função na reivindicação de conjuntos ou de outras reivindicações nos tokens de um usuário.

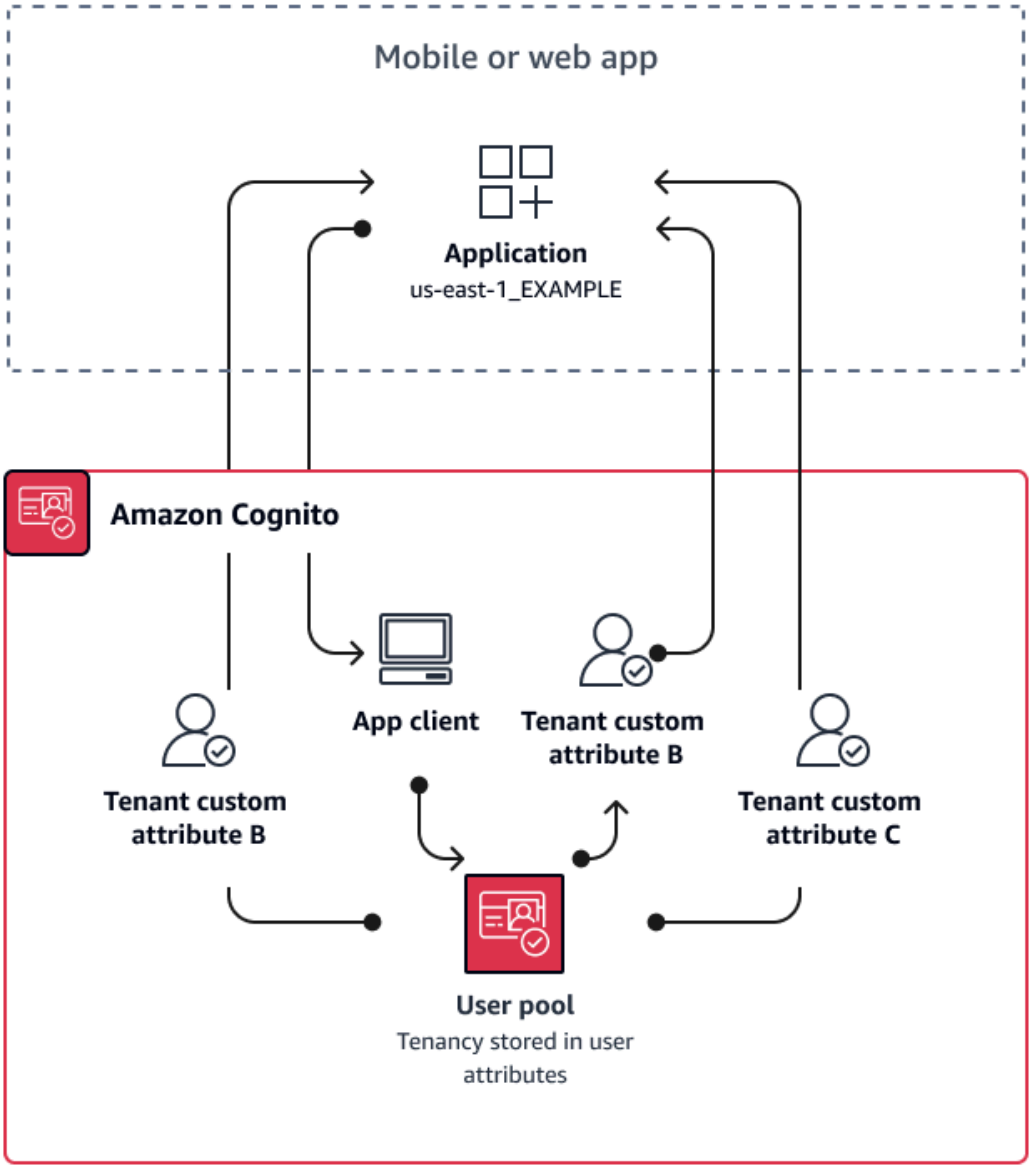
Nível de esforço

O nível de esforço para manter a multilocação apenas com a participação em conjuntos é baixo. No entanto, para expandir a função dos conjuntos de grupos de usuários além da capacidade integrada de seleção de perfis do IAM, você deve criar uma lógica de aplicação que processe a associação de conjunto nos tokens dos usuários e determine o que fazer no cliente. Você pode integrar o Amazon Verified Permissions às aplicações para tomar decisões de autorização do lado do cliente. Atualmente, os identificadores de grupo não são processados nas operações IsAuthorizedWithTokenda API de permissões verificadas, mas você pode desenvolver um código personalizado que analise o conteúdo dos tokens, incluindo declarações de associação a grupos.

Práticas recomendadas de multilocação por atributo personalizado

O Amazon Cognito permite o uso de atributos personalizados com nomes que você escolhe. Um cenário em que os atributos personalizados são úteis é quando eles distinguem a locação dos usuários em um grupo de usuários compartilhado. Quando você atribui aos usuários um valor para um atributo, como `custom:tenantID`, sua aplicação pode atribuir acesso a recursos específicos do locatário adequadamente. Um atributo personalizado que define um ID de locatário deve ser imutável ou somente leitura para o cliente de aplicação.

O diagrama a seguir mostra os locatários que compartilham um cliente de aplicação e um grupo de usuários, com atributos personalizados no grupo de usuários que indicam o locatário ao qual eles pertencem.



Quando atributos personalizados determinam a locação, você pode distribuir uma única aplicação ou URL de login. Depois que o usuário fizer login, a aplicação poderá processar a reivindicação `custom:tenantID`, determinar quais ativos carregar, a marca a ser aplicada e os recursos a serem exibidos. Para decisões avançadas de controle de acesso a partir dos atributos do usuário, configure seu grupo de usuários como um provedor de identidades no Amazon Verified Permissions e gere decisões de acesso a partir do conteúdo do ID ou dos tokens de acesso.

Quando implementar a multilocação de atributos personalizados

Quando a locação for superficial. Um atributo de locatário pode contribuir para os resultados de marca e layout. Quando você deseja ter um isolamento significativo entre os locatários, os atributos personalizados não são a melhor escolha. Qualquer diferença entre locatários que precise ser configurada no nível do grupo de usuários ou de cliente de aplicação, como MFA ou marca de interface do usuário hospedada, exige que você crie distinções entre os locatários de uma forma que os atributos personalizados não oferecem. Com bancos de identidades, você pode até escolher o perfil do IAM de seus usuários na reivindicação de atributo personalizado em seu token de ID.

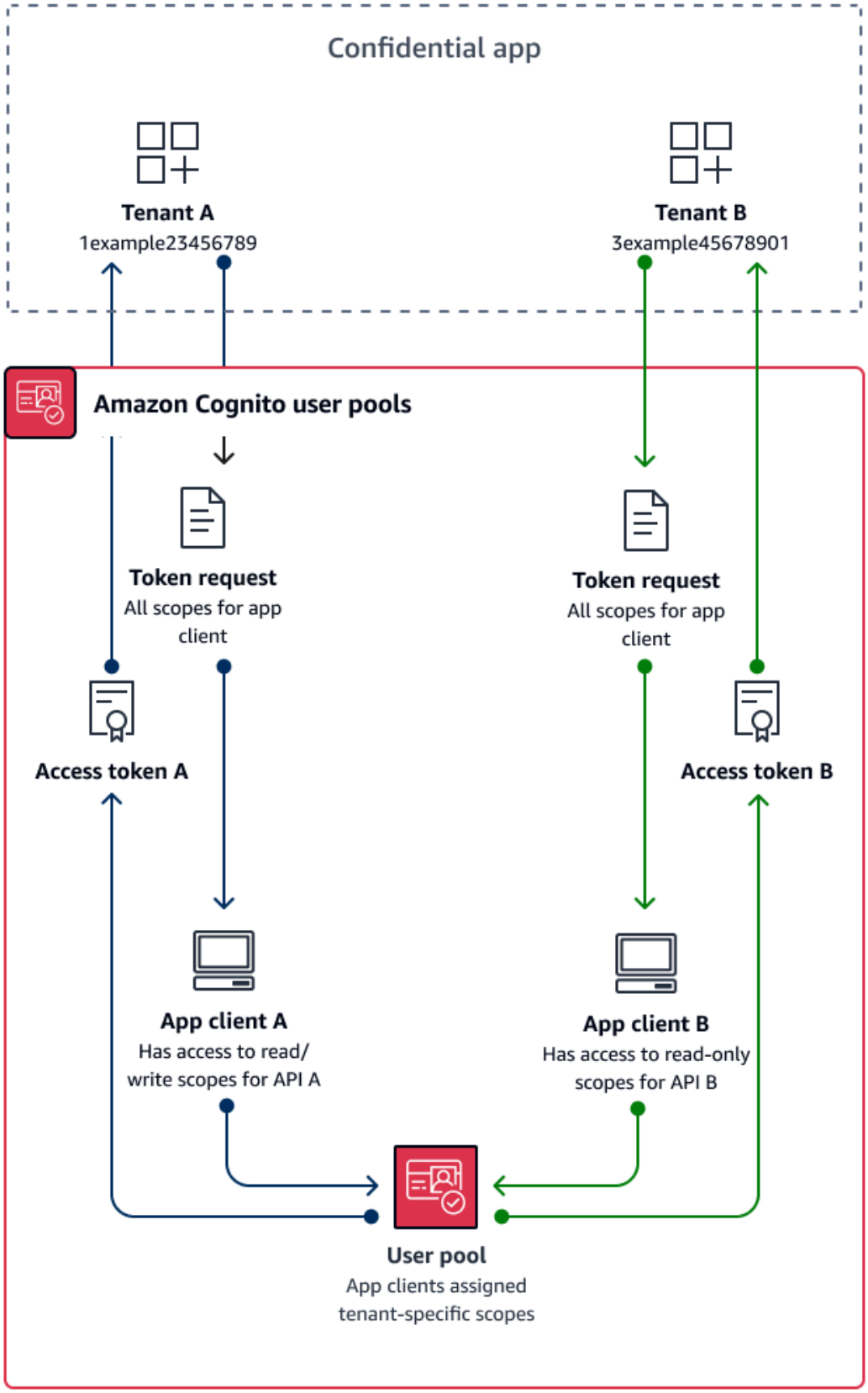
Nível de esforço

Como a multilocação de atributos personalizados transfere o dever das decisões de autorização com base no locatário na aplicação, o nível de esforço tende a ser alto. Se você já conhece bem uma configuração de cliente que analisa reivindicações de OIDC ou no Amazon Verified Permissions, essa abordagem pode exigir o menor nível de esforço.

Práticas recomendadas de multilocação por escopo personalizado

O Amazon Cognito oferece suporte a escopos OAuth 2.0 personalizados para servidores de recursos. Você pode implementar a multilocação de clientes de aplicativos em grupos de usuários para modelos de autorização machine-to-machine (M2M) com escopos personalizados. A multilocação baseada em escopo reduz o esforço necessário para implementar a multilocação M2M ao definir o acesso no cliente de aplicação ou na configuração da aplicação.

O diagrama a seguir ilustra uma opção para multilocação de escopo personalizado. Ele mostra cada locatário com um cliente de aplicação dedicado que tem acesso aos escopos relevantes em um grupo de usuários.


Confidential app
Tenant A
1example23456789
Tenant B
3example45678901
Amazon Cognito user pools
Token request
All scopes for app client
Token request
All scopes for app client
Access token A
Access token B
App client A
Has access to read/ write scopes for API A
App client B
Has access to read-only scopes for API B
User pool
App clients assigned tenant-specific scopes

Quando implementar a multilocação de escopos personalizados

Quando seu uso é autorização M2M com credenciais de cliente em um cliente confidencial. Como prática recomendada, crie servidores de recursos exclusivos para um cliente de aplicação. A multilocação de escopo personalizado pode depender da solicitação ou do cliente.

Dependente da solicitação

> Implemente a lógica de aplicação para solicitar somente os escopos que correspondam aos requisitos do seu locatário. Por exemplo, um cliente de aplicação pode emitir acesso de leitura e gravação à API A e à API B, mas a aplicação de locatário A solicita somente o escopo de leitura da API A e o escopo que indica locação. Esse modelo permite combinações mais complexas de escopos compartilhados entre locatários.

Dependente do cliente

> Solicite todos os escopos atribuídos a um cliente de aplicação em suas solicitações de autorização. Para fazer isso, omita o parâmetro de solicitação `scope` da solicitação para o Endpoint de token. Esse modelo permite que os clientes de aplicação armazenem os indicadores de acesso que você deseja adicionar aos escopos personalizados.

Nos dois casos, suas aplicações recebem tokens de acesso com escopos que indicam seus privilégios para as fontes de dados das quais dependem. Os escopos também podem apresentar outras informações à aplicação:

- Designar a locação
- Contribuir com o registro de solicitações
- APIs Indique que o aplicativo está autorizado a consultar
- Informar as verificações iniciais para clientes ativos.

Nível de esforço

A multilocação com escopo personalizado exige um nível variável de esforço em relação à escala da sua aplicação. Você deve criar uma lógica de aplicação que permita que as aplicações analisem tokens de acesso e façam as solicitações de API apropriadas.

Por exemplo, um escopo de servidor de recursos vem no formato `[resource server identifier]/[name]`. É improvável que o identificador do servidor de recursos seja relevante

para a decisão de autorização do escopo do locatário, exigindo que o nome do escopo seja analisado de forma consistente.

Exemplo de recurso

O AWS CloudFormation modelo a seguir cria um grupo de usuários para multilocação de escopo personalizado com um servidor de recursos e um cliente de aplicativo.

```
AWSTemplateFormatVersion: "2010-09-09"
Description: A sample template illustrating scope-based multi-tenancy
Resources:
  MyUserPool:
    Type: "AWS::Cognito::UserPool"
  MyUserPoolDomain:
    Type: AWS::Cognito::UserPoolDomain
    Properties:
      UserPoolId: !Ref MyUserPool
      # Note that the value for "Domain" must be unique across all of AWS.
      # In production, you may want to consider using a custom domain.
      # See: https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-
pools-add-custom-domain.html#cognito-user-pools-add-custom-domain-adding
      Domain: !Sub "example-userpool-domain-${AWS::AccountId}"
  MyUserPoolResourceServer:
    Type: "AWS::Cognito::UserPoolResourceServer"
    Properties:
      Identifier: resource1
      Name: resource1
      Scopes:
        - ScopeDescription: Read-only access
          ScopeName: readScope
      UserPoolId: !Ref MyUserPool
  MyUserPoolTenantBatch1ResourceServer:
    Type: "AWS::Cognito::UserPoolResourceServer"
    Properties:
      Identifier: TenantBatch1
      Name: TenantBatch1
      Scopes:
        - ScopeDescription: tenant1 identifier
          ScopeName: tenant1
        - ScopeDescription: tenant2 identifier
          ScopeName: tenant2
      UserPoolId: !Ref MyUserPool
  MyUserPoolClientTenant1:
```

```
    Type: "AWS::Cognito::UserPoolClient"
    Properties:
      AllowedOAuthFlows:
        - client_credentials
      AllowedOAuthFlowsUserPoolClient: true
      AllowedOAuthScopes:
        - !Sub "${MyUserPoolTenantBatch1ResourceServer}/tenant1"
        - !Sub "${MyUserPoolResourceServer}/readScope"
      GenerateSecret: true
      UserPoolId: !Ref MyUserPool
Outputs:
  UserPoolClientId:
    Description: User pool client ID
    Value: !Ref MyUserPoolClientTenant1
  UserPoolDomain:
    Description: User pool domain
    Value: !Sub "https://${MyUserPoolDomain}.auth.${AWS::Region}.amazoncognito.com"
```

Recomendações de segurança para locações múltiplas

Para ajudar a tornar sua aplicação mais segura, recomendamos o seguinte:

- Valide a locação na aplicação com o Amazon Verified Permissions. Crie políticas que examinem o direito ao grupo de usuários, ao cliente do aplicativo, ao grupo ou ao atributo personalizado antes de permitir a solicitação de um usuário em seu aplicativo. AWS criou fontes de identidade de Permissões Verificadas pensando nos grupos de usuários do Amazon Cognito. O Verified Permissions tem orientações adicionais para o gerenciamento de vários locatários.
- Use apenas um endereço de e-mail verificado para autorizar o acesso do usuário a um locatário com base na correspondência de domínio. Não confie em endereços de e-mail e números de telefone, a menos que sua aplicação os verifique ou o IdP externo forneça uma prova de verificação. Para obter mais detalhes sobre como configurar essas permissões, consulte Permissões e escopos de atributos.
- Use atributos imutáveis ou somente leitura para os atributos personalizados de perfil de usuário que identificam locatários. Você só pode definir o valor dos atributos imutáveis ao criar um usuário ou quando um usuário se cadastra no seu grupo de usuários. Além disso, conceda aos clientes da aplicação acesso somente leitura aos atributos.
- Use o mapeamento 1:1 entre o IdP externo de um locatário e o cliente da aplicação para impedir o acesso não autorizado entre locatários. Um usuário autenticado por um IdP externo e que tenha

um cookie de sessão válido do Amazon Cognito pode acessar outras aplicações de locatários que confiam no mesmo IdP.

- Ao implementar a lógica de correspondência e autorização de locatário em sua aplicação, restrinja os usuários de modo que eles não possam modificar os critérios que autorizam o acesso do usuário aos locatários. Além disso, se um IdP externo estiver sendo usado para federação, restrinja os administradores do provedor de identidade do locatário para que não possam modificar o acesso do usuário.

Cenários comuns do Amazon Cognito

Este tópico descreve seis cenários comuns para o uso do Amazon Cognito.

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 de aplicações Web e móveis. Os grupos de identidades fornecem AWS credenciais temporárias para conceder aos usuários acesso a outros Serviços da AWS.

Grupo de usuários é um diretório de usuários no Amazon Cognito. Os usuários da aplicação podem fazer login diretamente por meio de um grupo de usuários ou federar por meio de um provedor de identidades (IdP) de terceiros. O grupo de usuários gerencia a sobrecarga de lidar com os tokens que são retornados do login social por meio do Facebook, Google, Amazon e Apple, e do OpenID Connect (OIDC) e SAML. IdPs Quer os usuários façam login diretamente ou por meio de terceiros, todos os membros do grupo de usuários têm um perfil de diretório que você pode acessar por meio de um SDK.

Com um pool de identidades, seus usuários podem obter AWS credenciais temporárias para acessar AWS serviços, como Amazon S3 e DynamoDB. Os grupos de identidades oferecem suporte a usuários convidados anônimos, bem como à federação por meio de terceiros IdPs.

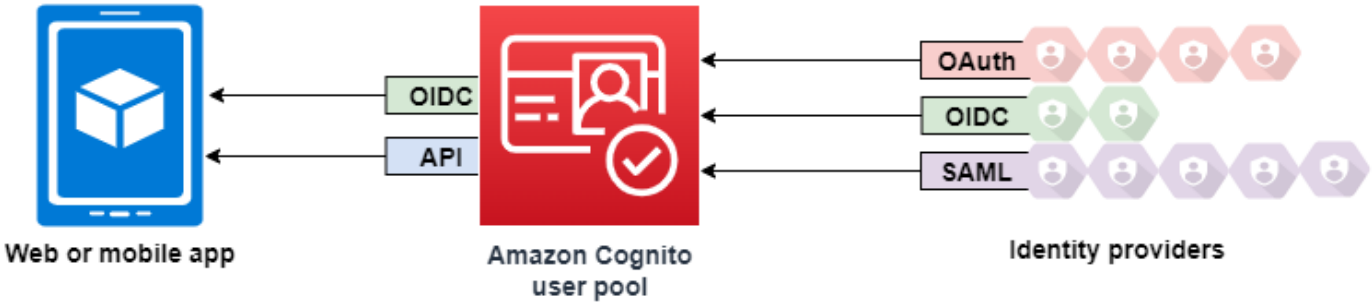
Tópicos

Autenticar com um grupo de usuários

Você pode permitir que os usuários sejam autenticados com um grupo de usuários. Os usuários da aplicação podem fazer login diretamente por meio de um grupo de usuários ou federar por meio de um provedor de identidades (IdP) de terceiros. O grupo de usuários gerencia a sobrecarga de lidar com os tokens que são retornados do login social por meio do Facebook, Google, Amazon e Apple, e do OpenID Connect (OIDC) e SAML. IdPs

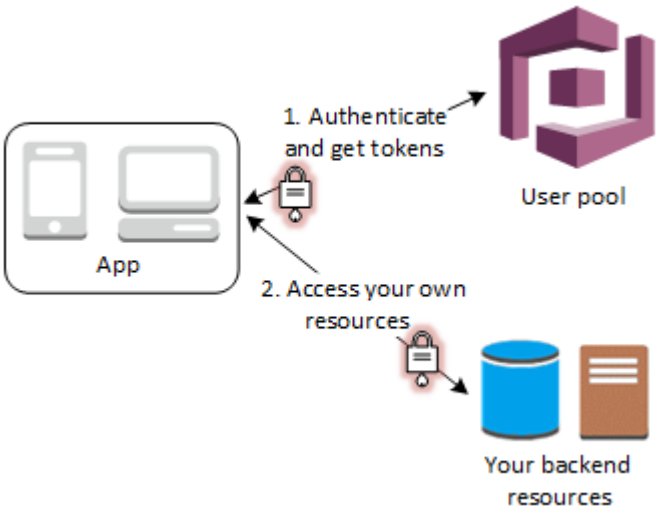
Depois de uma autenticação bem-sucedida, sua aplicação Web ou móvel receberá tokens do grupo de usuários do Amazon Cognito. Você pode usar esses tokens para recuperar AWS credenciais que permitem que seu aplicativo acesse outros AWS serviços, ou você pode optar por usá-los para controlar o acesso aos seus recursos do lado do servidor ou ao Amazon API Gateway.

Para obter mais informações, consulte Um exemplo de sessão de autenticação e Compreendendo os tokens web JSON do grupo de usuários () JWTs.



Acessar recursos de backend com tokens do grupo de usuários

Depois de um login no grupo de usuários bem-sucedido, sua aplicação Web ou móvel receberá tokens do grupo de usuários do Amazon Cognito. Você pode usar esses tokens para controlar o acesso aos recursos no lado do servidor. Também é possível criar grupos de usuários para gerenciar permissões e representar diferentes tipos de usuários. Para obter mais informações sobre o uso de grupos para controlar o acesso aos seus recursos, consulte Como adicionar grupos a um grupo de usuários.



Assim que você configura um domínio para o grupo de usuários, o Amazon Cognito provisiona uma interface de usuário da web hospedada que permite adicionar páginas de cadastro e login à aplicação. Usando essa base OAuth 2.0, você pode criar seu próprio servidor de recursos para permitir que seus usuários acessem recursos protegidos. Para obter mais informações, consulte Escopos, M2M e servidores de recursos.

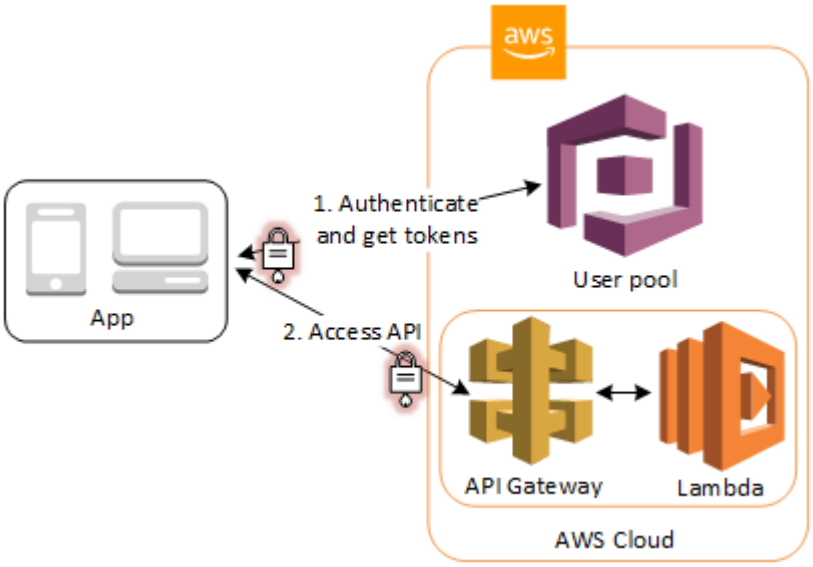
Para obter mais informações sobre a autenticação do grupo de usuários, consulte Um exemplo de sessão de autenticação e Compreendendo os tokens web JSON do grupo de usuários () JWTs.

Acessar recursos com o API Gateway e o Lambda com um grupo de usuários

Você pode permitir que seus usuários acessem a API por meio do API Gateway. O API Gateway valida os tokens de uma autenticação bem-sucedida do grupo de usuários e os usa para conceder aos usuários acesso a recursos, incluindo funções Lambda ou sua própria API.

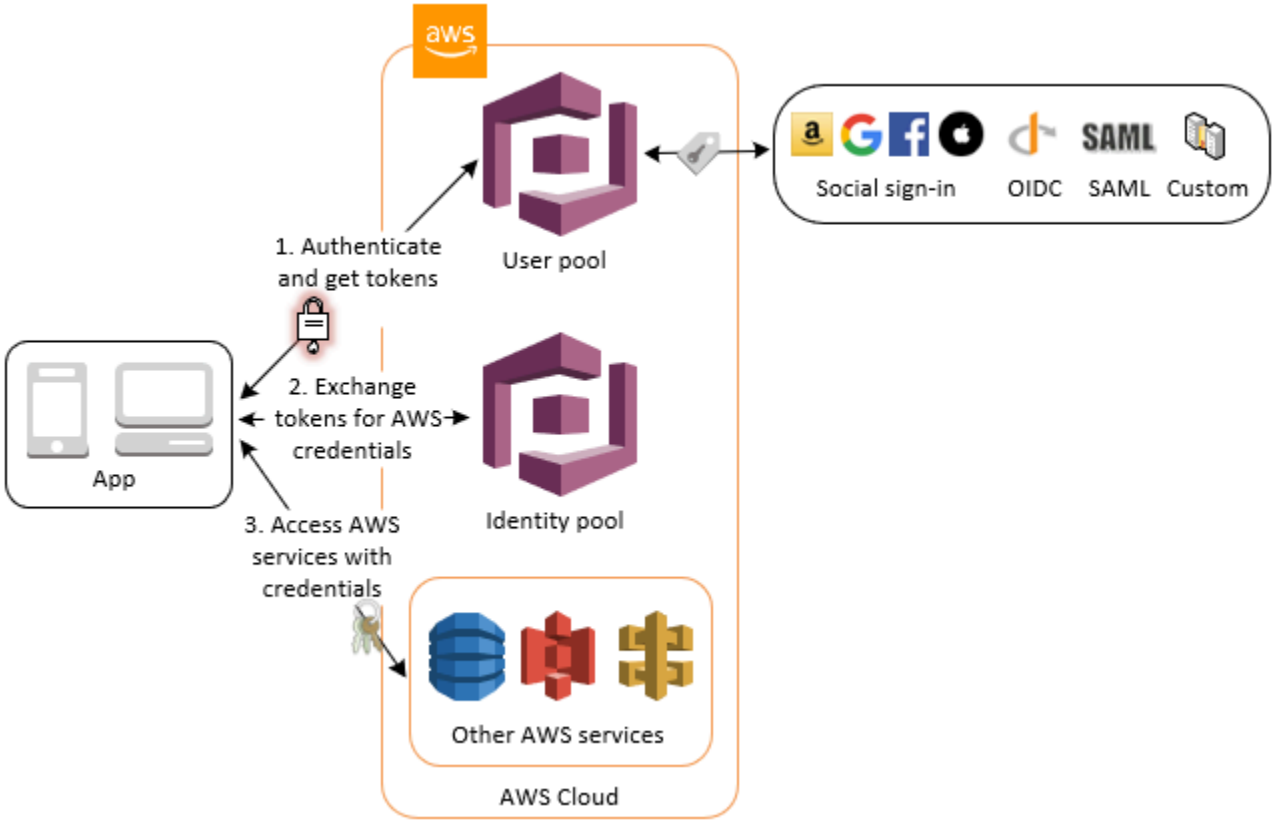
Você pode usar grupos em um grupo de usuários para controlar permissões com o API Gateway mapeando a associação ao grupo para funções do IAM. Os grupos dos quais um usuário é membro estão incluídos no token de ID fornecido por um grupo de usuários quando o usuário do aplicativo faz login. Para obter mais informações sobre grupos de usuários, consulte Como adicionar grupos a um grupo de usuários.

Você pode enviar seus tokens do grupo de usuários com uma solicitação ao API Gateway para verificação por uma função Lambda autorizadora do Amazon Cognito. Para obter mais informações sobre o API Gateway, consulte Usar o API Gateway com grupos de usuários do Amazon Cognito.



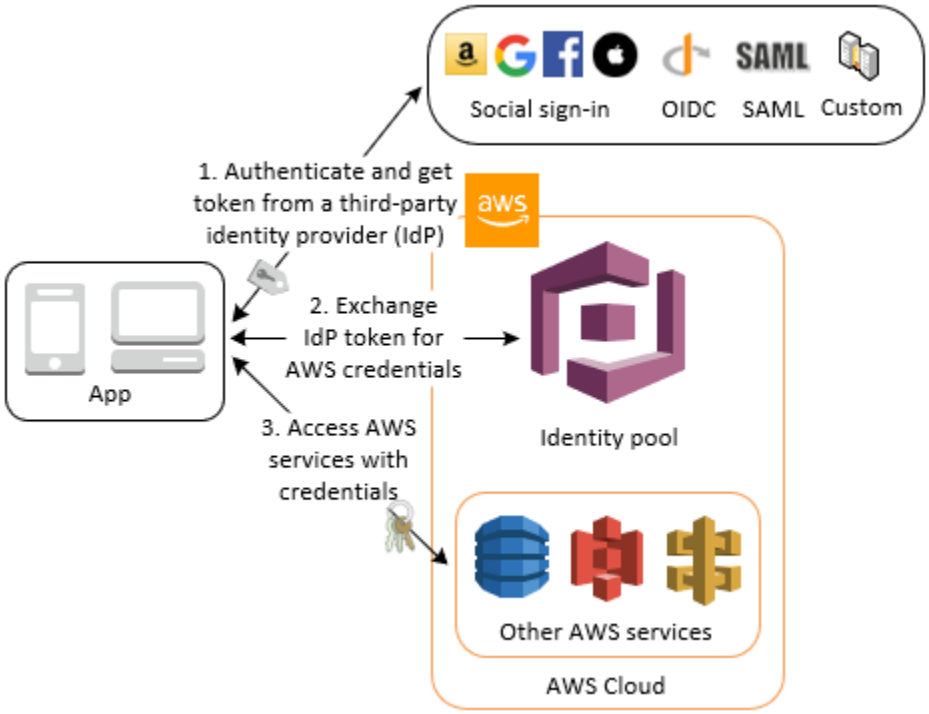
Acesse AWS serviços com um grupo de usuários e um pool de identidades

Depois de uma autenticação bem-sucedida no grupo de usuários, sua aplicação receberá tokens do grupo de usuários do Amazon Cognito. Você pode trocá-los por acesso temporário a outros AWS serviços com um pool de identidades. Para obter mais informações, consulte Acessando Serviços da AWS usando um pool de identidades após o login e Conceitos básicos dos bancos de identidades do Amazon Cognito.



Autentique-se com terceiros e acesse AWS serviços com um pool de identidades

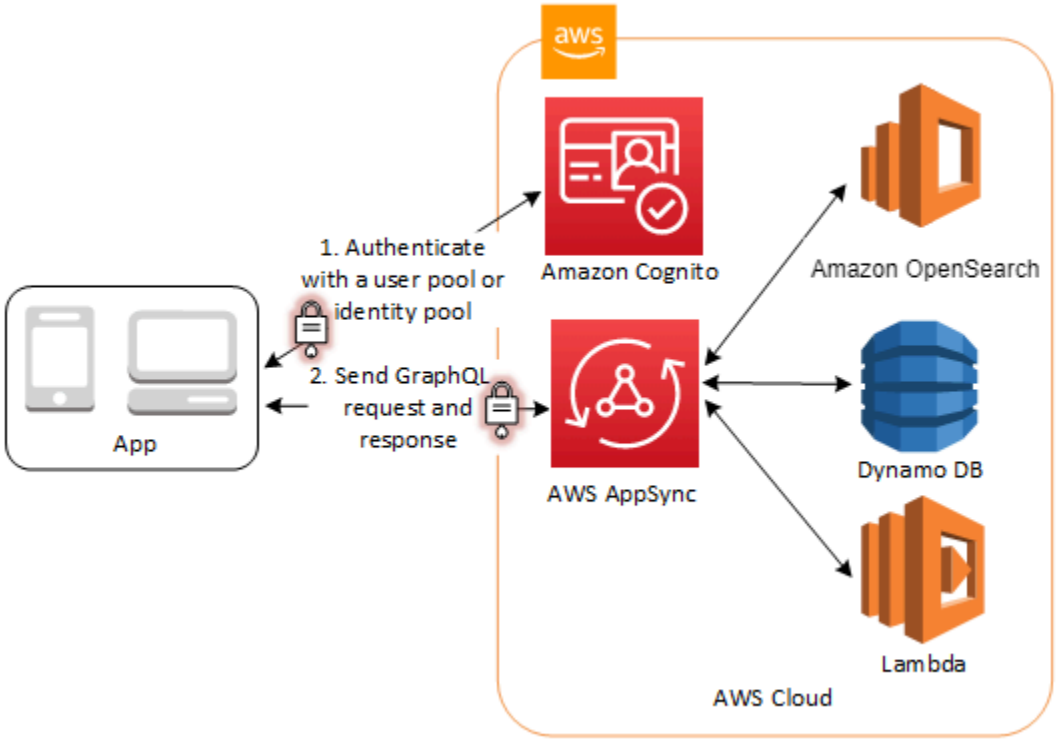
Você pode permitir que seus usuários acessem os AWS serviços por meio de um pool de identidades. Um grupo de identidades requer um token IdP de um usuário autenticado por um provedor de identidade de terceiros (ou nada se for um convidado anônimo). Em troca, o grupo de identidades concede AWS credenciais temporárias que você pode usar para acessar outros AWS serviços. Para obter mais informações, consulte Conceitos básicos dos bancos de identidades do Amazon Cognito.



Acesse AWS AppSync recursos com o Amazon Cognito

Você pode conceder aos seus usuários acesso a AWS AppSync recursos com tokens de uma autenticação bem-sucedida do grupo de usuários do Amazon Cognito. Para obter mais informações, consulte a AMAZON_COGNITO_USER_POOLS autorização no Guia do AWS AppSync desenvolvedor.

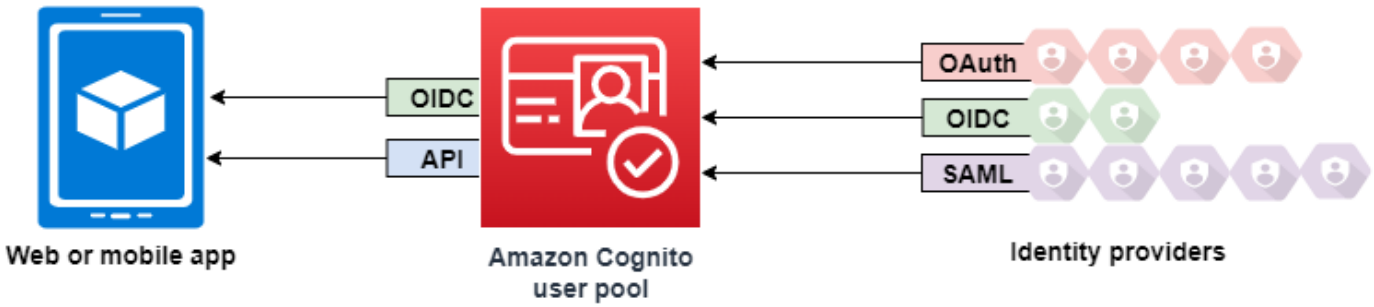
Você também pode assinar solicitações para a API AWS AppSync GraphQL com as credenciais do IAM que você recebe de um grupo de identidades. Veja a AWS_IAMautorização.


aws
Amazon Cognito
Amazon OpenSearch
1. Authenticate with a user pool or identity pool
2. Send GraphQL request and response
App
AWS AppSync
Dynamo DB
Lambda
AWS Cloud

Grupos de usuários do Amazon Cognito

Um grupo de usuários do Amazon Cognito é um diretório de usuários para autenticação e autorização de aplicativos móveis e aplicações web. Do ponto de vista da aplicação, um grupo de usuários do Amazon Cognito é um provedor de identidades (IdP) OpenID Connect (OIDC). Um grupo de usuários adiciona camadas de outros recursos para segurança, federação de identidades, integração de aplicações e personalização da experiência do usuário.

Você pode, por exemplo, verificar se as sessões dos usuários são de fontes confiáveis. É possível combinar o diretório do Amazon Cognito com um provedor de identidades externo. Com seu AWS SDK preferido, você pode escolher o modelo de autorização de API que funciona melhor para seu aplicativo. E pode adicionar funções do AWS Lambda que modificam ou inspecionam o comportamento padrão do Amazon Cognito.



Tópicos

- Recursos
- Planos de recursos de grupos de usuários
- Praticas recomendadas de segurança para grupos de usuários do Amazon Cognito
- Autenticação com grupos de usuários do Amazon Cognito
- Login do grupo de usuários com provedores de identidades de terceiros
- Login gerenciado do grupo de usuários
- Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda
- Como gerenciar usuários em seu grupo de usuários
- Compreendendo os tokens web JSON do grupo de usuários () JWTs
- Como acessar recursos após o login bem-sucedido

- Escopos, M2M e servidores de recursos
- Configurar recursos do grupo de usuários
- Usar atributos de segurança de grupos de usuários do Amazon Cognito
- Referência de login gerenciado e endpoints do grupo de usuários

Recursos

Os grupos de usuários do Amazon Cognito têm os recursos a seguir.

Cadastrar-se

Os grupos de usuários do Amazon Cognito têm métodos orientados pelo usuário, orientados pelo administrador e programáticos para adicionar perfis de usuário ao grupo de usuários. Os grupos de usuários do Amazon Cognito são compatíveis com os modelos de inscrição a seguir. É possível usar qualquer combinação desses modelos em sua aplicação.

> **Important**
>
> Se você ativar a inscrição de usuário no grupo de usuários, qualquer pessoa na internet poderá se inscrever em uma conta e entrar nas suas aplicações. Não habilite o autorregistro no grupo de usuários, a menos que queira abrir a aplicação para inscrição pública.
> Para alterar essa configuração, atualize a inscrição por autoatendimento no menu Inscrição em Autenticação no console do grupo de usuários ou atualize o valor de AllowAdminCreateUserOnlyem uma CreateUserPoolsolicitação de API. UpdateUserPool
> Para obter informações sobre os atributos de segurança que você pode configurar nos grupos de usuários, consulte Usar atributos de segurança de grupos de usuários do Amazon Cognito.

1. Os usuários podem inserir informações em sua aplicação e criar um perfil de usuário nativo para seu grupo de usuários. Você pode chamar as operações de inscrição da API para registrar usuários em seu grupo de usuários. Você pode abrir essas operações de inscrição para qualquer pessoa ou autorizá-las com um segredo ou AWS credenciais do cliente.
2. Você pode redirecionar os usuários para um IdP de terceiros que eles possam autorizar a transmitir as informações deles ao Amazon Cognito. O Amazon Cognito processa tokens de ID OIDC, `userInfo` dados OAuth 2.0 e declarações SAML 2.0 em perfis de usuário em seu grupo

de usuários. Você controla os atributos que deseja que o Amazon Cognito receba com base nas regras de mapeamento de atributos.

3. É possível ignorar a inscrição pública ou federada e criar usuários com base em sua própria fonte de dados e esquema. Adicione usuários diretamente no console ou na API do Amazon Cognito. Importe usuários de um arquivo CSV. Execute uma just-in-time AWS Lambda função que procure seu novo usuário em um diretório existente e preencha seu perfil de usuário a partir dos dados existentes.

Depois que os usuários se inscreverem, você poderá adicioná-los aos grupos que o Amazon Cognito lista nos tokens de acesso e ID. Você também pode vincular grupos de usuários a perfis do IAM ao transmitir o token de ID para um banco de identidades.

Tópicos relacionados

- Como gerenciar usuários em seu grupo de usuários
- Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado
- Exemplos de código para o Amazon Cognito Identity Provider usando AWS SDKs

Fazer login

O Amazon Cognito pode ser um diretório de usuários autônomo e um provedor de identidades (IdP) da aplicação. Os usuários podem fazer login com páginas de login gerenciado hospedadas pelo Amazon Cognito ou com um serviço de autenticação de usuários personalizado pela API de grupos de usuários do Amazon Cognito. O nível da aplicação por trás do frontend personalizado pode autorizar solicitações no backend com qualquer um dos vários métodos para confirmar solicitações legítimas.

Os usuários podem configurar e assinar com nomes de usuário e senhas, chaves de acesso e senhas de uso único de e-mail e SMS. Você pode oferecer login consolidado com diretórios de usuários externos, autenticação multifator (MFA) após o login, dispositivos confiáveis e fluxos de autenticação personalizados que você cria.

Para fazer login de usuários com um diretório externo, opcionalmente combinado com o diretório de usuários incorporado ao Amazon Cognito, você pode adicionar as integrações a seguir.

1. Faça login e importe dados do usuário do cliente com o login social OAuth 2.0. O Amazon Cognito suporta login com Google, Facebook, Amazon e Apple até 2.0. OAuth

2. Faça login e importe dados de usuários corporativos e acadêmicos com o login SAML e OIDC. Também é possível configurar o Amazon Cognito para aceitar declarações de qualquer provedor de identidades (IdP) SAML ou OpenID Connect (OIDC).
3. Vincule perfis de usuários externos a perfis de usuário nativos. Um usuário vinculado pode fazer login com uma identidade de usuário de terceiros e receber o acesso que você atribui a um usuário no diretório interno.

Tópicos relacionados

- Login do grupo de usuários com provedores de identidades de terceiros
- Vincular usuários federados a um perfil de usuário existente

Machine-to-machine autorização

Algumas sessões não são uma human-to-machine interação. Talvez você precise de uma conta de serviço que possa autorizar uma solicitação a uma API por meio de um processo automatizado. Para gerar tokens de acesso para machine-to-machine autorização com escopos OAuth 2.0, você pode adicionar um cliente de aplicativo que gere concessões de credenciais de cliente.

Tópicos relacionados

- Escopos, M2M e servidores de recursos

Login gerenciado

Quando você não quiser criar uma interface do usuário, poderá apresentar aos usuários páginas de login gerenciado personalizadas. O login gerenciado é um conjunto de páginas da web para inscrição, login, autenticação multifator (MFA) e redefinição de senha. Você pode adicionar login gerenciado ao seu domínio existente ou usar um identificador de prefixo em um AWS subdomínio.

Tópicos relacionados

- Login gerenciado do grupo de usuários
- Como configurar um domínio de grupo de usuários

Segurança

Os usuários locais podem fornecer um fator de autenticação adicional com um código de uma mensagem SMS ou de um e-mail ou uma aplicação que gere códigos de autenticação multifator (MFA). Você pode criar mecanismos para configurar e processar a MFA em sua aplicação ou deixar que o login gerenciado faça isso. Os grupos de usuários do Amazon Cognito podem ignorar a MFA quando os usuários fazem login em dispositivos confiáveis.

Se você não quiser exigir inicialmente a MFA dos usuários, poderá solicitá-la de maneira condicional. Com a autenticação adaptativa, o Amazon Cognito pode detectar possíveis atividades mal-intencionadas e exigir que seu usuário configure a MFA ou bloqueie o login.

Se o tráfego de rede para seu grupo de usuários puder ser malicioso, você poderá monitorá-lo e agir com a AWS WAF web ACLs.

Tópicos relacionados

- Adicionar MFA a um grupo de usuários
- Segurança avançada com proteção contra ameaças
- Associar uma ACL AWS WAF da web a um grupo de usuários

Experiência personalizada do cliente

Na maioria dos estágios da inscrição, login ou atualização do perfil de um usuário, você pode personalizar como o Amazon Cognito lida com a solicitação. Com os acionadores do Lambda, você pode modificar um token de ID ou rejeitar uma solicitação de inscrição com base em condições personalizadas. É possível criar seu próprio fluxo de autenticação personalizado.

Você pode carregar o CSS e logotipos personalizados para dar ao login gerenciado uma aparência familiar para os usuários.

Tópicos relacionados

- Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda
- Acionadores do Lambda de desafio personalizado de autenticação
- Aplicar a identidade visual às páginas de login gerenciado

Monitoramento e análise

Os grupos de usuários do Amazon Cognito registram em log solicitações de API, incluindo solicitações ao login gerenciado, no AWS CloudTrail. Você pode analisar métricas de desempenho no Amazon CloudWatch Logs, enviar registros personalizados CloudWatch com acionadores Lambda, monitorar a entrega de e-mails e mensagens SMS e monitorar o volume de solicitações de API no console de Service Quotas.

Com o plano de recursos Plus, você pode monitorar as tentativas de autenticação do usuário em busca de indicadores de comprometimento com a tecnologia de aprendizado automatizado e remediar imediatamente os riscos. Esses recursos avançados de segurança também registram a atividade do usuário no seu grupo de usuários e, opcionalmente, no Amazon S3 CloudWatch , Logs ou Amazon Data Firehose.

Também é possível registrar em log dados do dispositivo e da sessão das solicitações de API em uma campanha do Amazon Pinpoint. Com o Amazon Pinpoint, você pode enviar notificações push da aplicação com base em sua análise da atividade do usuário.

Tópicos relacionados

- Login no Amazon Cognito AWS CloudTrail
- Rastreamento de cotas e uso em CloudWatch e Service Quotas
- Exportação de logs dos grupos de usuários do Amazon Cognito
- Como usar o Amazon Pinpoint para análise de grupos de usuários

Integração de bancos de identidades do Amazon Cognito

A outra metade do Amazon Cognito são bancos de identidades. Os grupos de identidades fornecem credenciais que autorizam e monitoram solicitações de API para Serviços da AWS, por exemplo, Amazon DynamoDB ou Amazon S3, de seus usuários. É possível criar políticas de acesso baseadas em identidade que protejam os dados com base em como você classifica os usuários em seu grupo de usuários. Os bancos de identidades também podem aceitar tokens e declarações do SAML 2.0 de vários provedores de identidades, independentemente da autenticação do grupo de usuários.

Tópicos relacionados

- Acessando Serviços da AWS usando um pool de identidades após o login

- Banco de identidades do Amazon Cognito

Planos de recursos de grupos de usuários

Noções básicas sobre o custo é uma etapa crucial na preparação para implementar a autenticação de grupos de usuários do Amazon Cognito. O Amazon Cognito tem planos de recursos para grupos de usuários. Cada plano tem um conjunto de recursos e um custo mensal por usuário ativo. Cada plano de recursos libera o acesso a mais recursos do que o anterior.

Os grupos de usuários têm diversos recursos que você pode ativar e desativar. Por exemplo, você pode ativar a autenticação multifator (MFA) e desativar o login com provedores de identidade terceirizados (). IdPs Algumas mudanças exigem que você mude seu plano de recursos. As seguintes características do seu grupo de usuários determinam o custo que você AWS cobra mensalmente pelo uso.

- Os recursos que você escolhe
- As solicitações por segundo que sua aplicação faz à API de grupos de usuários
- O número de usuários com atividade de autenticação, atualização ou consulta em um mês, também chamado de usuários ativos mensais ou MAUs
- O número de usuários ativos mensais do SAML 2.0 ou do OpenID Connect (OIDC) de terceiros IdPs
- O número de clientes de aplicativos e grupos de usuários que concedem credenciais de clientes para autorização machine-to-machine

Para obter as informações mais atuais sobre os preços do grupo de usuários, consulte os preços do Amazon Cognito.

As seleções de plano de recursos se aplicam a um grupo de usuários. Grupos de usuários diferentes na mesma Conta da AWS podem ter diferentes seleções de planos. Você não pode aplicar planos de recursos separados a clientes de aplicações em um grupo de usuários. A seleção de planos padrão para novos grupos de usuários é Essentials.

Você pode alternar entre os planos de recursos a qualquer momento para atender aos requisitos das aplicações. Algumas mudanças entre os planos exigem que você desative os recursos ativos. Para obter mais informações, consulte Desativar recursos para alterar planos de recursos.

Planos de recursos de grupos de usuários

Lite

O Lite é um plano de recursos de baixo custo para grupos de usuários com menor número de usuários ativos mensais. Esse plano é suficiente para diretórios de usuários com recursos básicos de autenticação. Ele inclui recursos de login e a IU hospedada clássica, uma antecessora mais fina e menos personalizável do login gerenciado. Muitos recursos mais novos, como personalização do token de acesso e autenticação por chave de acesso, não estão incluídos no plano Lite.

Essentials

O Essentials tem todos os recursos mais recentes de autenticação de grupos de usuários. Esse plano adiciona novas opções às suas aplicações, independentemente de suas páginas de login serem gerenciadas ou personalizadas. O Essentials tem recursos avançados de autenticação, como login baseado em opções e MFA do e-mail.

Plus

O Plus inclui tudo do plano Essentials e adiciona recursos avançados de segurança que protegem seus usuários. Monitore as solicitações de login, cadastro e gerenciamento de senhas do usuário em busca de indicadores de comprometimento. Por exemplo, grupos de usuários podem detectar se os usuários estão fazendo login de um local inesperado ou usando uma senha que tenha sido parte de uma violação pública.

Os grupos de usuários com o plano Plus geram logs de detalhes da atividade do usuário e avaliações de risco. Você pode aplicar sua própria análise de uso e segurança a esses logs ao exportá-los para serviços externos.

> **Note**
>
> Anteriormente, alguns recursos do grupo de usuários foram incluídos em uma estrutura de preços de recursos de segurança avançados. Os recursos incluídos nessa estrutura agora estão no plano Essentials ou Plus.

Tópicos

- Recursos do plano Essentials
- Recursos do plano Plus
- Desativar recursos para alterar planos de recursos

Selecionar um plano de recursos

Console de gerenciamento da AWS

Como escolher um plano de recursos

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Selecione o menu Configurações e revise a guia Planos de recursos.
5. Analise os recursos disponíveis para você nos planos Lite, Essentials e Plus.
6. Para alterar seu plano, selecione Mudar para o Essentials ou Mudar para o Plus. Para mudar para o plano Lite, escolha Outros planos e, em seguida, Comparar com o Lite.
7. Na próxima tela, revise sua escolha e selecione Confirmar.

CLI/API/SDK

As UpdateUserPooloperações CreateUserPoole definem seu plano de recursos no `UserPoolTier` parâmetro. Quando você não especifica um valor para `UserPoolTier`, seu grupo de usuários usa `Essentials` como padrão. Se você definir `AdvancedSecurityMode` como `AUDIT` ou `ENFORCED`, seu nível de grupo de usuários deverá ser `PLUS` e o padrão ser `PLUS` quando não especificado.

Consulte Exemplos em CreateUserPool para obter a sintaxe. Consulte Consulte também em CreateUserPool para obter links para essa função em ou AWS SDKs para uma variedade de linguagens de programação.

```
"UserPoolTier": "PLUS"
```

No AWS CLI, essa opção é `--user-pool-tier` argumento.

```
--user-pool-tier PLUS
```

Consulte create-user-poole update-user-poolna referência de AWS CLI comandos para obter mais informações.

Recursos por plano

Recursos e planos em grupos de usuários

Recurso	Description	Plano de recursos
Proteja-se contra senhas inseguras	Verifique as senhas em texto simples em busca de indicadores de compromet imento no runtime	Plus
Proteja-se contra tentativas prejudiciais de login	Verifique as propriedades da sessão em busca de indicador es de comprometimento no runtime	Plus
Registre em log e analise a atividade do usuário	Gere logs das propriedades da sessão de autenticação do usuário e das pontuações de risco	Plus
Exporte logs de atividades do usuário	Envie a sessão do usuário e os registros de risco para um servidor externo AWS service (Serviço da AWS)	Plus
Personalize páginas de login gerenciado com um editor visual	Use um editor visual no console do Amazon Cognito para aplicar identidade visual e estilo às suas páginas de login gerenciado	Essentials + Plus
MFA aprimorada com códigos únicos de e-mail	Solicite ou exija que os usuários locais forneçam um fator adicional de login por	Essentials + Plus

Recurso	Description	Plano de recursos
	mensagem de e-mail após a autenticação do nome de usuário	
Personalize os escopos e as reivindicações do token de acesso no runtime	Use um acionador do Lambda para estender os recursos de autorização dos tokens de acesso ao grupo de usuários	Essentials + Plus
Login sem senha com códigos únicos	Permita que os usuários recebam uma senha de uso único por e-mail ou SMS como primeiro fator de autenticação	Essentials + Plus
Login por chave de acesso com autenticadores de hardware ou software FIDO2	Permita que os usuários usem uma chave criptográfica armazenada em um FIDO2 autenticador como primeiro fator de autenticação	Essentials + Plus
Inscrição e login	Execute operações de autenticação e permita que novos usuários se inscrevam em uma conta em sua aplicação.	Lite + Essentials + Plus
User groups (Grupos de usuários)	Crie agrupamentos lógicos de usuários e atribua perfis do IAM padrão para operações de banco de identidades.	Lite + Essentials + Plus
Faça login com provedores sociais, SAML e OIDC	Forneça aos usuários as opções de fazer login diretamente ou com o provedor que preferirem.	Lite + Essentials + Plus

Recurso	Description	Plano de recursos
OAuth 2.0/Servidor de autorização OIDC	Atue como emissor do OIDC.	Lite + Essentials + Plus
Páginas de login	Uma coleção hospedada de páginas da web para autenticação. O login gerenciado está disponível nos níveis Essentials e Plus. A IU hospedada clássica está disponível em todos os níveis de recursos.	Lite + Essentials + Plus
Autenticação personalizada, SRP, por senha e por token de atualização	Solicite aos usuários um nome de usuário e senha na aplicação.	Lite + Essentials + Plus
Machine-to-machine (M2M) com credenciais do cliente	Emita tokens de acesso para autorização de entidades não humanas.	Lite + Essentials + Plus
Autorização de API com servidores de recursos	Emita tokens de acesso com escopos personalizados que autorizam o acesso a sistemas externos.	Lite + Essentials + Plus
Importe usuários	Configure trabalhos de importação de arquivos CSV e realize a just-in-time migração dos usuários à medida que eles se conectam.	Lite + Essentials + Plus
MFA com aplicações autenticadoras e códigos SMS únicos	Solicite ou exija que os usuários locais forneçam uma mensagem SMS adicional ou um fator de login da aplicação autenticadora após a autenticação do nome de usuário	Lite + Essentials + Plus

Recurso	Description	Plano de recursos
Personalize os escopos e as reivindicações do token de ID no runtime	Use um acionador do Lambda para ampliar os recursos de autenticação dos tokens de identidade (ID) do grupo de usuários	Lite + Essentials + Plus
Ações de runtime personali zadas com acionadores do Lambda	Personalize o processo de login no runtime com funções do Lambda que realizam ações externas e influenciam a autenticação	Lite + Essentials + Plus
Personalize páginas de login gerenciado com CSS	Baixe um modelo CSS e altere alguns estilos em suas páginas de login gerenciado	Lite + Essentials + Plus

Recursos do plano Essentials

O plano de recursos Essentials tem a maioria dos melhores e mais recentes recursos de grupos de usuários do Amazon Cognito. Ao mudar do plano Lite para o Essentials, você obtém novos recursos para suas páginas de login gerenciado, autenticação multifator com senhas de uso único por mensagem de e-mail, uma política de senha aprimorada e tokens de acesso personalizados. Para continuar up-to-date com os novos recursos do grupo de usuários, escolha o plano Essentials para seus grupos de usuários.

As seções a seguir apresentam uma breve visão geral dos recursos que você pode adicionar à sua aplicação com o plano Essentials. Para obter informações detalhadas, consulte as páginas a seguir.

Recursos adicionais do

- Personalização do token de acesso: Acionador do Lambda antes da geração do token
- MFA do e-mail: MFA de mensagens SMS e e-mail
- Histórico de senha: Senhas, recuperação de contas e políticas de senha
- IU aprimorada: Aplicar a identidade visual às páginas de login gerenciado

Tópicos

Personalização do token de acesso

Os tokens de acesso de grupos de usuários concedem permissões às aplicações: para acessar uma API, recuperar atributos do usuário do endpoint userInfo ou estabelecer uma associação de grupo para um sistema externo. Em cenários avançados, você pode adicionar aos dados do token de acesso padrão do diretório do grupo de usuários parâmetros temporários adicionais que sua aplicação determina em tempo de execução. Por exemplo, talvez você queira verificar as permissões de API de um usuário com o Amazon Verified Permissions e ajustar os escopos no token de acesso adequadamente.

O plano Essentials complementa as funções existentes de um acionador de pré-geração de tokens. Com planos de nível inferior, você pode personalizar tokens de ID com reivindicações, funções e associação a grupos adicionais. O Essentials adiciona novas versões do evento de entrada acionador que personalizam reivindicações de tokens de acesso, funções, associação a grupos e escopos. A personalização do token de acesso está disponível para concessões de credenciais de clientes machine-to-machine (M2M) com a versão três do evento.

Para personalizar tokens de acesso

1. Selecione o plano de recursos Essentials ou Plus.
2. Crie uma função do Lambda para o acionador. Para usar nossa função de exemplo, configure-a para Node.js.
3. Preencha sua função do Lambda com nosso código de exemplo ou crie o seu. Sua função deve processar um objeto de solicitação do Amazon Cognito e retornar as alterações que você deseja incluir.
4. Atribua sua nova função como o acionador de pré-geração de tokens da versão dois ou três. Os eventos da versão dois personalizam tokens de acesso para identidades de usuários. A versão três personaliza os tokens de acesso para identidades de usuários e máquinas.

Saiba mais

- Personalizar o token de acesso
- Como personalizar tokens de acesso nos grupos de usuários do Amazon Cognito

MFA do e-mail

Grupos de usuários do Amazon Cognito podem ser configurados para usar o e-mail como o segundo fator na autenticação multifator (MFA). Com a MFA do e-mail, o Amazon Cognito pode enviar aos usuários um e-mail com um código de verificação que eles devem inserir para concluir o processo de autenticação. Isso adiciona uma importante camada a mais de segurança ao fluxo de login do usuário. Para habilitar a MFA baseada em e-mail, o grupo de usuários deve ser configurado para usar a configuração de envio de e-mail do Amazon SES em vez da configuração de e-mail padrão.

Quando seu usuário seleciona a MFA por mensagem de e-mail, o Amazon Cognito envia um código de verificação único para o endereço de e-mail registrado do usuário sempre que ele tenta fazer login. Em seguida, o usuário deve retornar esse código ao seu grupo de usuários para concluir o fluxo de autenticação e obter acesso. Mesmo que o nome de usuário e a senha do usuário estejam comprometidos, o usuário ainda precisa fornecer um fator adicional — o código enviado por e-mail — antes de acessar os recursos da aplicação.

Para obter mais informações, consulte MFA de mensagens SMS e e-mail. Veja a seguir uma visão geral de como configurar seu grupo de usuários e usuários para MFA.

Como configurar a MFA do e-mail no console do Amazon Cognito

1. Selecione o plano de recursos Essentials ou Plus.
2. No menu Login do seu grupo de usuários, edite a Autenticação multifator.
3. Escolha o nível de aplicação de MFA que você deseja configurar. Com a opção Exigir MFA, os usuários da API recebem automaticamente um desafio para configurar, confirmar e fazer login com API MFA. Em grupos de usuários que exigem MFA, o login gerenciado solicita que eles escolham e configurem um fator de MFA. Com a MFA opcional, sua aplicação deve oferecer aos usuários a opção de configurar a MFA e definir a preferência do usuário pela MFA do e-mail.
4. Em Métodos de MFA, selecione Mensagem de e-mail como uma das opções.

Saiba mais

- MFA de mensagens SMS e e-mail

Prevenção de reutilização de senhas

Por padrão, uma política de senhas de grupos de usuários do Amazon Cognito define os requisitos de tamanho e tipos de caracteres da senha, além da expiração temporária da senha. O plano Essentials adiciona a capacidade de aplicar o histórico de senhas. Quando um usuário tenta redefinir sua senha, seu grupo de usuários pode impedir que ele a defina com uma senha anterior. Para obter mais informações sobre a configuração da política de senha, consulte Como adicionar requisitos de senha do grupo de usuários. Veja a seguir uma visão geral de como configurar o grupo de usuários com uma política de histórico de senhas.

Como configurar o histórico de senhas no console do Amazon Cognito

1. Selecione o plano de recursos Essentials ou Plus.
2. No menu Métodos de autenticação do seu grupo de usuários, localize Política de senha e selecione Editar.
3. Configure outras opções disponíveis e defina um valor para Impedir o uso de senhas anteriores.

Saiba mais

- Senhas, recuperação de contas e políticas de senha

Login hospedado e servidor de autorização do login gerenciado

Os grupos de usuários do Amazon Cognito têm páginas da web opcionais que oferecem suporte às seguintes funções: um IdP do OpenID Connect (OIDC), um provedor de serviços ou parte confiável de IdPs terceiros e páginas públicas interativas com o usuário para cadastro e login. Essas páginas são chamadas coletivamente de login gerenciado. Quando você escolhe um domínio para seu grupo de usuários, o Amazon Cognito ativa automaticamente essas páginas. O plano Lite tem a interface hospedada, já o plano Essentials abre essa versão avançada das páginas de inscrição e login.

As páginas de login gerenciadas têm uma up-to-date interface limpa com mais recursos e opções para personalizar sua marca e estilos. O plano Essentials é o nível mais baixo do plano que desbloqueia o acesso ao login gerenciado.

Como configurar o login gerenciado no console do Amazon Cognito

1. No menu Configurações, selecione o plano de recursos Essentials ou Plus.

2. No menu Domínio, atribua um domínio ao seu grupo de usuários e selecione uma versão da marca do login gerenciado.
3. No menu Login gerenciado, na guia Estilos, selecione Criar um estilo e atribua o estilo a um cliente de aplicação ou crie um novo cliente de aplicação.

Saiba mais

- Login gerenciado do grupo de usuários

Autenticação baseada em opções

O nível Essentials introduz um novo fluxo de autenticação para operações de autenticação na IU aprimorada e nas operações de API baseadas em SDK. Esse fluxo é uma autenticação baseada em opções. A autenticação baseada em opções é um método em que a autenticação de seus usuários começa não com uma declaração de um método de login no na aplicação, mas com uma consulta de possíveis métodos de login seguida por uma escolha. Você pode configurar seu grupo de usuários para oferecer suporte à autenticação baseada em opções e desbloquear a autenticação por nome de usuário, senha, sem senha e chave de acesso. Na API, esse é o fluxo `USER_AUTH`.

Como configurar a autenticação baseada em opções no console do Amazon Cognito

1. Selecione o plano de recursos Essentials ou Plus.
2. No menu Login do seu grupo de usuários, edite Opções para login baseado em opções. Selecione e configure os métodos de autenticação que você deseja habilitar na autenticação baseada em opções.
3. No menu Métodos de autenticação do seu grupo de usuários, edite a configuração das operações de login.

Saiba mais

- Autenticação com grupos de usuários do Amazon Cognito

Recursos do plano Plus

O plano de recursos Plus tem recursos avançados de segurança para grupos de usuários do Amazon Cognito. Esses recursos registram e analisam o contexto do usuário no runtime em busca

de possíveis problemas de segurança em dispositivos, locais, dados de solicitações e senhas. Em seguida, eles reduzem os riscos potenciais com respostas automáticas que bloqueiam ou adicionam proteções de segurança às contas dos usuários. Você também pode exportar seus registros de segurança para o Amazon S3, Amazon Data Firehose ou Amazon CloudWatch Logs para análise posterior.

Ao mudar do plano Essentials para o plano Plus, você obtém todos os recursos do Essentials e os recursos adicionais a seguir. Isso inclui o conjunto de opções de segurança de proteção contra ameaças, também conhecido como recursos avançados de segurança. Para configurar seus grupos de usuários para se adaptarem automaticamente às ameaças em seu frontend de autenticação, escolha o plano Plus para seus grupos de usuários.

As seções a seguir apresentam uma breve visão geral dos recursos que você pode adicionar à sua aplicação com o plano Plus. Para obter informações detalhadas, consulte as páginas a seguir.

Recursos adicionais do

- Autenticação adaptável: Trabalhar com autenticação adaptável
- Credenciais comprometidas: Trabalhar com a detecção de credenciais comprometidas
- Exportação de log: Exportação de logs dos grupos de usuários do Amazon Cognito

Tópicos

Proteção contra ameaças: autenticação adaptável

O plano Plus inclui um recurso de autenticação adaptável. Quando você ativa esse recurso, seu grupo de usuários faz uma avaliação de risco de cada sessão de autenticação de usuário. Com base nas classificações de risco resultantes, você pode bloquear a autenticação ou promover a MFA para usuários que fazem login com um nível de risco acima de um limite determinado por você. Com a autenticação adaptável, seu grupo de usuários e sua aplicação bloqueiam ou configuram automaticamente a MFA para usuários com contas sob suspeita de ataque. Você também pode fornecer feedback sobre as classificações de risco do seu grupo de usuários para ajustar as avaliações futuras.

Como configurar a autenticação adaptável no console do Amazon Cognito

1. Selecione o plano de recursos Plus.
2. No menu Proteção contra ameaças do seu grupo de usuários, edite a Autenticação padrão e personalizada em Proteção contra ameaças.
3. Você pode definir o Modo de imposição para autenticação padrão ou personalizada como Função completa.
4. Em Autenticação adaptativa, configure respostas automáticas de risco para diferentes níveis de risco.

Saiba mais

- Trabalhar com autenticação adaptável
- Coletar dados para proteção contra ameaças em aplicações

Proteção contra ameaças: detecção de credenciais comprometidas

O plano Plus inclui um recurso de detecção de credenciais comprometidas. Esse recurso protege contra o uso de senhas inseguras e a ameaça de acesso não intencional às aplicações que essa prática cria. Quando você permite que seus usuários façam login com nome de usuário e senha, eles podem reutilizar uma senha que usaram em outro lugar. Essa senha pode ter vazado ou ser simplesmente adivinhada. Com a detecção de credenciais comprometidas, seu grupo de usuários lê as senhas enviadas pelos usuários e as compara aos bancos de dados de senhas. Se a operação resultar na decisão de que provavelmente a senha está comprometida, você pode configurar seu grupo de usuários para bloquear o login e, em seguida, iniciar uma redefinição de senha para o usuário em sua aplicação.

A detecção de credenciais comprometidas pode reagir a senhas inseguras quando novos usuários se inscrevem, quando usuários existentes fazem login e quando usuários tentam redefinir as senhas. Com esse recurso, seu grupo de usuários pode impedir ou avisar sobre o login com senhas inseguras onde quer que os usuários as insiram.

Como configurar a detecção de credenciais comprometidas no console do Amazon Cognito

1. Selecione o plano de recursos Plus.
2. No menu Proteção contra ameaças do seu grupo de usuários, edite a Autenticação padrão e personalizada em Proteção contra ameaças.

3. Você pode definir o Modo de imposição para autenticação padrão ou personalizada como Função completa.
4. Em Credenciais comprometidas, configure os tipos de operações de autenticação que deseja verificar e a resposta automática que o seu grupo de usuários deve gerar.

Saiba mais

- Trabalhar com a detecção de credenciais comprometidas

Proteção contra ameaças: logs de atividades de usuários

O plano Plus adiciona um recurso de log que fornece análises de segurança e detalhes das tentativas de autenticação dos usuários. Você pode ver avaliações de risco, endereços IP de usuários, agentes de usuário e outras informações sobre o dispositivo conectado à aplicação. Você pode agir com base nessas informações com os recursos integrados de proteção contra ameaças ou pode analisar os logs nos próprios sistemas e tomar as medidas apropriadas. Você pode exportar os registros da proteção contra ameaças para o Amazon S3, CloudWatch Logs ou Amazon DynamoDB.

Como configurar o registro de atividades de usuários no console do Amazon Cognito

1. Selecione o plano de recursos Plus.
2. No menu Proteção contra ameaças do seu grupo de usuários, edite a Autenticação padrão e personalizada em Proteção contra ameaças.
3. Você pode definir o Modo de imposição para autenticação padrão ou personalizada como Somente auditoria. Essa é a configuração mínima para logs. Você também pode ativá-lo no modo de Função completa e configurar outros recursos de proteção contra ameaças.
4. Para exportar seus registros para outra pessoa AWS service (Serviço da AWS) para análise de terceiros, acesse o menu de streaming de registros do seu grupo de usuários e configure um destino de exportação.

Saiba mais

- Como exportar eventos de autenticação de usuários
- Exportação de logs dos grupos de usuários do Amazon Cognito

Desativar recursos para alterar planos de recursos

Os planos de recursos adicionam opções de configuração ao seu grupo de usuários. Você pode configurar e usar esses recursos somente quando o plano de recursos relacionado estiver ativo. Por exemplo, você pode configurar a personalização do token de acesso nos planos Plus e Essentials, mas não no plano Lite. Para desativar esses recursos, você deve desativar cada componente ativo. A opção Alternar para no menu Configurações no console do Amazon Cognito notifica você sobre os recursos que devem ser desativados para alteração do plano de recursos. Neste capítulo, você pode aprender as alterações que a desativação faz na configuração do grupo de usuários e como desativar esses recursos individualmente.

Personalização do token de acesso

- Para mudar para um plano que não inclua a personalização de tokens de acesso, você deve remover o acionador do Lambda de pré-geração de tokens do seu grupo de usuários. Para adicionar um novo acionador de pré-geração de token sem a personalização do token de acesso, atribua uma nova função ao acionador e configure-o para eventos da `V1_0`. Esses eventos de acionador da versão 1 só podem processar alterações nos tokens de ID.

  Para desativar manualmente a personalização do token de acesso, remova o acionador de pré-geração do token e adicione um novo acionador da versão um.

Proteção contra ameaças

- Para mudar para um plano sem proteção contra ameaças, desative todos os recursos do menu Proteção contra ameaças do seu grupo de usuários.

Exportação de log

- Para mudar para um plano sem exportação de logs, desative essa opção no menu Fluxo de logs do seu grupo de usuários. Seu grupo de usuários deixa de gerar logs de atividades de usuários locais ou exportados. Você também pode enviar uma solicitação de SetLogDeliveryConfigurationAPI que remove qualquer configuração com um `EventSource` valor de`UserActivity`.

MFA do e-mail

- Para mudar para um plano sem MFA do e-mail, acesse o menu Login do seu grupo de usuários. Edite a Autenticação multifator e desmarque Mensagem de e-mail como um dos métodos de MFA disponíveis.

Praticas recomendadas de segurança para grupos de usuários do Amazon Cognito

Esta página descreve as práticas recomendadas de segurança que você pode implementar para proteção contra ameaças comuns. A configuração escolhida dependerá do caso de uso de cada aplicação. Recomendamos que, pelo menos, sejam aplicados privilégios mínimos às operações administrativas e sejam tomadas medidas para proteger os segredos da aplicação e do usuário. Outra etapa avançada, mas eficaz, que você pode realizar é configurar e aplicar ACLs a AWS WAF web aos grupos de usuários.

Proteja seu grupo de usuários no nível da rede

AWS WAF A web ACLs pode proteger o desempenho e o custo dos mecanismos de autenticação que você cria com o Amazon Cognito. Com a web ACLs, você pode implementar grades de proteção na frente da API e das solicitações de login gerenciadas. ACLs Crie filtros de camada de rede e de aplicativos na Web que podem reduzir o tráfego ou exigir um CAPTCHA com base nas regras que você cria. As solicitações não são transmitidas para seus recursos do Amazon Cognito até que atendam às qualificações nas regras da Web ACL. Para obter mais informações, consulte AWS WAF web ACLs.

Proteger contra o abuso de mensagens SMS

Ao permitir a inscrição pública em seu grupo de usuários, você pode configurar a verificação da conta com códigos que o Amazon Cognito envia em mensagens de texto SMS. As mensagens SMS podem ser associadas a atividades indesejadas e aumentar sua AWS fatura. Configure sua infraestrutura para ser resiliente contra o envio de mensagens SMS em situações de fraude. Para obter mais informações, revise as seguintes postagens de AWS Blogs.

- Reduce risks of user sign-up fraud and SMS pumping with Amazon Cognito user pools
- Defesa contra o bombeamento de SMS: novos AWS recursos para ajudar a combater o tráfego inflado artificialmente

Entender a autenticação pública

Os grupos de usuários do Amazon Cognito têm recursos de gerenciamento de identidade e acesso do cliente (CIAM) que oferecem suporte a casos de uso em que membros do público podem cadastrar uma conta de usuário e acessar suas aplicações. Quando um grupo de usuários permite

a inscrição por autoatendimento, ele está aberto a solicitações de contas de usuário da internet pública. As solicitações de autoatendimento vêm de operações de API, como SignUpe InitiateAuth, e da interação do usuário com o login gerenciado. Você pode configurar grupos de usuários para mitigar abusos decorrentes de solicitações públicas ou desabilitar totalmente as operações de autenticação pública.

As configurações a seguir são algumas das maneiras pelas quais você pode gerenciar solicitações de autenticação pública e interna em seus grupos de usuários e clientes de aplicações.

Exemplos de configurações do grupo de usuários que afetam o acesso público ao grupo de usuários

Configuração	Opções disponíveis	Configurado em	Efeito na autenticação pública	Configuração do console	Operação e parâmetro da API
Cadastro por autoatendimento	Permita que os usuários se inscrevam em uma conta ou criem contas de usuário como administrador.	Grupo de usuários	Impedir cadastro público	Cadastrar-se – Cadastro por autoatendimento	CreateUserPool, UpdateUserPool `AdminCreateUserConfig – AllowAdminCreateUserOnly`
Confirmação do administrador	Envie códigos de confirmação para novos usuários ou exija que os administradores os confirmem.	Grupo de usuários	Impedir a confirmação de cadastro sem ação do administrador	Cadastrar-se – Verificação e confirmação assistidas pelo Cognito	CreateUserPool, UpdateUserPool `AccountRecoverySettings – admin_only`

Configuração	Opções disponíveis	Configurado em	Efeito na autenticação pública	Configuração do console	Operação e parâmetro da API
Divulgação de usuário	Entregue mensagens de “usuário não encontrado” no login e na redefinição de senha ou evite a divulgação.	Cliente da aplicação	Proteger contra adivinhação do nome de login, endereço de e-mail ou números de telefone	Clientes da aplicação – Impedir erros de existência de usuário	CreateUserPoolClient, UpdateUserPoolClient `PreventUserExistenceErrors`
Segredo do cliente	Exigir ou não um hash secreto no cadastro, login e redefinição de senha	Cliente da aplicação	Proteger contra solicitações de autenticação de fontes não autorizadas	Clientes de aplicações — Segredo do cliente	CreateUserPoolClient `GenerateSecret`
Web ACLs	Habilitar ou não um firewall de rede para solicitações de autenticação	Grupo de usuários	Limitar ou impedir o acesso com base nas características de solicitação definidas pelo administrador e nas regras de endereço IP	AWS WAF – Configurações WAF	AssociateWebACL `ResourceArn`

Configuração	Opções disponíveis	Configurado em	Efeito na autenticação pública	Configuração do console	Operação e parâmetro da API
IdP externo	Permitir o login de usuários de terceiros IdPs, no diretório do grupo de usuários ou em ambos	Cliente da aplicação	Exclua usuários locais ou usuários federados do cadastro e login.	Clientes da aplicação — Provedores de identidade	CreateUserPoolClient, UpdateUserPoolClient `SupportedIdentityProviders`
Servidor de autorização	Hospedar ou não páginas da web públicas para autenticação	Grupo de usuários	Desativar páginas da web públicas e permitir somente a autenticação baseada em SDK	Domínio	CreateUserPoolDomain A criação de qualquer domínio de grupo de usuários disponibiliza páginas da web públicas.

Configuração	Opções disponíveis	Configurado em	Efeito na autenticação pública	Configuração do console	Operação e parâmetro da API
Proteção contra ameaças	Habilitar ou desabilitar o monitoramento de sinais de atividade maliciosa ou senhas inseguras	Grupo de usuários ou cliente da aplicação	Poder bloquear automaticamente o login ou exigir MFA quando os usuários mostram indicadores de comprometimento	Proteção contra ameaças — Configurações de proteção	SetRiskConfiguration Os parâmetros de `SetRiskConfiguration` definem suas configurações de proteção contra ameaças.

Proteger clientes confidenciais com segredos do cliente

O segredo do cliente é uma string opcional associada a um cliente de aplicação. Todas as solicitações de autenticação para clientes de aplicações com segredos de cliente devem incluir um hash secreto gerado com base no nome de usuário, ID do cliente e segredo do cliente. Aqueles que não conhecem o segredo do cliente são excluídos da sua aplicação desde o início.

No entanto, os segredos do cliente têm limitações. Se você incorporar um segredo de cliente a um software-cliente público, o segredo do cliente estará aberto à inspeção. Assim, abre-se a capacidade de criar usuários, enviar solicitações de redefinição de senha e realizar outras operações no cliente da sua aplicação. Os segredos do cliente devem ser implementados somente quando uma aplicação é a única entidade que tem acesso ao segredo. Normalmente, isso é possível em aplicações clientes confidenciais do lado do servidor. Isso também se aplica às aplicações M2M, nas quais é necessário

um segredo do cliente. Armazene o segredo do cliente em um armazenamento local criptografado ou AWS Secrets Manager. Nunca deixe o segredo de seu cliente ser visível na internet pública.

Proteger outros segredos

Seu sistema de autenticação com grupos de usuários do Amazon Cognito pode lidar com dados, senhas e credenciais da AWS . Estas são algumas das práticas recomendadas para lidar com segredos que sua aplicação pode acessar.

Senhas

Os usuários podem inserir senhas ao entrarem na sua aplicação. O Amazon Cognito tem tokens de atualização que sua aplicação pode empregar para continuar as sessões de usuário expiradas sem uma nova solicitação de senha. Não coloque nenhuma senha ou hash de senha no armazenamento local. Projete sua aplicação para tratar as senhas como não visíveis e transmiti-las somente para seu grupo de usuários.

Como prática recomendada, implemente a autenticação sem senha com WebAuthn chaves de acesso. Se você precisar implementar senhas, use o fluxo de autenticação Secure Remote Password (SRP) e a autenticação multifator (MFA).

AWS credenciais

A autenticação administrativa e as operações administrativas do grupo de usuários exigem autenticação com AWS credenciais. Para implementar essas operações em um aplicativo, conceda acesso seguro às AWS credenciais temporárias. Conceda acesso às credenciais somente às aplicações executadas em um componente do servidor que você controla. Não coloque aplicativos que tenham AWS credenciais em sistemas públicos de controle de versão, como o. GitHub Não codifique AWS credenciais em aplicativos públicos do lado do cliente.

Verificador de código PKCE

O Proof Key for Code Exchange, ou PKCE, é para concessões de código de autorização do OpenID Connect (OIDC) com seu servidor de autorização de grupo de usuários. As aplicações compartilham segredos do verificador de código com seu grupo de usuários quando solicitam códigos de autorização. Para trocar códigos de autorização por tokens, os clientes devem reafirmar que conhecem o verificador do código. Essa prática evita a emissão de tokens com códigos de autorização interceptados.

Os clientes devem gerar um novo verificador de código aleatório com cada solicitação de autorização. O uso de um verificador de código estático ou previsível significa que o invasor só

precisa interceptar o verificador codificado e o código de autorização. Projete sua aplicação para que ela não exponha os valores do verificador de código aos usuários.

Privilégio mínimo de administração do grupo de usuários

As políticas do IAM podem definir o nível de acesso das entidades principais à administração do grupo de usuários e às operações de autenticação administrativa do Amazon Cognito. Por exemplo:

- Para um servidor web, conceda permissões para autenticação com operações administrativas de API.
- Para um Centro de Identidade do AWS IAM usuário que gerencia um grupo de usuários no seu Conta da AWS, conceda permissões para manutenção e geração de relatórios do grupo de usuários.

O nível de granularidade dos recursos no Amazon Cognito é limitado a dois tipos de recursos para fins de política do IAM: grupo de usuários e banco de identidades. Observe que não é possível aplicar permissões para gerenciar clientes de aplicação individuais. Configure grupos de usuários com o conhecimento de que as permissões que você concede são efetivas em todos os clientes da aplicação. Quando sua organização tem vários locatários de aplicação e seu modelo de segurança exige a separação das responsabilidades administrativas entre os locatários, implemente a multilocação com um locatário por grupo de usuários.

Embora seja possível criar políticas do IAM com permissões para operações de autenticação do usuário como `InitiateAuth`, essas permissões não têm efeito. As operações de API públicas e autorizadas por tokens não estão sujeitas às permissões do IAM. Das operações de autenticação de grupos de usuários disponíveis, você só pode conceder permissões para operações administrativas do servidor, como `AdminInitiateAuth`.

Você pode limitar os níveis de administração do grupo de usuários com listas de privilégios mínimos `Action`. O exemplo de política a seguir é para um administrador que pode gerenciar servidores de recursos IdPs, clientes de aplicativos e o domínio do grupo de usuários, mas não usuários ou o grupo de usuários.

JSON

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

```
  "Statement": [
    {
      "Sid": "UserPoolClientAdministrator",
      "Action": [
        "cognito-idp:CreateIdentityProvider",
        "cognito-idp:CreateManagedLoginBranding",
        "cognito-idp:CreateResourceServer",
        "cognito-idp:CreateUserPoolDomain",
        "cognito-idp:DeleteIdentityProvider",
        "cognito-idp:DeleteResourceServer",
        "cognito-idp:DeleteUserPoolDomain",
        "cognito-idp:DescribeIdentityProvider",
        "cognito-idp:DescribeManagedLoginBranding",
        "cognito-idp:DescribeManagedLoginBrandingByClient",
        "cognito-idp:DescribeResourceServer",
        "cognito-idp:DescribeUserPool",
        "cognito-idp:DescribeUserPoolClient",
        "cognito-idp:DescribeUserPoolDomain",
        "cognito-idp:GetIdentityProviderByIdentifier",
        "cognito-idp:GetUICustomization",
        "cognito-idp:ListIdentityProviders",
        "cognito-idp:ListResourceServers",
        "cognito-idp:ListUserPoolClients",
        "cognito-idp:ListUserPools",
        "cognito-idp:SetUICustomization",
        "cognito-idp:UpdateIdentityProvider",
        "cognito-idp:UpdateManagedLoginBranding",
        "cognito-idp:UpdateResourceServer",
        "cognito-idp:UpdateUserPoolClient",
        "cognito-idp:UpdateUserPoolDomain"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_EXAMPLE"
    }
  ]
}
```

O exemplo de política a seguir concede gerenciamento e autenticação de usuários e grupos a uma aplicação do lado do servidor.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "UserAdminAuthN",
      "Action": [
        "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",
        "cognito-idp:AdminSetUserPassword",
        "cognito-idp:AdminSetUserSettings",
        "cognito-idp:AdminUpdateAuthEventFeedback",
        "cognito-idp:AdminUpdateDeviceStatus",
        "cognito-idp:AdminUpdateUserAttributes",
        "cognito-idp:AdminUserGlobalSignOut",
        "cognito-idp:AssociateSoftwareToken",
        "cognito-idp:ListGroups",
        "cognito-idp:ListUsers",
        "cognito-idp:ListUsersInGroup",
        "cognito-idp:RevokeToken",
        "cognito-idp:UpdateGroup",
        "cognito-idp:VerifySoftwareToken"
      ],
      "Effect": "Allow",
```

```
      "Resource": "arn:aws:cognito-idp:us-west-2:123456789012:userpool/us-
west-2_EXAMPLE"
    }
  ]
}
```

Proteger e verificar os tokens

Os tokens podem conter referências internas à associação ao grupo e aos atributos do usuário que talvez você não queira divulgar ao usuário final. Não armazene tokens de ID e acesso no armazenamento local. Os tokens de atualização são criptografados com uma chave que somente seu grupo de usuários pode acessar e não são visíveis para usuários e aplicações. Revogue os tokens de atualização quando os usuários saírem ou quando você determinar que a persistência da sessão de um usuário é indesejada por motivos de segurança.

Use tokens de acesso para autorizar o acesso somente a sistemas que verifiquem de modo independente se o token é válido e não está expirado. Para obter recursos de verificação, consulte Verificar um token web JSON.

Determinar os provedores de identidades de confiança

Ao configurar seu grupo de usuários com provedores de identidade SAML ou OIDC (IdPs), você IdPs pode criar novos usuários, definir atributos de usuário e acessar os recursos do aplicativo. Os provedores de SAML e OIDC são normalmente usados em cenários business-to-business (B2B) ou corporativos em que você ou seu cliente imediato controlam a associação e a configuração do provedor.

Os provedores sociais oferecem contas de usuário para qualquer pessoa na internet e estão menos sob seu controle do que os provedores corporativos. Ative as redes sociais IdPs em seu cliente de aplicativo somente quando estiver pronto para permitir que clientes públicos façam login e acessem recursos em seu aplicativo.

Entender o efeito dos escopos no acesso aos perfis de usuário

Você pode solicitar escopos de controle de acesso em suas solicitações de autenticação para o servidor de autorização do grupo de usuários. Esses escopos podem conceder aos usuários acesso a recursos externos e acesso para ver e modificar os próprios perfis dos usuários. Configure seus clientes de aplicação para oferecer suporte aos escopos mínimos necessários para a operação da aplicação.

O `aws.cognito.signin.user.admin` escopo está presente em todos os tokens de acesso emitidos pela autenticação do SDK com operações como InitiateAuth. Ele é designado para operações de autoatendimento de perfil de usuário em sua aplicação. Também é possível solicitar esse escopo no servidor de autorização. Esse escopo é necessário para operações autorizadas por tokens, como e. UpdateUserAttributesGetUser O efeito dessas operações é limitado pelas permissões de leitura e gravação do seu cliente da aplicação.

Os escopos `openid`, `profile`, `email` e `phone` autorizam solicitações para o endpoint userinfo em seu servidor de autorização do grupo de usuários. Eles definem os atributos que o endpoint pode retornar. O escopo `openid`, quando solicitado sem outros escopos, retorna todos os atributos disponíveis, mas quando você solicita mais escopos na solicitação, a resposta é reduzida aos atributos representados pelos escopos adicionais. O escopo `openid` também indica uma solicitação de um token de ID; quando você omite esse escopo da sua solicitação para o seu Autorizar endpoint, o Amazon Cognito emite somente um token de acesso e, quando aplicável, um token de atualização. Para obter mais informações, consulte Escopos do OpenID Connect em Termos do cliente da aplicação.

Limpar as entradas para os atributos do usuário

Os atributos do usuário que podem acabar como métodos de entrega e nomes de usuário, por exemplo `email`, têm restrições de formato. Outros atributos podem ter tipos de dados de string, booleanos ou numéricos. Os valores dos atributos de cadeia de caracteres são compatíveis com uma variedade de entradas. Configure sua aplicação para se proteger contra tentativas de gravar dados indesejados em seu diretório de usuários ou nas mensagens que o Amazon Cognito entrega aos usuários. Realize a validação do lado do cliente dos valores de atributos de string enviados pelo usuário em sua aplicação antes de enviá-los para o Amazon Cognito.

Os grupos de usuários mapeiam atributos do IdPs seu grupo de usuários com base em um mapeamento de atributos que você especifica. Mapeie somente atributos de IdP seguros e previsíveis para atributos de string do grupo de usuários.

Autenticação com grupos de usuários do Amazon Cognito

O Amazon Cognito inclui vários métodos para autenticar os usuários. Os usuários podem fazer login com WebAuthn senhas e chaves de acesso. O Amazon Cognito pode enviar a eles uma senha de uso único por e-mail ou SMS. Você pode implementar funções do Lambda que orquestram sua própria sequência de desafios e respostas. Esses são fluxos de autenticação. Nos fluxos de autenticação, os usuários fornecem um segredo e o Amazon Cognito verifica o segredo e, em

seguida, emite tokens web JSON (JWTs) para os aplicativos processarem com bibliotecas do OIDC. Neste capítulo, falaremos sobre como configurar grupos de usuários e clientes da aplicação para vários fluxos de autenticação em vários ambientes de aplicações. Você aprenderá sobre as opções para o uso das páginas de login hospedadas do login gerenciado e para criar sua própria lógica e front-end em um AWS SDK.

Todos os grupos de usuários, independentemente de você ter um domínio ou não, podem autenticar usuários na API de grupos de usuários. Se adicionar um domínio ao grupo de usuários, você poderá usar os endpoints do grupo de usuários. A API de grupos de usuários é compatível com uma variedade de modelos de autorização e fluxos de solicitações de API.

Para verificar a identidade dos usuários, o Amazon Cognito é compatível com fluxos de autenticação que incorporam tipos de desafio, além de senhas, como senhas de uso único e chaves de acesso enviadas por e-mail e SMS.

Tópicos

Implementar fluxos de autenticação

Se você está implementando o login gerenciado ou um front-end de aplicativo personalizado com um AWS SDK para autenticação, você deve configurar seu cliente de aplicativo para os tipos de autenticação que deseja implementar. As informações a seguir descrevem a configuração dos fluxos de autenticação em seus clientes da aplicação e em sua aplicação.

App client supported flows

> Você pode configurar fluxos compatíveis para seus clientes de aplicativos no console do Amazon Cognito ou com a API em um AWS SDK. Após configurar o cliente da aplicação para oferecer suporte a esses fluxos, você poderá implantá-los em sua aplicação.

O procedimento a seguir configura os fluxos de autenticação disponíveis para um cliente da aplicação com o console do Amazon Cognito.

Como configurar um cliente da aplicação para fluxos de autenticação (console)

1. Faça login AWS e navegue até o console de grupos de usuários do Amazon Cognito. Selecione um grupo de usuários ou crie um.
2. Na configuração do grupo de usuários, clique no menu Clientes da aplicação. Selecione um cliente da aplicação ou crie um.
3. Em Informações do cliente de aplicação, clique em Editar.
4. Em Fluxos do cliente da aplicação, escolha os fluxos de autenticação que você deseja oferecer suporte.

Como configurar um cliente da aplicação para fluxos de autenticação (API/SDK)

Para configurar os fluxos de autenticação disponíveis para um cliente de aplicativo com a API do Amazon Cognito, defina o valor de `ExplicitAuthFlows` em uma solicitação CreateUserPoolClientor UpdateUserPoolClient. Veja a seguir um exemplo que fornece senha remota segura (SRP) e autenticação baseada em opções para um cliente.

```
"ExplicitAuthFlows": [
   "ALLOW_USER_AUTH",
   "ALLOW_USER_SRP_AUTH
]
```

Ao configurar fluxos compatíveis com o cliente da aplicação, você poderá especificar os valores da API e as opções abaixo.

Suporte ao fluxo do cliente da aplicação

Fluxo de autentica ção	Compatibilidade	Console	solicitações de
Autenticação baseada em opções	Lado do servidor, lado do cliente	Selecionar um tipo de autenticação no login	`ALLOW_USE R_AUTH`

Fluxo de autenticação	Compatibilidade	Console	solicitações de
Fazer login com senhas persistentes	Lado do cliente	Fazer login com nome de usuário e senha	`ALLOW_USER_PASSWORD_AUTH`
Fazer login com senhas persistentes e carga útil segura	Lado do servidor, lado do cliente	Fazer login com senha remota segura (SRP)	`ALLOW_USER_SRP_AUTH`
Atualizar tokens	Lado do servidor, lado do cliente	Receber novos tokens de usuário de sessões autenticadas existentes	`ALLOW_REFRESH_TOKEN_AUTH`
Autenticação no lado do servidor	Lado do servidor	Fazer login com credenciais administrativas do lado do servidor	`ALLOW_ADMIN_USER_PASSWORD_AUTH`
Autenticação personalizada	Aplicações personalizadas dos lados do servidor e do cliente. Não é compatível com o login gerenciado.	Fazer login com fluxos de autenticação personalizados dos acionadores do Lambda	`ALLOW_CUSTOM_AUTH`

Implement flows in your application

O login gerenciado disponibiliza automaticamente as opções de autenticação configuradas em suas páginas de login. Em aplicações personalizadas, inicie a autenticação com uma declaração do fluxo inicial.

- Para escolher entre uma lista de opções de fluxo para um usuário, declare a autenticação baseada em opções com o fluxo `USER_AUTH`. Esse fluxo tem métodos de autenticação disponíveis que não estão disponíveis nos fluxos de autenticação baseada em clientes, por exemplo, autenticação por chave de acesso e autenticação sem senha.

- Para escolher seu fluxo de autenticação com antecedência, declare a autenticação baseada em clientes juntamente com qualquer outro fluxo disponível no cliente da aplicação.

Quando você faz login com usuários, o corpo da sua AdminInitiateAuthsolicitação InitiateAuthou solicitação deve incluir um `AuthFlow` parâmetro.

Autenticação baseada em opções:

```
"AuthFlow": "USER_AUTH"
```

Autenticação baseada em clientes com SRP:

```
"AuthFlow": "USER_SRP_AUTH"
```

Coisas a saber sobre a autenticação com grupos de usuários

Considere as informações a seguir no design do modelo de autenticação com grupos de usuários do Amazon Cognito.

Fluxos de autenticação no login gerenciado e na IU hospedada

O login gerenciado tem mais opções de autenticação do que a IU hospedada clássica. Por exemplo, os usuários podem fazer autenticação sem senha e com chave de acesso somente no login gerenciado.

Fluxos de autenticação personalizados disponíveis somente na autenticação AWS do SDK

Não é possível criar fluxos de autenticação personalizados, nem autenticação personalizada com acionadores do Lambda, usando o login gerenciado ou a IU hospedada clássica. A autenticação personalizada está disponível na autenticação com AWS SDKs.

Login gerenciado para login do provedor de identidades (IdP) externo

Você não pode fazer login de usuários por meio de terceiros IdPs na autenticação com AWS SDKs. Você deve implementar o login gerenciado ou a interface de usuário hospedada clássica, redirecionar IdPs e processar o objeto de autenticação resultante com as bibliotecas do OIDC em seu aplicativo. Para obter mais informações sobre o login gerenciado, consulte Login gerenciado do grupo de usuários.

Efeito da autenticação sem senha em outros recursos do usuário

A ativação do login sem senha com senhas de uso único ou chaves de acesso no grupo de usuários e no cliente da aplicação afeta a criação e a migração de usuários. Quando o login sem senha está ativo:

1. Os administradores podem criar usuários sem senhas. O modelo de mensagem de convite padrão é alterado para não incluir mais o espaço reservado para senha `{###}`. Para obter mais informações, consulte Como criar contas de usuário como administrador.
2. Para SignUpoperações baseadas em SDK, os usuários não precisam fornecer uma senha ao se inscreverem. O login gerenciado e a IU hospedada exigem uma senha na página de cadastro, mesmo que a autenticação sem senha seja permitida. Para obter mais informações, consulte Como cadastrar e confirmar contas de usuários.
3. Os usuários importados de um arquivo CSV podem fazer login imediatamente com opções sem senha, sem a necessidade de redefinição de senha, se seus atributos incluírem um endereço de e-mail ou número de telefone para uma opção de login sem senha disponível. Para obter mais informações, consulte Como importar usuários para grupos de usuários com base em um arquivo CSV.
4. A autenticação sem senha não invoca o acionador do Lambda de migração de usuários.
5. Os usuários que fazem login com um primeiro fator de senha de uso único (OTP) não podem adicionar um fator de autenticação multifator (MFA) à sessão. As chaves de acesso com verificação do usuário podem atender aos requisitos de MFA quando configuradas com. `MULTI_FACTOR_WITH_USER_VERIFICATION`

A parte confiável da chave de acesso não URLs pode estar na lista pública de sufixos

Você pode usar nomes de domínio que você possui, como `www.example.com`, como o ID de parte confiável (RP) na configuração da chave de acesso. Essa configuração se destina a oferecer suporte a aplicações personalizadas executadas em domínios que você possui. A lista de sufixos públicos, ou PSL, contém domínios de alto nível protegidos. O Amazon Cognito retorna um erro quando você tenta definir o URL de RP como um domínio na PSL.

Tópicos

- Duração do fluxo da sessão de autenticação
- Comportamento de bloqueio em tentativas fracassadas de login

Duração do fluxo da sessão de autenticação

Dependendo dos recursos do grupo de usuários, você pode acabar respondendo a vários desafios para `InitiateAuth` e `RespondToAuthChallenge` antes da aplicação recuperar tokens do Amazon Cognito. O Amazon Cognito inclui uma string de sessão na resposta a cada solicitação. Para combinar suas solicitações de API em um fluxo de autenticação, inclua a string da sessão da resposta à solicitação anterior em cada solicitação subsequente. Por padrão, os usuários têm três minutos para concluir cada desafio antes que a string da sessão expire. Para ajustar esse período, altere o cliente da aplicação Authentication flow session duration (Duração da sessão do fluxo de autenticação). O procedimento a seguir descreve como alterar essa definição na configuração do cliente da aplicação.

> **Note**
>
> As configurações de duração da sessão do fluxo de autenticação se aplicam à autenticação com a API de grupos de usuários do Amazon Cognito. O login gerenciado define a duração da sessão como 3 minutos para autenticação multifator e 8 minutos para códigos de redefinição de senha.

Amazon Cognito console

Como configurar a duração da sessão do fluxo de autenticação do cliente da aplicação (Console de gerenciamento da AWS)

1. Na guia App integration (Integração de aplicações) no grupo de usuários, selecione o nome do cliente da aplicação no contêiner App clients and analytics (Clientes e análise de aplicações).
2. Selecione Editar no contêiner Informações do cliente da aplicação.
3. Altere o valor de Duração da sessão de fluxo de autenticação para a duração de validade desejada, em minutos, para códigos de MFA por e-mail ou SMS. Isso também altera a quantidade de tempo que qualquer usuário tem para concluir qualquer desafio de autenticação no cliente da aplicação.
4. Escolha Salvar alterações.

User pools API

Como configurar a duração da sessão do fluxo de autenticação do cliente da aplicação (API do Amazon Cognito)

1. Prepare uma solicitação `UpdateUserPoolClient` com as configurações existentes de seu grupo de usuários usando uma solicitação `DescribeUserPoolClient`. A solicitação `UpdateUserPoolClient` deve incluir todas as propriedades existentes do cliente da aplicação.
2. Altere o valor de `AuthSessionValidity` para a duração de validade desejada, em minutos, para códigos de MFA por SMS. Isso também altera a quantidade de tempo que qualquer usuário tem para concluir qualquer desafio de autenticação no cliente da aplicação.

Para obter mais informações sobre clientes de aplicação, consulte Configurações específicas da aplicação com clientes de aplicação.

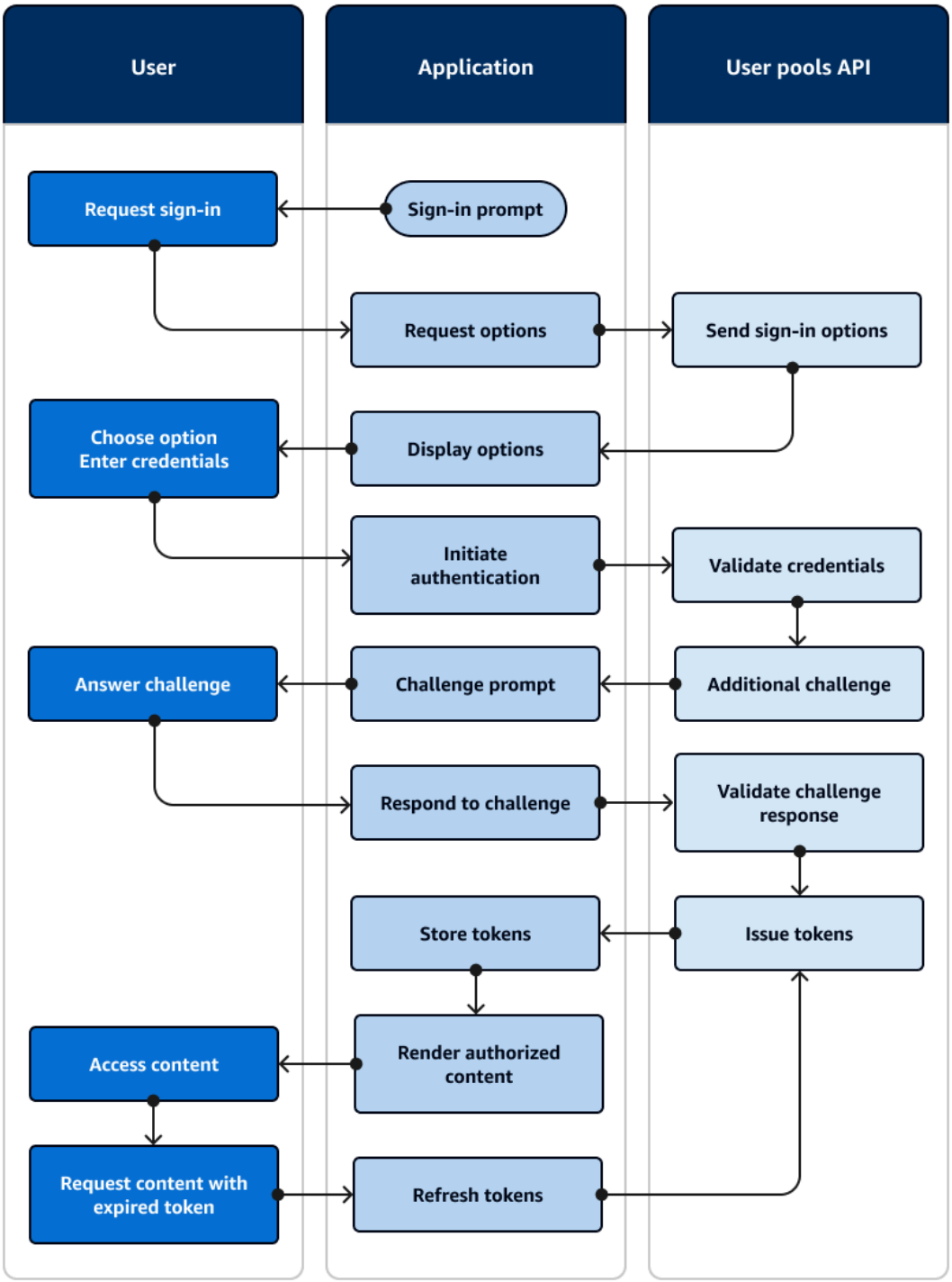
Comportamento de bloqueio em tentativas fracassadas de login

Após cinco tentativas de login com falha com a senha do usuário, independentemente de serem solicitadas com operações de API não autenticadas ou autorizadas pelo IAM, o Amazon Cognito bloqueia o usuário por 1 segundo. A duração do bloqueio dobra após cada tentativa adicional fracassada, até um máximo de aproximadamente 15 minutos.

As tentativas feitas durante um período de bloqueio geram uma exceção `Password attempts exceeded` e não afetam a duração dos períodos de bloqueio subsequentes. Para um número cumulativo de tentativas fracassadas de login n, sem incluir exceções `Password attempts exceeded`, o Amazon Cognito bloqueia o usuário por 2^(n-5) segundos. Para redefinir o bloqueio como o estado inicial n=0, o usuário deve fazer login com êxito após o término do período de bloqueio ou não iniciar nenhuma tentativa de login por 15 minutos consecutivos a qualquer momento após um bloqueio. Esse comportamento está sujeito a alterações. Esse comportamento não se aplica aos desafios personalizados, a menos que eles também realizem a autenticação baseada em senha.

Um exemplo de sessão de autenticação

O diagrama e o step-by-step guia a seguir ilustram um cenário típico em que um usuário faz login em um aplicativo. A aplicação de exemplo apresenta ao usuário várias opções de login. Ele seleciona uma inserindo suas credenciais, fornece um fator de autenticação adicional e faz login.



Imagine uma aplicação com uma página de login na qual os usuários podem fazer login com nome de usuário e senha, solicitar um código de uso único enviado por e-mail ou escolher uma opção de impressão digital.

1. Solicitação de login: a aplicação mostra uma tela inicial com um botão Fazer login.
2. Solicitar login: o usuário seleciona Fazer login. Com base em um cookie ou cache, a aplicação recupera o nome de usuário ou solicita que ele o insira.
3. Opções de solicitação: a aplicação solicita as opções de login do usuário por meio de uma solicitação de API `InitiateAuth` com o fluxo `USER_AUTH`, solicitando os métodos de login disponíveis para o usuário.
4. Enviar opções de login: o Amazon Cognito responde com `PASSWORD`, `EMAIL_OTP` e `WEB_AUTHN`. A resposta inclui um identificador de sessão para você reproduzir na próxima resposta.
5. Opções de exibição: a aplicação mostra elementos de IU para que o usuário insira seu nome de usuário e senha, obtenha um código de uso único ou escaneie sua impressão digital.
6. Escolha option/Enter as credenciais: o usuário insere seu nome de usuário e senha.
7. Iniciar autenticação: a aplicação fornece as informações de login do usuário por meio de uma solicitação de API `RespondToAuthChallenge` que confirma o login com nome de usuário e senha e fornece o nome de usuário e a senha.
8. Validar credenciais: o Amazon Cognito confirma as credenciais do usuário.
9. Desafio adicional: o usuário tem a autenticação multifator configurada com uma aplicação autenticadora. O Amazon Cognito retorna um desafio `SOFTWARE_TOKEN_MFA`.
10. Solicitação de desafio: a aplicação exibe um formulário solicitando uma senha de uso único com marcação temporal (TOTP) da aplicação autenticadora do usuário.
11. Responder ao desafio: o usuário envia a TOTP.
12. Responder ao desafio: em outra solicitação `RespondToAuthChallenge`, a aplicação fornece a TOTP do usuário.
13. Validar a resposta ao desafio: o Amazon Cognito confirma o código do usuário e determina que o grupo de usuários está configurado para não emitir desafios adicionais para o usuário atual.
14. Emitir tokens: o Amazon Cognito retorna tokens web JSON de ID, acesso e atualização (). JWTs A autenticação inicial do usuário está concluída.
15. Armazenar tokens: a aplicação armazena em cache os tokens do usuário para poder referenciar os dados do usuário, autorizar o acesso a recursos e atualizar os tokens quando eles expirarem.
16. Renderizar conteúdo autorizado: a aplicação determina o acesso do usuário aos recursos com base em sua identidade e funções e fornece o conteúdo da aplicação.
17. Acessar conteúdo: o usuário está conectado e começa a usar a aplicação.
18. Solicitar conteúdo com token expirado: posteriormente, o usuário solicita um recurso que requer autorização. O token em cache do usuário expirou.

19. Tokens de atualização: a aplicação faz uma solicitação `InitiateAuth` com o token de atualização salvo do usuário.
20. Emitir tokens: o Amazon Cognito retorna novo ID e acesso. JWTs A sessão do usuário é atualizada com segurança sem solicitações adicionais de credenciais.

Você pode usar acionadores do AWS Lambda para personalizar a maneira como os usuários se autenticam. Esses triggers emitem e verificam seus próprios desafios como parte do fluxo de autenticação.

Também é possível usar o fluxo de autenticação de administrador para servidores de backend seguros. É possível usar o fluxo de autenticação de migração do usuário para permitir essa migração sem exigir que os usuários redefinam suas senhas.

Configurar métodos de autenticação para login gerenciado

Você pode invocar páginas de login gerenciado, um frontend da web para autenticação de grupos de usuários, quando quiser que os usuários façam login, logout ou redefinam suas senhas. Nesse modelo, a aplicação importa bibliotecas do OIDC para processar tentativas de autenticação baseadas em navegador com páginas de login gerenciado de grupos de usuários. As formas de autenticação disponíveis para os usuários dependem da configuração do grupo de usuários e do cliente da aplicação. Implemente o fluxo `ALLOW_USER_AUTH` no cliente da aplicação e o Amazon Cognito solicitará que os usuários selecionem um método de login entre as opções disponíveis. Implemente `ALLOW_USER_PASSWORD_AUTH` e atribua um provedor SAML e as páginas de login solicitarão aos usuários a opção de inserir seu nome de usuário e senha ou de se conectar ao IdP.

O console de grupos de usuários do Amazon Cognito pode auxiliar na configuração da autenticação de login gerenciado para sua aplicação. Ao criar um novo grupo de usuários, especifique a plataforma para a qual você está desenvolvendo e o console fornece exemplos de implementação de OIDC e OAuth bibliotecas com código inicial para implementar fluxos de entrada e saída. Você pode criar login gerenciado com diversas implementações de partes confiáveis do OIDC. Recomendamos que você trabalhe com bibliotecas de partes confiáveis do OIDC certificadas sempre que possível. Para obter mais informações, consulte Conceitos básicos dos grupos de usuários.

Normalmente, as bibliotecas confiáveis do OIDC verificam periodicamente o `.well-known/openid-configuration` endpoint do seu grupo de usuários para determinar o emissor, URLs como o endpoint do token e o endpoint da autorização. Como prática recomendada, implemente esse comportamento de descoberta automática sempre que possível. A configuração manual dos

endpoints do emissor apresenta potencial de erro. Por exemplo, você pode alterar o domínio do grupo de usuários. O caminho para `openid-configuration` não está vinculado ao domínio do grupo de usuários, portanto, as aplicações que descobrem automaticamente os endpoints de serviço detectarão automaticamente a alteração do domínio.

Configurações do grupo de usuários para login gerenciado

Recomenda-se permitir o login com vários provedores para sua aplicação ou usar o Amazon Cognito como um diretório de usuários independente. É recomendável também coletar atributos do usuário, configurar e solicitar a MFA ou exigir endereços de e-mail como nomes de usuário. Não é possível editar diretamente os campos no login gerenciado e na IU hospedada. Em vez disso, a configuração do grupo de usuários define automaticamente o tratamento dos fluxos de autenticação de login gerenciado.

Os itens de configuração do grupo de usuários a seguir determinam os métodos de autenticação que o Amazon Cognito apresenta aos usuários no login gerenciado e na IU hospedada.

User pool options (Sign-in menu)

As opções a seguir estão no menu Fazer login de um grupo de usuários no console do Amazon Cognito.

Opções de login do grupo de usuários do Cognito

Tem opções para nomes de usuário. Suas páginas de login gerenciado e IU hospedada aceitam somente os nomes de usuário nos formatos que você selecionar. Por exemplo, ao configurar um grupo de usuários com e-mail como a única opção de login, as páginas de login gerenciado só aceitarão nomes de usuário em formato de e-mail.

Atributos obrigatórios

Ao definir um atributo como obrigatório no grupo de usuários, o login gerenciado solicita aos usuários um valor para esse atributo no momento do cadastro.

Opções para login baseado em opções

Tem configurações para métodos de autenticação em Autenticação baseada em opções. Aqui, você pode ativar ou desativar métodos de autenticação, como chave de acesso e sem senha. Esses métodos estão disponíveis somente para grupos de usuários com domínios de login gerenciado e planos de recursos acima do nível Lite.

Autenticação multifator

O login gerenciado e a IU hospedada lidam com as operações de registro e autenticação para MFA. Quando a MFA é obrigatória no grupo de usuários, as páginas de login solicitam automaticamente que os usuários configurem seu fator adicional. Elas também solicitam que os usuários com MFA configurada concluam a autenticação com um código de MFA. Quando a MFA está desativada ou opcional no grupo de usuários, as páginas de login não solicitam a configuração da MFA.

Recuperação de contas de usuários

A configuração de recuperação de contas por autoatendimento do grupo de usuários determina se as páginas de login exibem um link no qual os usuários podem redefinir suas senhas.

User pool options (Domain menu)

As opções a seguir estão no menu Domínio de um grupo de usuários no console do Amazon Cognito.

Domínio

Sua escolha de domínio do grupo de usuários define o caminho para o link que os usuários abrem quando você invoca seus navegadores para autenticação.

Versão de marca

Sua escolha de uma versão de marca define se o domínio do grupo de usuários exibe o login gerenciado ou a IU hospedada.

User pool options (Social and external providers menu)

A opção a seguir está no menu Provedores sociais e externos de um grupo de usuários no console do Amazon Cognito.

Provedores

Os provedores de identidade (IdPs) que você adiciona ao seu grupo de usuários podem ficar ativos ou inativos para cada cliente de aplicativo no grupo de usuários.

App client options

As opções a seguir estão no menu Clientes da aplicação de um grupo de usuários no console do Amazon Cognito. Para analisar essas opções, selecione um cliente da aplicação na lista.

Guia de configuração rápida

O guia de configuração rápida tem exemplos de código para diversos ambientes de desenvolvimento. Ele inclui as bibliotecas necessárias para integrar a autenticação de login gerenciado à sua aplicação.

Informações do cliente da aplicação

Edite essa configuração para definir atribuída IdPs ao aplicativo que é representado pelo cliente do aplicativo atual. Nas páginas de login gerenciado, o Amazon Cognito exibe opções para os usuários. Essas opções são determinadas pelos métodos e IdP atribuídos. Por exemplo, se você atribuir um IdP SAML 2.0 denominado `MySAML` e um login de grupo de usuários local, as páginas de login gerenciado exibirão solicitações de método de autenticação e um botão para `MySAML`.

Configurações de Autenticação

Edite esta configuração para definir métodos de autenticação para a aplicação. Nas páginas de login gerenciado, o Amazon Cognito exibe opções para os usuários. Essas opções são determinadas pela disponibilidade do grupo de usuários como um IdP e pelos métodos que você atribui. Por exemplo, se você atribuir a autenticação `ALLOW_USER_AUTH` baseada em opções, as páginas de login gerenciado exibirão as opções disponíveis, como inserir um endereço de e-mail e fazer login com uma chave de acesso. As páginas de login gerenciadas também renderizam botões para os atribuídos IdPs.

Páginas de login

Defina o efeito visual das páginas interativas de login gerenciado ou da IU hospedada com as opções disponíveis nesta guia. Para obter mais informações, consulte Aplicar a identidade visual às páginas de login gerenciado.

Gerencie métodos de autenticação em AWS SDKs

Os usuários nos grupos de usuários do Amazon Cognito podem fazer login com uma variedade de opções de login inicial, ou fatores. Para alguns fatores, os usuários podem complementar com a autenticação multifator (MFA). Esses primeiros fatores incluem nome de usuário e senha, senha de uso único, chave de acesso e autenticação personalizada. Para obter mais informações, consulte Fluxos de autenticação. Quando seu aplicativo tem componentes de interface de usuário integrados e importa um módulo AWS SDK, você deve criar a lógica do aplicativo para autenticação. Você deve

escolher um dos dois métodos principais e, a partir desse método, os mecanismos de autenticação que deseja implementar.

Você pode implementar a autenticação baseada em clientes, onde sua aplicação, ou cliente, declara o tipo de autenticação com antecedência. A outra opção é a autenticação baseada em opções, em que a aplicação coleta um nome de usuário e solicita os tipos de autenticação disponíveis para os usuários. Você pode implementar esses modelos juntos no mesma aplicação ou dividi-los entre clientes da aplicação, de acordo com seus requisitos. Cada método tem recursos exclusivos, como autenticação personalizada na autenticação baseada em clientes e autenticação sem senha na autenticação baseada em opções.

Em aplicativos personalizados que realizam autenticação com a implementação do AWS SDK da API de grupos de usuários, você deve estruturar suas solicitações de API de acordo com a configuração do grupo de usuários, a configuração do cliente do aplicativo e as preferências do lado do cliente. Uma sessão `InitiateAuth` que começa com um `AuthFlow` de `USER_AUTH` inicia a autenticação baseada em opções. O Amazon Cognito responde à API com um desafio que consiste em um método de autenticação preferencial ou uma lista de opções. Uma sessão que começa com `AuthFlow` de `CUSTOM_AUTH` inicia diretamente a autenticação personalizada com acionadores do Lambda.

Alguns métodos de autenticação são fixos em um dos dois tipos de fluxo, e alguns métodos estão disponíveis em ambos.

Tópicos

Autenticação baseada em opções

Sua aplicação pode solicitar os métodos de autenticação a seguir na autenticação baseada em opções. Declare essas opções no `PREFERRED_CHALLENGE` parâmetro de InitiateAuthou AdminInitiateAuth, ou no `ChallengeName` parâmetro de RespondToAuthChallengeou AdminRespondToAuthChallenge.

1. `EMAIL_OTP` e `SMS_OTP`

   Fazer login sem senha com senhas de uso único
2. `WEB_AUTHN`

Login sem senha com chaves de acesso WebAuthn

3. PASSWORD

   Fazer login com senhas persistentes

   Fazer login com senhas persistentes e carga útil segura

   MFA após o login

Para revisar essas opções em seu contexto de API, consulte `ChallengeName` em RespondToAuthChallenge.

O login baseado em opções gera um desafio em resposta à solicitação inicial. Esse desafio verifica se a opção solicitada está disponível ou fornece uma lista das opções disponíveis. Sua aplicação pode exibir essas opções para os usuários, que então inserem as credenciais do método de login preferencial e prosseguem com a autenticação nas respostas do desafio.

Você tem as seguintes opções de autenticação baseada em opções em seu fluxo de autenticação. Todas as solicitações desse tipo exigem que sua aplicação primeiro colete um nome de usuário ou o recupere de um cache.

1. Solicite opções somente com `AuthParameters` de `USERNAME`. O Amazon Cognito retorna um desafio `SELECT_CHALLENGE`. A partir daí, sua aplicação pode solicitar que o usuário selecione um desafio e então retornar essa resposta ao grupo de usuários.
2. Solicite um desafio preferencial com `AuthParameters` de `PREFERRED_CHALLENGE` e os parâmetros de seu desafio preferencial, se houver. Por exemplo, se você solicitar um `PREFERRED_CHALLENGE` de `PASSWORD_SRP`, também deverá incluir `SRP_A`. Se seu usuário, grupo de usuários e cliente do aplicativo estiverem todos configurados para o desafio preferido, o Amazon Cognito responderá com a próxima etapa desse desafio, por exemplo, `PASSWORD_VERIFIER` no `PASSWORD_SRP` fluxo ou CodeDeliveryDetailsnos `EMAIL_OTP` fluxos e. `SMS_OTP` Se o desafio preferencial não estiver disponível, o Amazon Cognito responderá com `SELECT_CHALLENGE` e uma lista dos desafios disponíveis.
3. Primeiro, faça o login dos usuários e, em seguida, solicite suas opções de autenticação baseada em opções. Uma GetUserAuthFactorssolicitação com o token de acesso de um usuário conectado retorna seus fatores de autenticação baseados em opções disponíveis e suas configurações de MFA. Com essa opção, um usuário pode primeiro fazer login com nome de usuário e senha e

depois ativar uma forma diferente de autenticação. Também é possível usar essa operação para verificar opções adicionais para um usuário que tenha feito login com um desafio preferencial.

Para configurar o cliente da aplicação para autenticação baseada em opções, adicione `ALLOW_USER_AUTH` aos fluxos de autenticação permitidos. Você também deve escolher os fatores baseados em opções que deseja permitir na configuração do grupo de usuários. O processo a seguir ilustra como escolher os fatores da autenticação baseada em opções.

Amazon Cognito console

Como configurar opções de autenticação baseada em opções em um grupo de usuários

1. Faça login AWS e navegue até o console de grupos de usuários do Amazon Cognito. Selecione um grupo de usuários ou crie um.
2. Na configuração do grupo de usuários, clique no menu Fazer login. Localize Opções para login baseado em opções e clique em Editar.
3. A opção Senha está sempre disponível. Isso inclui os fluxos `PASSWORD` e `PASSWORD_SRP`. Selecione as Opções adicionais que deseja adicionar às opções dos usuários. Você pode adicionar Chave de acesso para `WEB_AUTHN`, Senha única para mensagem de e-mail para `EMAIL_OTP` e Senha única para mensagem SMS para `SMS_OTP`.
4. Escolha Salvar alterações.

API/SDK

O corpo parcial CreateUserPoolou de UpdateUserPoolsolicitação a seguir configura todas as opções disponíveis para autenticação baseada em opções.

```
"Policies": {
    "SignInPolicy": {
        "AllowedFirstAuthFactors": [
            "PASSWORD",
            "WEB_AUTHN",
            "EMAIL_OTP",
            "SMS_OTP"
        ]
    }
},
```

Autenticação baseada em clientes

A autenticação baseada em clientes é compatível com os fluxos de autenticação a seguir. Declare essas opções no `AuthFlow` parâmetro de InitiateAuthou AdminInitiateAuth.

1. `USER_PASSWORD_AUTH` e `ADMIN_USER_PASSWORD_AUTH`

   Fazer login com senhas persistentes

   MFA após o login

   Este fluxo de autenticação é equivalente a `PASSWORD` na autenticação baseada em opções.
2. `USER_SRP_AUTH`

   Fazer login com senhas persistentes e carga útil segura

   MFA após o login

   Este fluxo de autenticação é equivalente a `PASSWORD_SRP` na autenticação baseada em opções.
3. `REFRESH_TOKEN_AUTH`

   Tokens de atualização

   Este fluxo de autenticação só está disponível na autenticação baseada em clientes.
4. `CUSTOM_AUTH`

   Autenticação personalizada

   Este fluxo de autenticação só está disponível na autenticação baseada em clientes.

Com a autenticação baseada em clientes, o Amazon Cognito presume que você determinou como o usuário deseja se autenticar antes de iniciar os fluxos de autenticação. A lógica para determinar o fator de login que um usuário deseja fornecer deve ser determinada com configurações padrão ou solicitações personalizadas e, em seguida, declarada na primeira solicitação ao grupo de usuários. A solicitação `InitiateAuth` declara um `AuthFlow` de login que corresponde diretamente a uma das opções listadas, por exemplo, `USER_SRP_AUTH`. Com essa declaração, a solicitação também inclui os parâmetros para iniciar a autenticação, por exemplo, `USERNAME`, `SECRET_HASH` e `SRP_A`. O Amazon Cognito pode acompanhar essa solicitação com desafios adicionais, como

`PASSWORD_VERIFIER` para SRP ou `SOFTWARE_TOKEN_MFA` para login por senha com MFA com TOTP.

Para configurar o cliente da aplicação para autenticação baseada em clientes, adicione quaisquer fluxos de autenticação diferentes de `ALLOW_USER_AUTH` aos fluxos de autenticação permitidos. Os exemplos são`ALLOW_USER_PASSWORD_AUTH`,`ALLOW_CUSTOM_AUTH`,`ALLOW_REFRESH_TOKEN_AUTH`. Para permitir fluxos de autenticação baseada em clientes, nenhuma configuração adicional do grupo de usuários é obrigatória.

Fluxos de autenticação

O processo de autenticação com grupos de usuários do Amazon Cognito pode ser melhor descrito como um fluxo no qual que os usuários fazem uma escolha inicial, enviam credenciais e respondem a desafios adicionais. Quando você implementa a autenticação de login gerenciado na sua aplicação, o Amazon Cognito gerencia o fluxo dessas solicitações e desafios. Ao implementar fluxos com um AWS SDK no back-end do seu aplicativo, você deve criar a lógica das solicitações, solicitar que os usuários forneçam informações e responder aos desafios.

Como administrador da aplicação, as características do usuário, os requisitos de segurança e o modelo de autorização ajudam a determinar como você deseja permitir que os usuários façam login. Pergunte-se as questões a seguir.

- Quero permitir que os usuários façam login com credenciais de outros provedores de identidade (IdPs)?
- Um nome de usuário e senha são provas suficientes de identidade?
- Minhas solicitações de autenticação por nome de usuário e senha poderiam ser interceptadas? Quero que minha aplicação transmita senhas ou negocie a autenticação usando hashes e salts?
- Quero permitir que os usuários ignorem a inserção de senha e recebam uma senha de uso único para fazer login?
- Quero permitir que os usuários façam login com impressão digital, detecção facial ou chave de segurança de hardware?
- Quando devo exigir a autenticação multifator (MFA), se é que devo?
- Quero manter as sessões dos usuários sem solicitar novamente as credenciais?
- Quero estender meu modelo de autorização além dos recursos integrados do Amazon Cognito?

Quando tiver as respostas para essas perguntas, poderá aprender como ativar os recursos relevantes e implementá-los nas solicitações de autenticação que sua aplicação realiza.

Depois de configurar os fluxos de login para um usuário, você pode verificar o status atual do MFA e dos fatores de autenticação com base em escolhas com solicitações para a operação da API. GetUserAuthFactors Essa operação requer autorização com o token de acesso de um usuário conectado. Ela retorna os fatores de autenticação do usuário e as configurações de MFA.

Tópicos

- Faça login com terceiros IdPs
- Fazer login com senhas persistentes
- Fazer login com senhas persistentes e carga útil segura
- Fazer login sem senha com senhas de uso único
- Login sem senha com chaves de acesso WebAuthn
- MFA após o login
- Tokens de atualização
- Autenticação personalizada
- Fluxo de autenticação de migração de usuários

Faça login com terceiros IdPs

Os grupos de usuários do Amazon Cognito servem como intermediários de sessões de autenticação entre serviços IdPs como Sign in with Apple, Login with Amazon e OpenID Connect (OIDC). Esse processo também é chamado de login federado ou autenticação federada. A autenticação federada não usa nenhum dos fluxos de autenticação que você pode implementar no cliente da aplicação. Em vez disso, você atribui um grupo de usuários configurado IdPs ao seu cliente de aplicativo. O login federado ocorre quando os usuários selecionam seu IdP no login gerenciado ou sua aplicação invoca uma sessão com um redirecionamento para a página de login do IdP.

Com o login federado, você delega fatores de autenticação principal e de MFA ao IdP do usuário. O Amazon Cognito não adiciona os outros fluxos avançados desta seção a um usuário federado, a menos que você os vincule a um usuário local. Usuários federados não vinculados possuem nomes de usuário, mas eles são um repositório de dados de atributos mapeados que normalmente não são usados para login, independentemente do fluxo baseado em navegador.

Recursos de implementação

- Login do grupo de usuários com provedores de identidades de terceiros

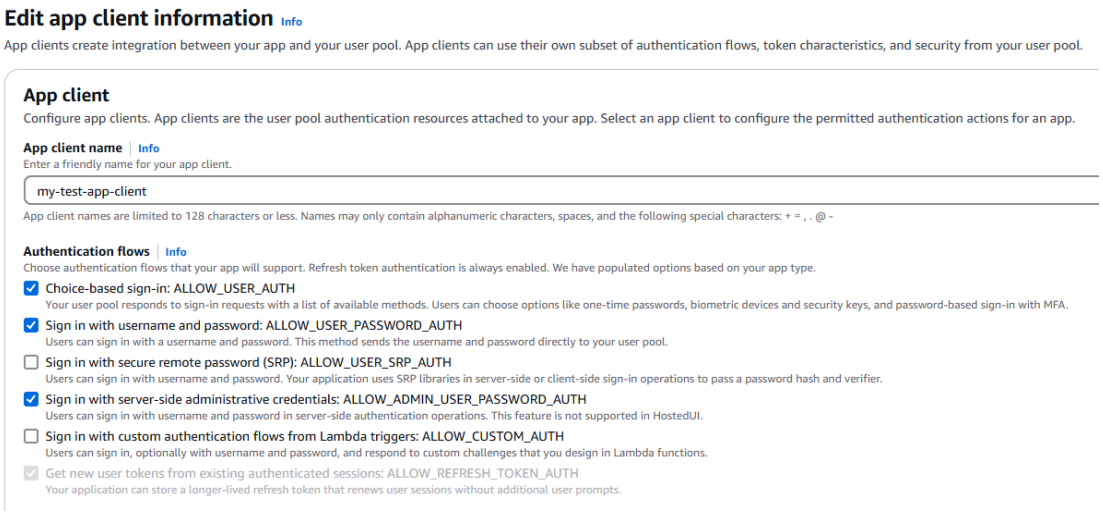
Fazer login com senhas persistentes

Nos grupos de usuários do Amazon Cognito, cada usuário tem um nome de usuário. Pode ser um número de telefone, endereço de e-mail ou um identificador escolhido ou fornecido pelo administrador. Usuários desse tipo podem fazer login com seu nome de usuário e senha e, opcionalmente, fornecer MFA. Grupos de usuários podem realizar login com nome de usuário e senha com operações de API públicas ou autorizadas pelo IAM e métodos de SDK. A aplicação pode enviar diretamente a senha ao grupo de usuários para autenticação. Seu grupo de usuários responde com desafios adicionais ou com os tokens web JSON (JWTs) que são o resultado de uma autenticação bem-sucedida.

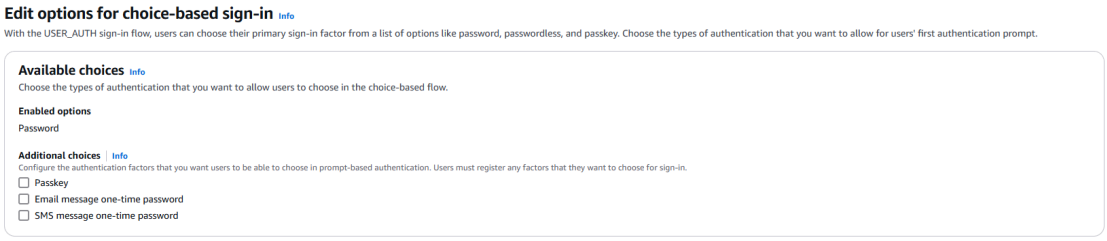
Activate password sign-in

Para ativar a autenticação baseada em clientes com nome de usuário e senha, configure o cliente da aplicação para permitir isso. No console do Amazon Cognito, navegue até o menu Clientes da aplicação em Aplicações na configuração do grupo de usuários. Para permitir o login com senha simples em uma aplicação nativa ou um aplicativo móvel do lado do cliente, edite um cliente da aplicação e selecione Fazer login com nome de usuário e senha: ALLOW_USER_PASSWORD_AUTH em Fluxos de autenticação. Para permitir o login com senha simples em uma aplicação do lado do servidor, edite um cliente da aplicação e clique em Fazer login com credenciais administrativas do lado do servidor: ALLOW_ADMIN_USER_PASSWORD_AUTH.

Para ativar a autenticação baseada em opções com nome de usuário e senha, configure o cliente da aplicação para permitir isso. Edite o cliente da aplicação e selecione Login baseado em opções: ALLOW_USER_AUTH.



Para verificar se a autenticação por senha está disponível em fluxos de autenticação baseada em opções, navegue até o menu Fazer login e revise a seção em Opções para login baseado em opções. Você pode fazer login com autenticação por senha simples se a senha estiver visível em Opções disponíveis. A opção Senha inclui as variantes simples e SRP da autenticação por nome de usuário e senha.



Configure `ExplicitAuthFlows` com suas opções username-and-password de autenticação preferidas em uma UpdateUserPoolClientsolicitação CreateUserPoolClientor.

```
"ExplicitAuthFlows": [
   "ALLOW_USER_PASSWORD_AUTH",
   "ALLOW_ADMIN_USER_PASSWORD_AUTH",
   "ALLOW_USER_AUTH"
]
```

Em uma UpdateUserPoolsolicitação CreateUserPoolor, configure `Policies` com os fluxos de autenticação com base em opções que você deseja oferecer suporte. O valor `PASSWORD` em `AllowedFirstAuthFactors` inclui as opções de fluxo de autenticação por senha simples e SRP.

```
"Policies": {
```

```
   "SignInPolicy": {
      "AllowedFirstAuthFactors": [
         "PASSWORD",
         "EMAIL_OTP",
         "WEB_AUTHN"
      ]
   }
}
```

Choice-based sign-in with a password

Para fazer login de um usuário em um aplicativo com autenticação por nome de usuário e senha, configure o corpo da sua InitiateAuthsolicitação AdminInitiateAuthou da seguinte forma. Essa solicitação de login será bem-sucedida ou continuará até o próximo desafio se o usuário atual for elegível para a autenticação por nome de usuário e senha. Caso contrário, ela responderá com uma lista de desafios de autenticação de fator primário disponíveis. Esse conjunto de parâmetros é o mínimo necessário para fazer login. Parâmetros adicionais estão disponíveis.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "PREFERRED_CHALLENGE" : "PASSWORD",
      "PASSWORD" : "[User's password]"
   },
   "ClientId": "1example23456789"
}
```

Você também pode omitir o valor `PREFERRED_CHALLENGE` e receber uma resposta contendo uma lista de fatores de login elegíveis para o usuário.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser"
   },
   "ClientId": "1example23456789"
}
```

Se você não enviou um desafio preferencial ou o usuário enviado não for elegível para o desafio preferencial, o Amazon Cognito retornará uma lista de opções em `AvailableChallenges`.

Quando `AvailableChallenges` inclui um `ChallengeName` de`PASSWORD`, você pode continuar a autenticação com uma resposta RespondToAuthChallengeou AdminRespondToAuthChallengedesafiar no formato a seguir. Você deve transmitir um parâmetro `Session` que associe a resposta do desafio à resposta da API à solicitação inicial de login. Esse conjunto de parâmetros é o mínimo necessário para fazer login. Parâmetros adicionais estão disponíveis.

```
{
   "ChallengeName": "PASSWORD",
   "ChallengeResponses": {
      "USERNAME" : "testuser",
      "PASSWORD" : "[User's Password]"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response"
}
```

O Amazon Cognito responde a solicitações de desafio preferencial elegíveis e bem-sucedidas e respostas do desafio `PASSWORD` com tokens ou um desafio adicional obrigatório, como autenticação multifator (MFA).

Client-based sign-in with a password

Para fazer login de um usuário em um aplicativo do lado do cliente com autenticação de nome de usuário e senha, configure o corpo da sua solicitação da seguinte forma. InitiateAuth Esse conjunto de parâmetros é o mínimo necessário para fazer login. Parâmetros adicionais estão disponíveis.

```
{
   "AuthFlow": "USER_PASSWORD_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "PASSWORD" : "[User's password]"
   },
   "ClientId": "1example23456789"
}
```

Para fazer login de um usuário em um aplicativo do lado do servidor com autenticação por nome de usuário e senha, configure o corpo da solicitação da seguinte maneira. AdminInitiateAuth Sua inscrição deve assinar essa solicitação com AWS as credenciais. Esse conjunto de parâmetros é o mínimo necessário para fazer login. Parâmetros adicionais estão disponíveis.

```
{
   "AuthFlow": "ADMIN_USER_PASSWORD_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "PASSWORD" : "[User's password]"
   },
   "ClientId": "1example23456789"
}
```

O Amazon Cognito responde a solicitações bem-sucedidas com tokens ou um desafio adicional obrigatório, como autenticação multifator (MFA).

Fazer login com senhas persistentes e carga útil segura

Outra forma dos métodos de login com nome de usuário e senha em grupos de usuários é com o protocolo de senha remota segura (SRP). Essa opção envia uma prova de conhecimento de uma senha (um hash de senha e um salt) que o grupo de usuários pode verificar. Sem nenhuma informação secreta legível na solicitação ao Amazon Cognito, a aplicação é a única entidade que processa as senhas inseridas pelos usuários. A autenticação SRP envolve cálculos matemáticos melhor executados por um componente existente que você pode importar no SDK. A SRP é geralmente implementada em aplicações do lado do cliente, como aplicativos móveis. Para obter mais informações sobre o protocolo, consulte The Stanford SRP Homepage. A Wikipedia também tem recursos e exemplos. Uma variedade de bibliotecas públicas estão disponíveis para realizar os cálculos de SRP para fluxos de autenticação.

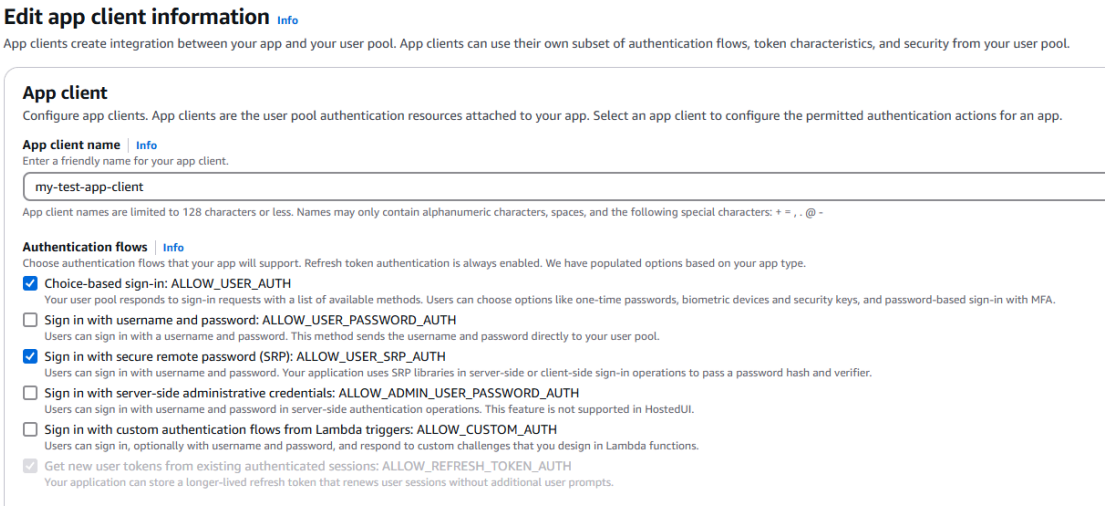
A initiate-challenge-respond sequência da autenticação do Amazon Cognito valida os usuários e suas senhas com o SRP. É necessário configurar o grupo de usuários e o cliente da aplicação para oferecer suporte à autenticação SRP e, em seguida, implementar a lógica das solicitações de login e das respostas do desafio na aplicação. Suas bibliotecas de SRP podem gerar números aleatórios e valores calculados que demonstram ao grupo de usuários que você possui a senha de um usuário. Sua aplicação preenche esses valores calculados nos campos `AuthParameters` e `ChallengeParameters` formatados em JSON e nas operações de API e métodos de SDK para autenticação de grupos de usuários do Amazon Cognito.

Activate SRP sign-in

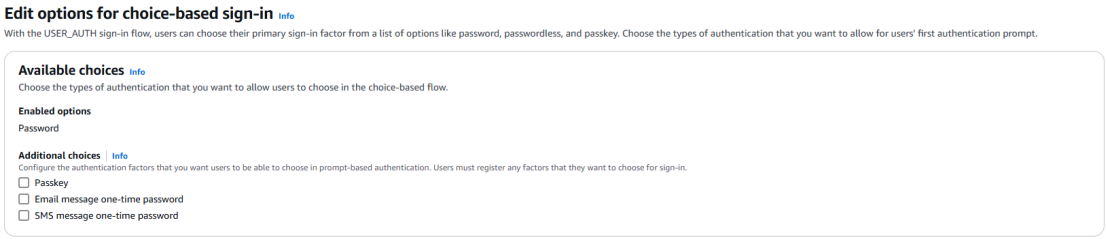
Para ativar a autenticação baseada em clientes com nome de usuário e SRP, configure o cliente da aplicação para permitir isso. No console do Amazon Cognito, navegue até o menu Clientes da aplicação em Aplicações na configuração do grupo de usuários. Para permitir o login com SRP

em uma aplicação nativa ou um aplicativo móvel do lado do cliente, edite um cliente da aplicação e selecione Fazer login com senha remota segura (SRP): ALLOW_USER_SRP_AUTH em Fluxos de autenticação.

Para ativar a autenticação baseada em opções com nome de usuário e SRP, edite o cliente da aplicação e selecione Login baseado em opções: ALLOW_USER_AUTH.



Para verificar se a autenticação por SRP está disponível em fluxos de autenticação baseada em opções, navegue até o menu Fazer login e revise a seção em Opções para login baseado em opções. Você pode fazer login com autenticação por SRP se a senha estiver visível em Opções disponíveis. A opção Senha inclui as variantes de autenticação por nome de usuário e senha em texto simples e por SRP.



Configure `ExplicitAuthFlows` com suas opções username-and-password de autenticação preferidas em uma UpdateUserPoolClientsolicitação CreateUserPoolClientor.

```
"ExplicitAuthFlows": [
   "ALLOW_USER_SRP_AUTH",
   "ALLOW_USER_AUTH"
]
```

Em uma UpdateUserPoolsolicitação CreateUserPoolor, configure `Policies` com os fluxos de autenticação com base em opções que você deseja oferecer suporte. O valor `PASSWORD` em `AllowedFirstAuthFactors` inclui as opções de fluxo de autenticação por senha de texto simples e SRP.

```
"Policies": {
   "SignInPolicy": {
      "AllowedFirstAuthFactors": [
         "PASSWORD",
         "EMAIL_OTP",
         "WEB_AUTHN"
      ]
   }
}
```

Choice-based sign-in with SRP

Para inscrever um usuário em um aplicativo com autenticação de nome de usuário e senha com SRP, configure o corpo da sua solicitação AdminInitiateAuthou InitiateAuthda seguinte forma. Essa solicitação de login será bem-sucedida ou continuará até o próximo desafio se o usuário atual for elegível para a autenticação por nome de usuário e senha. Caso contrário, ela responderá com uma lista de desafios de autenticação de fator primário disponíveis. Esse conjunto de parâmetros é o mínimo necessário para fazer login. Parâmetros adicionais estão disponíveis.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "PREFERRED_CHALLENGE" : "PASSWORD_SRP",
      "SRP_A" : "[g^a % N]"
   },
   "ClientId": "1example23456789"
}
```

Você também pode omitir o valor `PREFERRED_CHALLENGE` e receber uma resposta contendo uma lista de fatores de login elegíveis para o usuário.

```
{
   "AuthFlow": "USER_AUTH",
```

```
   "AuthParameters": {
      "USERNAME" : "testuser"
   },
   "ClientId": "1example23456789"
}
```

Se você não enviou um desafio preferencial ou o usuário enviado não for elegível para o desafio preferencial, o Amazon Cognito retornará uma lista de opções em `AvailableChallenges`. Quando `AvailableChallenges` inclui um `ChallengeName` de`PASSWORD_SRP`, você pode continuar a autenticação com uma resposta RespondToAuthChallengeou AdminRespondToAuthChallengedesafiar no formato a seguir. Você deve transmitir um parâmetro `Session` que associe a resposta do desafio à resposta da API à solicitação inicial de login. Esse conjunto de parâmetros é o mínimo necessário para fazer login. Parâmetros adicionais estão disponíveis.

```
{
   "ChallengeName": "PASSWORD_SRP",
   "ChallengeResponses": {
      "USERNAME" : "testuser",
      "SRP_A" : "[g^a % N]"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response"
}
```

O Amazon Cognito responde a solicitações de desafio preferencial elegíveis e respostas do desafio `PASSWORD_SRP` com um desafio `PASSWORD_VERIFIER`. Seu cliente deve concluir os cálculos do SRP e responder ao desafio em uma RespondToAuthChallengeAdminRespondToAuthChallengesolicitação.

```
{
   "ChallengeName": "PASSWORD_VERIFIER",
   "ChallengeResponses": {
      "PASSWORD_CLAIM_SIGNATURE" : "string",
      "PASSWORD_CLAIM_SECRET_BLOCK" : "string",
      "TIMESTAMP" : "string"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response]"
}
```

Em uma resposta bem-sucedida do desafio `PASSWORD_VERIFIER`, o Amazon Cognito emite tokens ou outro desafio obrigatório, como a autenticação multifator (MFA).

Client-based sign-in with SRP

A autenticação SRP é mais comum na autenticação do lado do cliente do que na autenticação do lado do servidor. No entanto, você pode usar a autenticação SRP com InitiateAuthe. AdminInitiateAuth Para conectar um usuário a uma aplicação, configure o corpo da solicitação `InitiateAuth` ou `AdminInitiateAuth` da forma a seguir. Esse conjunto de parâmetros é o mínimo necessário para fazer login. Parâmetros adicionais estão disponíveis.

O cliente gera `SRP_A` por meio de um gerador módulo N g elevado à potência de um inteiro aleatório secreto a.

```
{
   "AuthFlow": "USER_SRP_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "SRP_A" : "[g^a % N]"
   },
   "ClientId": "1example23456789"
}
```

O Amazon Cognito responde com um desafio `PASSWORD_VERIFIER`. Seu cliente deve concluir os cálculos do SRP e responder ao desafio em uma RespondToAuthChallengeAdminRespondToAuthChallengesolicitação.

```
{
   "ChallengeName": "PASSWORD_VERIFIER",
   "ChallengeResponses": {
      "PASSWORD_CLAIM_SIGNATURE" : "string",
      "PASSWORD_CLAIM_SECRET_BLOCK" : "string",
      "TIMESTAMP" : "string"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response]"
}
```

Em uma resposta bem-sucedida do desafio `PASSWORD_VERIFIER`, o Amazon Cognito emite tokens ou outro desafio obrigatório, como a autenticação multifator (MFA).

Fazer login sem senha com senhas de uso único

As senhas podem ser perdidas ou roubadas. Talvez você queira verificar somente se seus usuários têm acesso a um endereço de e-mail, número de telefone ou aplicação autenticadora verificado. A solução para isso é o login sem senha. A aplicação pode solicitar que os usuários insiram o nome de usuário, o endereço de e-mail ou o número de telefone. O Amazon Cognito então gera uma senha de uso único (OTP), um código que eles devem confirmar. Um código bem-sucedido conclui a autenticação.

Os fluxos de autenticação de senha única (OTP) não são compatíveis com a autenticação multifator (MFA) necessária em seu grupo de usuários. A autenticação por chave de acesso com verificação do usuário pode atender aos requisitos de MFA quando você configura o. `FactorConfiguration` `MULTI_FACTOR_WITH_USER_VERIFICATION` Se o MFA for opcional em seu grupo de usuários, os usuários que ativaram o MFA não poderão fazer login com um primeiro fator OTP. Usuários que não têm uma preferência de MFA em um grupo de usuários com MFA opcional podem fazer login sem senha. Para obter mais informações, consulte Informações importantes sobre a MFA de grupo de usuários.

Quando um usuário insere corretamente um código recebido por SMS ou e-mail como parte da autenticação sem senha, além de autenticar o usuário, o grupo de usuários marca o atributo de endereço de e-mail ou número de telefone não verificado do usuário como verificado. O status do usuário também muda de `UNCONFIRMED` para `CONFIRMED`, independentemente de você ter configurado o grupo de usuários para verificar automaticamente endereços de e-mail ou números de telefone.

Novas opções com login sem senha

Quando você ativa a autenticação sem senha no grupo de usuários, o funcionamento de alguns fluxos de usuários é alterado.

1. Os usuários podem se cadastrar sem uma senha e escolher um fator sem senha ao fazer login. Também é possível criar usuários sem senhas como administrador.
2. Os usuários que você importa por meio de um arquivo CSV podem fazer login imediatamente com um fator sem senha. Eles não precisam definir uma senha antes de fazer login.
3. Os usuários que não têm uma senha podem enviar solicitações de ChangePasswordAPI sem o `PreviousPassword` parâmetro.

Login automático com OTPs

Os usuários que se inscreverem e confirmarem suas contas de usuário por e-mail ou mensagem SMS OTPs podem fazer login automaticamente com o fator sem senha que corresponde à mensagem de confirmação. Na IU do login gerenciado, os usuários que confirmam suas contas e estão elegíveis para o login por OTP com o método de entrega do código de confirmação passam automaticamente para o primeiro login após fornecerem o código. Em seu aplicativo personalizado com um AWS SDK, transmita os seguintes parâmetros para uma InitiateAuthoperação or. AdminInitiateAuth

- O `Session` parâmetro da resposta da ConfirmSignUpAPI como parâmetro de `Session` solicitação.
- Um AuthFlowde`USER_AUTH`.

Você pode transmitir um PREFERRED_CHALLENGE de `EMAIL_OTP` ou `SMS_OTP`, mas não é obrigatório. O parâmetro `Session` fornece prova de autenticação e o Amazon Cognito o ignora `AuthParameters` quando você transmite um código de sessão válido.

A operação de login retorna a resposta que indica a autenticação bem-sucedida AuthenticationResult, sem desafios adicionais se as seguintes condições forem verdadeiras.

- O código `Session` é válido e não expirou.
- O usuário é elegível para o método de autenticação por OTP.

Activate passwordless sign-in

Console

Para ativar o login sem senha, configure o grupo de usuários para permitir o login primário com um ou mais tipos sem senha e, em seguida, configure o cliente da aplicação para permitir o fluxo `USER_AUTH`. No console do Amazon Cognito, navegue até o menu Fazer login em Autenticação na configuração do grupo de usuários. Edite Opções para login baseado em opções e selecione Senha única para mensagem de e-mail ou Senha única para mensagem SMS. É possível ativar as duas opções. Salve as alterações.

Navegue até o menu Clientes da aplicação e escolha um cliente ou crie um. Clique em Editar e selecione Selecionar um tipo de autenticação no login: ALLOW_USER_AUTH.

API/SDK

Na API de grupos de usuários, configure `SignInPolicy` com as opções sem senha apropriadas em uma solicitação CreateUserPoolor UpdateUserPool.

```
"SignInPolicy": {
    "AllowedFirstAuthFactors": [
        "EMAIL_OTP",
        "SMS_OTP"
    ]
}
```

Configure seu cliente de aplicativo `ExplicitAuthFlows` com a opção necessária em uma UpdateUserPoolClientsolicitação CreateUserPoolClientou.

```
"ExplicitAuthFlows": [
   "ALLOW_USER_AUTH"
]
```

Sign in with passwordless

O login sem senha não tem uma base de cliente `AuthFlow` que você possa especificar e. InitiateAuthAdminInitiateAuth A autenticação OTP só está disponível na opção baseada em`USER_AUTH`, onde você pode solicitar uma opção `AuthFlow` de login preferencial ou escolher a opção sem senha na de um usuário. AvailableChallenges Para conectar um usuário a uma aplicação, configure o corpo da solicitação `InitiateAuth` ou `AdminInitiateAuth` da forma a seguir. Esse conjunto de parâmetros é o mínimo necessário para fazer login. Parâmetros adicionais estão disponíveis.

Neste exemplo, não sabemos como o usuário deseja fazer login. Se adicionarmos um parâmetro `PREFERRED_CHALLENGE` e o desafio preferencial estiver disponível para o usuário, o Amazon Cognito responderá com esse desafio.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser"
   },
   "ClientId": "1example23456789"
}
```

Em vez disso, você pode adicionar `"PREFERRED_CHALLENGE": "EMAIL_OTP"` ou `"PREFERRED_CHALLENGE": "SMS_OTP"` a `AuthParameters` nesse exemplo. Se o usuário for elegível para esse método preferencial, o grupo de usuários enviará imediatamente um código para o endereço de e-mail ou número de telefone do usuário e retornará `"ChallengeName": "EMAIL_OTP"` ou `"ChallengeName": "SMS_OTP"`.

Se você não especificar um desafio preferencial, o Amazon Cognito responderá com um parâmetro `AvailableChallenges`.

```
{
   "AvailableChallenges": [
      "EMAIL_OTP",
      "SMS_OTP",
      "PASSWORD"
    ],
   "Session": "[Session ID]"
}
```

Esse usuário é elegível para login sem senha com OTP por e-mail, OTP por SMS e nome de usuário e senha. A aplicação pode solicitar que o usuário faça a seleção ou pode fazer uma seleção com base na lógica interna. Em seguida, ele prossegue com uma AdminRespondToAuthChallengesolicitação RespondToAuthChallengeor que seleciona o desafio. Suponha que o usuário queira concluir a autenticação sem senha com uma OTP enviada por e-mail.

```
{
   "ChallengeName": "SELECT_CHALLENGE",
   "ChallengeResponses": {
      "USERNAME" : "testuser",
      "ANSWER" : "EMAIL_OTP"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response]"
}
```

O Amazon Cognito responde com um desafio `EMAIL_OTP` e envia um código para o endereço de e-mail verificado do usuário. Sua aplicação deve então responder novamente a esse desafio.

Essa também seria a próxima resposta do desafio se você solicitasse `EMAIL_OTP` como `PREFERRED_CHALLENGE`.

```
{
   "ChallengeName": "EMAIL_OTP",
   "ChallengeResponses": {
      "USERNAME" : "testuser",
      "EMAIL_OTP_CODE" : "123456"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response]"
}
```

Login sem senha com chaves de acesso WebAuthn

As chaves de acesso são seguras e impõem um nível de esforço relativamente baixo aos usuários. O login com chave de acesso usa autenticadores, dispositivos externos que permitem a autenticação dos usuários. Senhas comuns expõem os usuários a vulnerabilidades como phishing, adivinhação de senhas e roubo de credenciais. Com as chaves de acesso, a aplicação pode se beneficiar de medidas de segurança avançadas em telefones celulares e outros dispositivos conectados ou integrados a sistemas de informação. Um fluxo de trabalho comum de login com chave de acesso começa com uma chamada para o dispositivo que invoca o gerenciador de senhas ou credenciais, por exemplo, o Keychain do iOS ou o gerenciador de senhas do Google Chrome. O gerenciador de credenciais do dispositivo solicita que o usuário selecione uma chave de acesso e a autorize com uma credencial existente ou um mecanismo de desbloqueio do dispositivo. Os telefones modernos têm leitores faciais, leitores de impressão digital, padrões de desbloqueio e outros mecanismos, alguns dos quais satisfazem simultaneamente os princípios de algo que você sabe e algo que você tem da autenticação forte. No caso da autenticação por chave de acesso com biometria, as chaves de acesso representam algo que você é.

Você pode querer substituir as senhas pela autenticação por impressão digital, reconhecimento facial ou chave de segurança. Isso é chave de acesso ou WebAuthnautenticação. É comum que os desenvolvedores de aplicações permitam que os usuários cadastrem um dispositivo biométrico após o primeiro login com senha. Com os grupos de usuários do Amazon Cognito, sua aplicação pode configurar essa opção de login para os usuários. A autenticação por chave de acesso pode atender aos requisitos de autenticação multifator (MFA) quando seu grupo de usuários estiver configurado como. `FactorConfiguration` `MULTI_FACTOR_WITH_USER_VERIFICATION` Nessa configuração, a autenticação por chave de acesso com verificação do usuário conta como autenticação multifatorial.

Os fluxos de autenticação de senha única (OTP) não são compatíveis com a autenticação multifator (MFA) necessária em seu grupo de usuários. A autenticação por chave de acesso com verificação do usuário pode atender aos requisitos de MFA quando você configura o `FactorConfiguration` `MULTI_FACTOR_WITH_USER_VERIFICATION` Se o MFA for opcional em seu grupo de usuários, os usuários que ativaram o MFA não poderão fazer login com um primeiro fator OTP. Usuários que não têm uma preferência de MFA em um grupo de usuários com MFA opcional podem fazer login sem senha. Para obter mais informações, consulte Informações importantes sobre a MFA de grupo de usuários.

O que são chaves de acesso?

As chaves de acesso simplificam a experiência do usuário, eliminando a necessidade de lembrar senhas complexas ou OTPs digitá-las. As chaves de acesso são baseadas WebAuthn e CTAP2 padrões elaborados pelo World Wide Web Consortium (W3C) e pela FIDO (Fast Identity Online) Alliance. Os navegadores e plataformas implementam esses padrões, fornecem aplicativos web ou móveis APIs para iniciar um processo de registro ou autenticação de chave de acesso e também uma interface de usuário para o usuário selecionar e interagir com um autenticador de chave de acesso.

Quando um usuário registra um autenticador em um site ou aplicativo, o autenticador cria um par de chaves público-privado. WebAuthn navegadores e plataformas enviam a chave pública para o back-end do aplicativo ou site. O autenticador mantém a chave privada, a chave IDs e os metadados sobre o usuário e o aplicativo. Quando o usuário deseja se autenticar na aplicação registrada com seu autenticador registrado, a aplicação gera um desafio aleatório. A resposta a esse desafio é a assinatura digital do desafio gerada com a chave privada do autenticador dessa aplicação e usuário, além de metadados relevantes. O navegador ou a plataforma da aplicação recebe a assinatura digital e a envia para o backend da aplicação. A aplicação então valida a assinatura com a chave pública armazenada.

> **Note**
>
> Sua aplicação não recebe nenhum segredo de autenticação que os usuários forneçam ao autenticador, nem recebe informações sobre a chave privada.

Veja a seguir alguns dos exemplos e recursos dos autenticadores atualmente disponíveis no mercado. Um autenticador pode atender a uma ou a todas estas categorias.

- Alguns autenticadores realizam a verificação de usuário com fatores como um PIN, entrada biométrica com reconhecimento facial/impressão digital ou uma senha antes de conceder acesso, garantindo que somente o usuário legítimo possa autorizar ações. Outros autenticadores não têm nenhum recurso de verificação de usuário, e alguns podem ignorar a verificação quando uma aplicação não a exige.
- Alguns autenticadores, por exemplo, tokens YubiKey de hardware, são portáteis. Eles se comunicam com dispositivos por meio de conexões USB, Bluetooth ou NFC. Alguns autenticadores são locais e vinculados a uma plataforma, como o Windows Hello em um PC ou o Face ID em um iPhone. Um autenticador vinculado ao dispositivo pode ser transportado pelo usuário se for pequeno o suficiente, como um dispositivo móvel. Às vezes, os usuários podem conectar o autenticador de hardware a várias plataformas diferentes com comunicação sem fio. Por exemplo, usuários em navegadores de desktop podem usar seu smartphone como autenticador de chave de acesso ao escanear um código QR.
- Algumas chaves de acesso vinculadas à plataforma são sincronizadas com a nuvem, permitindo seu uso em vários locais. Por exemplo, as chaves de acesso do Face ID nos iPhones sincronizam os metadados da chave de acesso com as contas da Apple dos usuários no iCloud Keychain. Essas chaves de acesso garantem uma autenticação perfeita em todos os dispositivos Apple, em vez de exigir que os usuários registrem cada dispositivo individualmente. Aplicações de autenticação baseadas em software, como 1Password, Dashlane e Bitwarden, sincronizam chaves de acesso em todas as plataformas nas quais o usuário instalou a aplicação.

Na WebAuthn terminologia, sites e aplicativos são partes confiáveis. Cada chave de acesso está associada a um ID de parte confiável específico, um identificador unificado que representa os sites ou aplicações que aceitam a autenticação por chave de acesso. Os desenvolvedores devem selecionar cuidadosamente o ID de parte confiável para garantir o escopo correto de autenticação. Um ID de parte confiável típico é o nome de domínio raiz de um servidor web. Uma chave de acesso com essa especificação de ID de parte confiável pode autenticar nesse domínio e em seus subdomínios. Navegadores e plataformas negam a autenticação por chave de acesso quando o URL do site que o usuário deseja acessar não corresponde ao ID de parte confiável. Da mesma forma, para aplicativos móveis, uma chave de acesso só pode ser usada se o caminho da aplicação estiver presente nos arquivos de associação `.well-known` que a aplicação disponibiliza no caminho indicado pelo ID de parte confiável.

As chaves de acesso são detectáveis. Elas podem ser reconhecidas e usadas automaticamente por um navegador ou plataforma sem exigir que o usuário insira um nome de usuário. Quando um usuário visita um site ou aplicação compatível com a autenticação por chave de acesso, ele pode

selecionar uma chave de acesso em uma lista que o navegador ou a plataforma já conhece, ou pode escanear um código QR.

Como o Amazon Cognito implementa a autenticação por chave de acesso?

As chaves de acesso são um recurso opcional disponível em todos os planos de recursos, exceto no Lite. Elas estão disponíveis somente no fluxo de autenticação baseada em opções. Com o login gerenciado, o Amazon Cognito lida com a lógica da autenticação por chave de acesso. Você também pode usar a API de grupos de usuários do Amazon Cognito AWS SDKs para fazer a autenticação por chave de acesso no back-end do seu aplicativo.

O Amazon Cognito reconhece chaves de acesso criadas usando um dos dois algoritmos criptográficos assimétricos, ES256 (-7) e (-257). RS256 A maioria dos autenticadores é compatível com os dois algoritmos. Por padrão, os usuários podem configurar qualquer tipo de autenticador, por exemplo, tokens de hardware, smartphones móveis e aplicações autenticadoras de software. No momento, o Amazon Cognito não é compatível com a aplicação de atestados.

No grupo de usuários, é possível configurar a verificação de usuário como preferencial ou obrigatória. Essa configuração é definida como preferencial por padrão em solicitações de API que não fornecem um valor, e é selecionada por padrão no console do Amazon Cognito. Quando você define a verificação de usuário como preferencial, os usuários podem configurar autenticadores que não têm o recurso de verificação de usuário, e as operações de registro e autenticação podem ser bem-sucedidas sem a verificação de usuário. Para exigir a verificação de usuário no registro e na autenticação por chave de acesso, altere essa configuração para obrigatória.

A definição do ID de parte confiável (RP) na configuração da chave de acesso é uma decisão importante. Quando você não especifica o contrário e a versão de marca do domínio é login gerenciado, o grupo de usuários considera, por padrão, o nome do domínio personalizado como o ID de RP. Se você não tiver um domínio personalizado e não especificar o contrário, o grupo de usuários utilizará como padrão o ID de RP do domínio de prefixo. Você também pode configurar o ID de RP para ser qualquer nome de domínio que não esteja na lista de sufixos públicos (PSL). A entrada do ID de RP se aplica ao registro e à autenticação por chave de acesso no login gerenciado e na autenticação do SDK. A chave de acesso só funciona em aplicativos móveis se o Amazon Cognito conseguir localizar um arquivo de associação `.well-known` com o ID de RP como domínio. Como prática recomendada, determine e defina o valor do ID de parte confiável antes que o site ou a aplicação esteja disponível publicamente. Se você alterar o ID de RP, os usuários deverão se registrar novamente com o novo ID de RP.

Cada usuário pode registrar até vinte chaves de acesso. O registro de uma chave de acesso só é possível após o usuário ter feito login no grupo de usuários pelo menos uma vez. O login gerenciado elimina grande parte do esforço necessário para o registro de chaves de acesso. Quando você habilita a autenticação por chave de acesso para um grupo de usuários e um cliente da aplicação, o grupo de usuários com um domínio de login gerenciado lembra os usuários finais de registrarem uma chave de acesso após se cadastrarem em uma nova conta. Você também pode invocar os navegadores dos usuários a qualquer momento para direcioná-los a uma página de login gerenciado para registro da chave de acesso. Os usuários devem fornecer um nome de usuário antes que o Amazon Cognito possa iniciar a autenticação por chave de acesso. O login gerenciado lida com isso automaticamente. A página de login solicita um nome de usuário, valida se o usuário tem pelo menos uma chave de acesso registrada e, em seguida, solicita o login por chave de acesso. Da mesma forma, as aplicações baseados em SDK devem solicitar um nome de usuário e fornecê-lo na solicitação de autenticação.

Quando você configura a autenticação do grupo de usuários com chaves de acesso e tem um domínio personalizado e um domínio de prefixo, o ID de RP usa como padrão o nome de domínio totalmente qualificado (FQDN) do domínio personalizado. Para definir um domínio de prefixo como o ID de RP no console do Amazon Cognito, exclua seu domínio personalizado ou insira o FQDN do domínio de prefixo como um domínio de terceiros.

Activate passkey sign-in

Console

Para ativar o login com chaves de acesso, configure o grupo de usuários para permitir o login primário com um ou mais tipos sem senha e, em seguida, configure o cliente da aplicação para permitir o fluxo `USER_AUTH`. No console do Amazon Cognito, navegue até o menu Fazer login em Autenticação na configuração do grupo de usuários. Edite Opções para login baseado em opções e adicione Chave de acesso à lista Opções disponíveis.

Navegue até o menu Métodos de autenticação e edite Chave de acesso.

- Verificação de usuário é a configuração para determinar se o grupo de usuários exige dispositivos com chave de acesso que realizem verificações adicionais para garantir que o usuário atual esteja autorizado a usar uma chave de acesso. Para incentivar os usuários a configurar um dispositivo com a verificação de usuário, mas não a tornar obrigatória, selecione Preferencial. Para oferecer suporte somente a dispositivos com verificação de usuário, selecione Obrigatório. Para obter mais informações, consulte User verification em w3.org.

- Domínio para ID de parte confiável é o identificador que a aplicação transmitirá nas solicitações de registro de chave de acesso dos usuários. Ele define a meta da relação de confiança com o emissor das chaves de acesso dos usuários. O ID de parte confiável pode ser: o domínio do seu grupo de usuários, se

  Domínio Cognito

  O domínio de prefixo do Amazon Cognito do seu grupo de usuários.

  Domínio personalizado

  O domínio personalizado do seu grupo de usuários.

  Domínio de terceiros

  O domínio para aplicações que não usam as páginas de login gerenciado dos grupos de usuários. Essa configuração geralmente está associada a grupos de usuários que não têm um domínio e realizam autenticação com um AWS SDK e a API de grupos de usuários no back-end.

Navegue até o menu Clientes da aplicação e escolha um cliente ou crie um. Clique em Editar e, em Fluxos de autenticação, selecione Selecionar um tipo de autenticação no login: ALLOW_USER_AUTH.

API/SDK

Na API de grupos de usuários, configure `SignInPolicy` com as opções de chave de acesso apropriadas em uma UpdateUserPoolsolicitação CreateUserPoolor. A opção `WEB_AUTHN` para autenticação por chave de acesso deve ser acompanhada por pelo menos uma outra opção. O registro da chave de acesso requer uma sessão de autenticação existente.

```
"SignInPolicy": {
    "AllowedFirstAuthFactors": [
        "PASSWORD",
        "WEB_AUTHN"
    ]
}
```

Configure sua preferência de verificação de usuário e ID de RP no `WebAuthnConfiguration` parâmetro de uma SetUserPoolMfaConfigsolicitação. `RelyingPartyId`, o destino pretendido dos resultados da autenticação por chave de acesso, pode ser o domínio personalizado ou de prefixo do grupo de usuários ou um domínio de sua escolha.

```
"WebAuthnConfiguration": {
   "RelyingPartyId": "example.auth.us-east-1.amazoncognito.com",
   "UserVerification": "preferred",
   "FactorConfiguration": "SINGLE_FACTOR"
}
```

Configure seu cliente de aplicativo `ExplicitAuthFlows` com a opção necessária em uma UpdateUserPoolClientsolicitação CreateUserPoolClientou.

```
"ExplicitAuthFlows": [
   "ALLOW_USER_AUTH"
]
```

Register a passkey (managed login)

O login gerenciado gerencia o registro das chaves de acesso dos usuários. Quando a autenticação por chave de acesso está ativa no grupo de usuários, o Amazon Cognito solicita que os usuários configurem uma chave de acesso ao se registrarem em uma nova conta.

O Amazon Cognito não solicita que os usuários configurem uma chave de acesso quando eles já se cadastraram e não configuraram uma chave de acesso, ou se você criou a conta deles como administrador. Os usuários nesse estado devem fazer login com outro fator, como uma senha ou uma OTP sem senha, antes de poderem registrar uma chave de acesso.

Como registrar uma chave de acesso

1. Direcione o usuário para a página de login.

   ```
   https://auth.example.com/oauth2/authorize/?
   client_id=1example23456789&response_type=code&scope=email+openid
   +phone&redirect_uri=https%3A%2F%2Fwww.example.com
   ```

2. Processe o resultado da autenticação do usuário. Neste exemplo, o Amazon Cognito o redireciona para `www.example.com` com um código de autorização que a aplicação troca por tokens.
3. Direcione o usuário para a página de registro de chave de acesso. O usuário terá um cookie de navegador que mantém a sessão ativa. O URL da chave de acesso aceita os parâmetros `client_id` e `redirect_uri`. O Amazon Cognito permite que somente usuários autenticados acessem essa página. Faça login do usuário com uma senha, uma

OTP enviada por e-mail ou uma OTP enviada por SMS e, em seguida, invoque um URL que corresponda ao padrão a seguir.

Você também pode adicionar outros parâmetros Autorizar endpointa essa solicitação, como `response_type` e `scope`.

```
https://auth.example.com/passkeys/add?
client_id=1example23456789&redirect_uri=https%3A%2F%2Fwww.example.com
```

Register a passkey (SDK)

Você registra as credenciais da chave de acesso com metadados em um objeto. PublicKeyCreationOptions Você pode gerar esse objeto com as credenciais de um usuário conectado e apresentá-las em uma solicitação de API ao emissor da chave de acesso. O emissor retornará um objeto RegistrationResponseJSON que confirma o registro da chave de acesso.

Para iniciar o processo de registro da chave de acesso, conecte um usuário com uma opção de login existente. Autorize a solicitação de StartWebAuthnRegistrationAPI autorizada pelo token com o token de acesso do usuário atual. Veja a seguir o corpo de um exemplo de solicitação `GetWebAuthnRegistrationOptions`.

```
{
   "AccessToken": "eyJra456defEXAMPLE"
}
```

A resposta do grupo de usuários contém o objeto `PublicKeyCreationOptions`. Apresente esse objeto em uma solicitação de API para o emissor do usuário. Ele fornece informações como a chave pública e o ID de parte confiável. O emissor responderá com um objeto `RegistrationResponseJSON`.

Apresente a resposta do registro em uma solicitação de CompleteWebAuthnRegistrationAPI, novamente autorizada com o token de acesso do usuário. Quando o grupo de usuários responder com uma resposta HTTP 200 com um corpo vazio, a chave de acesso do usuário estará registrada.

Sign in with a passkey

O login sem senha não tem um nome `AuthFlow` que você possa especificar e. InitiateAuthAdminInitiateAuth Em vez disso, você deve declarar um `AuthFlow` de `USER_AUTH` e

solicitar uma opção de login ou escolher a opção sem senha na resposta do grupo de usuários. Para conectar um usuário a uma aplicação, configure o corpo da solicitação `InitiateAuth` ou `AdminInitiateAuth` da forma a seguir. Esse conjunto de parâmetros é o mínimo necessário para fazer login. Parâmetros adicionais estão disponíveis.

Neste exemplo, sabemos que o usuário deseja fazer login com uma chave de acesso e adicionamos um parâmetro `PREFERRED_CHALLENGE`.

```
{
   "AuthFlow": "USER_AUTH",
   "AuthParameters": {
      "USERNAME" : "testuser",
      "PREFERRED_CHALLENGE" : "WEB_AUTHN"
   },
   "ClientId": "1example23456789"
}
```

O Amazon Cognito responde com um desafio `WEB_AUTHN`. Sua aplicação deve responder a esse desafio. Inicie uma solicitação de login com o provedor da chave de acesso do usuário. Ele retornará um objeto AuthenticationResponseJSON.

```
{
   "ChallengeName": "WEB_AUTHN",
   "ChallengeResponses": {
      "USERNAME" : "testuser",
      "CREDENTIAL" : "{AuthenticationResponseJSON}"
   },
   "ClientId": "1example23456789",
   "Session": "[Session ID from the previous response]"
}
```

MFA após o login

Você pode configurar usuários que concluem o login com um fluxo de nome de usuário e senha para serem solicitados a fazer uma verificação adicional com uma senha de uso único enviada por e-mail, SMS ou uma aplicação geradora de código. O MFA é diferente do login sem senha com senhas de uso único. No entanto, as chaves de acesso com verificação de usuário podem atender aos requisitos de MFA quando você `FactorConfiguration` configura `MULTI_FACTOR_WITH_USER_VERIFICATION` como em seu grupo de usuários.

`WebAuthnConfiguration` Para fluxos baseados em senha, o MFA em grupos de usuários é um modelo de resposta a desafios em que um usuário primeiro demonstra que sabe a senha e, em seguida, demonstra que tem acesso ao dispositivo registrado de segundo fator.

Recursos de implementação

- Adicionar MFA a um grupo de usuários

Tokens de atualização

Sua aplicação utiliza tokens de atualização para manter os usuários conectados sem a necessidade de inserir suas credenciais novamente. As aplicações podem apresentar tokens de atualização ao grupo de usuários e trocá-los por novos tokens de ID e acesso. Com o token de atualização, você pode garantir que um usuário conectado ainda esteja ativo, obter informações de atributos atualizadas e atualizar os direitos de controle de acesso sem a intervenção do usuário.

Recursos de implementação

- Tokens de atualização

Autenticação personalizada

Você pode querer configurar um método de autenticação para seus usuários que não esteja listado aqui. Você pode fazer isso com a autenticação personalizada usando acionadores do Lambda. Em uma sequência de funções do Lambda, o Amazon Cognito emite um desafio, faz uma pergunta que os usuários devem responder, verifica a precisão da resposta e determina se outro desafio deve ser emitido. As perguntas e respostas podem incluir perguntas de segurança, solicitações a um serviço CAPTCHA, solicitações a uma API de serviço de MFA externa ou tudo isso em sequência.

Recursos de implementação

- Acionadores do Lambda de desafio personalizado de autenticação

Fluxo de autenticação personalizado

Os grupos de usuários do Amazon Cognito também permitem usar fluxos de autenticação personalizados, os quais podem ajudar você a criar um modelo de autenticação baseado em desafio/resposta usando acionadores do AWS Lambda .

O fluxo de autenticação personalizado possibilita ciclos personalizados de desafio e resposta para atender a diferentes requisitos. O fluxo começa com uma chamada para a operação de API `InitiateAuth` que indica o tipo de autenticação que será usado e fornece todos os parâmetros de autenticação inicial. O Amazon Cognito responde à chamada do `InitiateAuth` com um dos seguintes tipos de informação:

- Um desafio para o usuário com uma sessão e parâmetros
- Um erro se houver falha na autenticação do usuário.
- ID, acesso e tokens de atualização, se os parâmetros fornecidos na chamada de `InitiateAuth` forem suficientes para que o usuário faça login. (Normalmente, o usuário ou a aplicação deve primeiro responder a um desafio, mas seu código personalizado deve determinar isso.)

Se o Amazon Cognito responder à chamada `InitiateAuth` com um desafio, a aplicação reunirá mais entradas e chamará a operação `RespondToAuthChallenge`. Essa chamada fornece as respostas do desafio e repassa a sessão. O Amazon Cognito responde à chamada `RespondToAuthChallenge` de forma semelhante à chamada `InitiateAuth`. Se o usuário tiver feito login, o Amazon Cognito fornecerá tokens ou, se o usuário não estiver conectado, o Amazon Cognito apresentará outro desafio ou um erro. Se o Amazon Cognito retornar outro desafio, a sequência se repetirá e a aplicação chamará `RespondToAuthChallenge` até que o usuário faça login com êxito ou um erro seja retornado. Mais detalhes sobre as operações de API `InitiateAuth` e `RespondToAuthChallenge` são fornecidos na documentação da API.

Fluxo de autenticação personalizado e desafios

Um aplicativo pode iniciar um fluxo de autenticação personalizado chamando `InitiateAuth` com `CUSTOM_AUTH` como o `Authflow`. Com um fluxo de autenticação personalizado, três acionadores do Lambda controlam os desafios e a verificação das respostas.

- O acionador `DefineAuthChallenge` do Lambda usa como entrada uma matriz de sessão de desafios e respostas anteriores. Depois, ele gera o nome do próximo desafio e os boolianos que indicam se o usuário está autenticado e pode receber tokens. Esse acionador do Lambda é uma máquina de estado que controla o caminho do usuário por meio dos desafios.
- O acionador `CreateAuthChallenge` do Lambda usa um nome de desafio como entrada e gera o desafio e os parâmetros para avaliar a resposta. Quando `DefineAuthChallenge` retorna `CUSTOM_CHALLENGE` como o próximo desafio, o fluxo de autenticação chama `CreateAuthChallenge`. O acionador `CreateAuthChallenge` do Lambda passa o próximo tipo de desafio no parâmetro de metadados de desafio.

- A função do `VerifyAuthChallengeResponse` Lambda avalia a resposta e retorna um booleano para indicar se a resposta foi válida.

Um fluxo de autenticação personalizado também pode usar uma combinação de desafios integrados, como verificação de senha SRP e MFA por SMS. Ele pode usar desafios personalizados, como CAPTCHA ou perguntas secretas.

Usar verificação de senha SRP no fluxo de autenticação personalizado

Para incluir a SRP em um fluxo de autenticação personalizado, você deve começar com ele.

- Para iniciar a verificação de senha SRP em um fluxo personalizado, o aplicativo chama `InitiateAuth` com `CUSTOM_AUTH` como o `Authflow`. No mapa de `AuthParameters`, a solicitação de sua aplicação inclui `SRP_A:` (o valor de SRP A) e `CHALLENGE_NAME: SRP_A`.
- O fluxo de `CUSTOM_AUTH` invoca o acionador do Lambda `DefineAuthChallenge` com uma sessão inicial de `challengeName: SRP_A` e `challengeResult: true`. Sua função do Lambda responde com `challengeName: PASSWORD_VERIFIER`, `issueTokens: false` e `failAuthentication: false`.
- Depois, a aplicação deve chamar `RespondToAuthChallenge` com `challengeName: PASSWORD_VERIFIER` e os outros parâmetros necessários para a SRP no mapa `challengeResponses`.
- Se o Amazon Cognito verificar a senha, `RespondToAuthChallenge` invocará o acionador `DefineAuthChallenge` do Lambda com uma segunda sessão de `challengeName: PASSWORD_VERIFIER` e `challengeResult: true`. Nesse ponto, o acionador do Lambda `DefineAuthChallenge` pode responder com `challengeName: CUSTOM_CHALLENGE` para iniciar o desafio personalizado.
- Se a MFA estiver habilitada para um usuário, depois que o Amazon Cognito verificar a senha, o usuário será desafiado a configurar ou fazer login com a MFA.

> **Note**
>
> A página da Web de login hospedada do Amazon Cognito não pode ativar Acionadores do Lambda de desafio personalizado de autenticação.

Para obter mais informações sobre os acionadores do Lambda, incluindo o código de exemplo, consulte Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda.

Fluxo de autenticação de migração de usuários

Um acionador de migração de usuários do Lambda ajuda a migrar usuários de um sistema de gerenciamento de usuários herdado para seu grupo de usuários. Se você escolher o fluxo de autenticação `USER_PASSWORD_AUTH`, os usuários não terão que redefinir suas senhas durante a migração de usuários. Esse fluxo envia as senhas dos usuários para o serviço por uma conexão SSL criptografada durante a autenticação.

Quando você concluir a migração de todos os usuários, alterne os fluxos para o fluxo de SRP mais seguro. O fluxo de SRP não envia senhas pela rede.

Para saber mais sobre acionadores do Lambda, consulte Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda.

Para obter mais informações sobre como migrar usuários com um acionador do Lambda, consulte Como importar usuários com um acionador do Lambda de migração de usuários.

Modelos de autorização para autenticação de API e SDK

Ao iniciar o desenvolvimento da sua aplicação com a autenticação de grupos de usuários, você deve decidir qual modelo de autorização de API se adequa ao tipo da sua aplicação. Um modelo de autorização é um sistema para fornecer autorizações para fazer solicitações com os componentes de autenticação nas integrações de API e SDK dos grupos de usuários do Amazon Cognito. O Amazon Cognito tem três modelos de autorização: autorizado pelo IAM, público e autorizado por token.

Com solicitações autorizadas pelo IAM, a autorização provém de uma assinatura de um conjunto de credenciais do AWS IAM no cabeçalho `Authorization` de uma solicitação. Para aplicações do lado do servidor, essa prática protege as operações de autenticação com autorização do IAM. Com solicitações de autenticação públicas (não autenticadas), nenhuma autorização é necessária. Isso é adequado para aplicações do lado do cliente distribuídas aos usuários. Com operações autorizadas por token, normalmente implementadas em combinação com operações públicas, a autorização provém de um token de sessão ou de um token de acesso incluído no cabeçalho `Authorization` da solicitação. A autenticação do Amazon Cognito normalmente exige que você implemente duas ou mais operações de API em ordem, e as operações de API utilizadas dependem das características da sua aplicação. Clientes públicos, onde a aplicação é distribuída aos usuários, usam operações públicas, nas quais as solicitações de login não exigem autorização. As operações autorizadas por

tokens mantêm a sessão dos usuários em aplicações públicas. Clientes do lado do servidor, onde a lógica da aplicação está hospedada em um sistema remoto, protegem as operações de autenticação com a autorização do IAM para solicitações de login. Os pares de operações de API a seguir e seus métodos de SDK correspondentes são mapeados para os modelos de autorização disponíveis.

Cada operação de autenticação pública tem alguma forma de equivalente do lado do servidor, por exemplo e. UpdateUserAttributesAdminUpdateUserAttributes Enquanto as operações do lado do cliente são iniciadas pelo usuário e exigem confirmação, as operações do lado do servidor pressupõem que a alteração foi confirmada por um administrador do grupo de usuários e as alterações entram em vigor imediatamente. Neste exemplo, o Amazon Cognito envia uma mensagem com um código de confirmação para o usuário, e o token de acesso do usuário autoriza uma VerifyUserAttributesolicitação que envia o código. A aplicação do lado do servidor pode definir imediatamente o valor de qualquer atributo, embora considerações especiais se apliquem à alteração do valor de endereços de e-mail e números de telefone quando usados para login.

Para comparar a autenticação de API e ver uma lista completa das operações de API e seus modelos de autorização, consulte Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado.

Client-side (public) authentication

Veja a seguir uma sequência típica de solicitações em uma aplicação do lado do cliente.

1. A InitiateAuthoperação pública envia credenciais primárias, como nome de usuário e senha.
2. A RespondToAuthChallengeoperação autorizada pelo token envia um token de sessão a partir da `InitiateAuth` resposta e da resposta a um desafio, por exemplo, MFA. A autorização do token de sessão indica solicitações que fazem parte dos ciclos de not-yet-complete autenticação.
3. A ConfirmDeviceoperação autorizada pelo token envia um token de acesso e executa a operação de gravação de adicionar um dispositivo lembrado ao perfil do usuário. A autorização do token de acesso indica solicitações que são para operações de autoatendimento do usuário após a conclusão da autenticação.

Para obter mais informações, consulte Opções de autenticação do lado do cliente e Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado.

Server-side authentication

Veja a seguir uma sequência típica de solicitações de uma operação do lado do servidor. Cada solicitação tem um cabeçalho de autorização do AWS Signature versão 4 assinado com as credenciais da máquina IAM emitidas para o servidor da aplicação.

1. A AdminInitiateAuthoperação envia credenciais primárias, como nome de usuário e senha.
2. AdminRespondToAuthChallengeoperação envia a resposta a um desafio, por exemplo, MFA.
3. A AdminUpdateDeviceStatusoperação define a chave do dispositivo a partir da `AdminInitiateAuth` resposta conforme lembrada.

Para obter mais informações, consulte Opções de autenticação do lado do servidor e Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado.

Um usuário faz a autenticação respondendo a desafios sucessivos até que ela falhe ou o Amazon Cognito emita tokens para o usuário. Você pode repetir essas etapas com o Amazon Cognito, em um processo que inclui desafios diferentes, para comportar qualquer fluxo de autenticação personalizado.

Tópicos

- Opções de autenticação do lado do servidor
- Opções de autenticação do lado do cliente
- Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado
- Lista de operações de API agrupadas por modelo de autorização

Opções de autenticação do lado do servidor

As aplicações Web e outras aplicações do lado do servidor implementam a autenticação em um servidor remoto que um cliente carrega em uma aplicação de exibição remota, como um navegador ou uma sessão SSH. As aplicações do lado do servidor geralmente têm as características a seguir.

- Elas são criadas em uma aplicação instalada em um servidor em linguagens como Java, Ruby ou Node.js.
- Elas se conectam a clientes da aplicação de grupos de usuários que podem ter um segredo do cliente, chamados de clientes confidenciais.

- Eles têm acesso às AWS credenciais.
- Elas invocam o login gerenciado para autenticação ou usam operações autorizadas pelo IAM na API de grupos de usuários com um SDK da AWS .
- Elas atendem clientes internos e podem atender clientes públicos.

As operações do lado do servidor com a API de grupos de usuários podem usar senhas, senhas de uso único ou chaves de acesso como o principal fator de login. Para aplicações no lado do servidor, a autenticação do grupo de usuários é semelhante à das aplicações no lado do cliente, exceto pelo seguinte:

- O aplicativo do lado do servidor faz uma AdminInitiateAuthsolicitação de API. Essa operação requer AWS credenciais com permissões que incluem `cognito-idp:AdminInitiateAuth` e. `cognito-idp:AdminRespondToAuthChallenge` A operação retorna o desafio exigido ou o resultado da autenticação.
- Quando o aplicativo recebe um desafio, ele faz uma solicitação de AdminRespondToAuthChallengeAPI. A operação de API `AdminRespondToAuthChallenge` também requer credenciais da AWS .

Para obter mais informações sobre a assinatura de solicitações da API do Amazon Cognito com AWS credenciais, consulte Processo de assinatura do Signature versão 4 na AWS Referência geral.

Na resposta `AdminInitiateAuth` `ChallengeParameters`, o atributo `USER_ID_FOR_SRP`, se estiver presente, incluirá o nome do usuário real, não um alias (como o endereço de e-mail ou o número de telefone). Na chamada para `AdminRespondToAuthChallenge`, nas `ChallengeResponses`, é necessário transmitir esse nome de usuário no parâmetro `USERNAME`.

> **Note**
>
> Como as implementações de administração de backend usam o fluxo de autenticação de administração, o fluxo não é compatível com dispositivos memorizados. Quando você ativa o rastreamento de dispositivo, a autenticação de administração é executada com êxito, mas qualquer chamada para atualizar o token de acesso falha.

Opções de autenticação do lado do cliente

Aplicativos móveis e outros tipos de aplicações do lado do cliente são instalados nos dispositivos dos usuários e executam a lógica de autenticação e interface do usuário localmente. Eles geralmente têm as características a seguir.

- Eles são desenvolvidos em linguagens como React Native, Flutter e Swift e implantados nos dispositivos do usuário.
- Eles se conectam a clientes da aplicação de grupos de usuários que não têm um segredo do cliente, chamados de clientes públicos.
- Eles não têm acesso às AWS credenciais que autorizariam solicitações de API autorizadas pelo IAM.
- Eles invocam o login gerenciado para autenticação ou usam operações públicas e autorizadas por token na API de grupos de usuários com um SDK. AWS
- Eles atendem clientes públicos e permitem que qualquer pessoa se cadastre e faça login.

As operações do lado do cliente com a API de grupos de usuários podem usar senhas, senhas de uso único ou chaves de acesso como o principal fator de login. O processo a seguir funciona para aplicativos do lado do cliente do usuário que você cria com AWS Amplifyou o. AWS SDKs

1. O usuário insere suas respectivas credenciais no aplicativo.
2. A aplicação chama a operação `InitiateAuth` com o nome de usuário e os detalhes da Secure Remote Password (SRP).

   Essa operação da API retorna os parâmetros de autenticação.

   > **Note**
   >
   > O aplicativo gera detalhes do SRP com os recursos do Amazon Cognito SRP incorporados ao. AWS SDKs

3. O aplicativo chama a operação `RespondToAuthChallenge`. Se a chamada for bem-sucedida, o Amazon Cognito retornará os tokens do usuário e o fluxo de autenticação será concluído.

   Se o Amazon Cognito exigir outro desafio, a chamada para `RespondToAuthChallenge` não retornará tokens. Em vez disso, a chamada retornará uma sessão.

4. Se `RespondToAuthChallenge` retornar uma sessão, o aplicativo chamará `RespondToAuthChallenge` novamente, dessa vez com a sessão e a resposta ao desafio (por exemplo, código de MFA).

Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado

Os grupos de usuários do Amazon Cognito são uma combinação de várias tecnologias de autenticação. Eles confiam em provedores de identidade externos (IdPs). Eles são IdPs para aplicativos que implementam autenticação com o OpenID Connect (OIDC). SDKs Eles fornecem autenticação como emissores de tokens web JSON (JWTs) semelhante à autenticação OIDC, mas em métodos de API que fazem parte do. AWS SDKs Também podem servir como pontos de entrada seguros para suas aplicações.

Quando quiser se inscrever, fazer login e gerenciar usuários no grupo de usuários, você terá duas opções.

1. As páginas de login gerenciado e a IU hospedada clássica incluem os endpoints interativos de login gerenciado e os endpoints de federação que lidam com funções de IdP e de partes confiáveis. Eles formam um pacote de páginas da web públicas que o Amazon Cognito ativa quando você seleciona um domínio para o grupo de usuários. Para começar rapidamente com os recursos de autenticação e autorização dos grupos de usuários do Amazon Cognito, incluindo páginas para cadastro, login, gerenciamento de senhas e autenticação multifator (MFA), use a interface de usuário integrada do login gerenciado.

   Os outros endpoints do pool de usuários facilitam a autenticação com provedores de identidade terceirizados (IdPs). Os serviços que eles realizam incluem o seguinte:

   a. Endpoints de retorno de chamada do provedor de serviços para reivindicações autenticadas de você, como e. IdPs `saml2/idpresponse` `oauth2/idpresponse` Quando o Amazon Cognito é um provedor de serviços (SP) intermediário entre sua aplicação e o IdP, os endpoints de retorno de chamada representam o serviço.

   b. Endpoints que fornecem informações sobre seu ambiente, como `oauth2/userInfo` e `/.well-known/jwks.json`. Seu aplicativo usa esses endpoints quando verifica tokens ou recupera dados do perfil do usuário com bibliotecas de desenvolvedores OIDC ou 2.0. OAuth

2. A API de grupos de usuários do Amazon Cognito é um conjunto de ferramentas para sua aplicação web ou aplicativo móvel autenticar usuários após coletar informações de login em seu próprio frontend personalizado. A autenticação da API de grupos de usuários produz os tokens web JSON a seguir.

a. Um token de identidade com declarações de atributos verificáveis do usuário.
b. Um token de acesso que autoriza o usuário a criar solicitações de API autorizadas por token para um endpoint de serviço da AWS.

> **Note**
>
> Por padrão, os tokens de acesso da autenticação da API de grupos de usuários contêm apenas o escopo `aws.cognito.signin.user.admin`. Para gerar um token de acesso com escopos adicionais, por exemplo, para autorizar uma solicitação para uma API de terceiros, solicite os escopos durante a autenticação por meio dos endpoints do grupo de usuários ou adicione escopos personalizados em um Acionador do Lambda antes da geração do token. A personalização do token de acesso adiciona custos à sua AWS fatura.

c. Um token de atualização que autoriza solicitações de novos tokens de ID e acesso e atualiza a identidade do usuário e as propriedades de controle de acesso.

Você pode vincular um usuário federado, que normalmente faria login por meio dos endpoints de grupos de usuários, a um usuário cujo perfil seja local para sua lista de usuários. Um usuário local existe exclusivamente em seu diretório de grupo de usuários sem federação por meio de um IdP externo. Se você vincular sua identidade federada a um usuário local em uma solicitação de AdminLinkProviderForUserAPI, ele poderá fazer login com a API de grupos de usuários. Para obter mais informações, consulte Vincular usuários federados a um perfil de usuário existente.

A API de grupos de usuários do Amazon Cognito tem duplo propósito.

1. Ela cria e configura os recursos de grupos de usuários do Amazon Cognito. Por exemplo, você pode criar grupos de usuários, adicionar AWS Lambda acionadores e configurar o domínio do grupo de usuários que hospeda suas páginas de login gerenciadas.
2. Ela realiza operações de cadastro, login e outras operações para usuários locais e vinculados.

Exemplo de cenário com a API de grupos de usuários do Amazon Cognito

1. O usuário seleciona o botão “Criar uma conta” que você criou na aplicação. Ele inserem um endereço de e-mail e uma senha.
2. Seu aplicativo envia uma solicitação de SignUpAPI e cria um novo usuário no seu grupo de usuários.

3. A aplicação solicita que o usuário forneça um código de confirmação enviado por e-mail. O usuário insere o código que recebeu em uma mensagem de e-mail.
4. Seu aplicativo envia uma solicitação de ConfirmSignUpAPI com o código de confirmação do usuário.
5. A aplicação solicita que o usuário informe o nome de usuário e a senha, e ele insere essas informações.
6. Seu aplicativo envia uma solicitação de InitiateAuthAPI e armazena um token de ID, um token de acesso e um token de atualização. A aplicação chama as bibliotecas do OIDC para gerenciar os tokens do usuário e manter uma sessão persistente para esse usuário.

Na API de grupos de usuários do Amazon Cognito, você não pode conectar usuários que se federam por meio de um IdP. É necessário autenticar esses usuários por meio dos endpoints de grupo de usuários. Para ter mais informações sobre os endpoints do grupo de usuários que incluem login gerenciado, consulte Referência de login gerenciado e endpoints do grupo de usuários.

Os usuários federados podem começar no login gerenciado e selecionar o IdP deles ou você pode ignorar o login gerenciado e enviar os usuários diretamente ao seu IdP para fazer login. Quando a solicitação de API para Autorizar endpoint inclui um parâmetro de IdP, o Amazon Cognito redireciona silenciosamente o usuário para a página de login do IdP.

Exemplo de cenário com páginas de login gerenciado

1. O usuário seleciona o botão “Criar uma conta” que você criou na aplicação.
2. O login gerenciado apresenta ao usuário uma lista dos provedores de identidades social nos quais você registrou as credenciais de desenvolvedor. O usuário escolhe a Apple.
3. A aplicação inicia uma solicitação para Autorizar endpoint com o nome do provedor `SignInWithApple`.
4. O navegador do usuário abre a página de autenticação da Apple. O usuário faz login e opta por autorizar que o Amazon Cognito leia as informações do perfil dele.
5. O Amazon Cognito confirma o token de acesso da Apple e consulta o perfil Apple do usuário.
6. O usuário apresenta um código de autorização do Amazon Cognito para a aplicação.
7. A biblioteca OIDC na aplicação troca o código de autorização com o Endpoint de token e armazena um token de ID, token de acesso e token de atualização emitidos pelo grupo de usuários. A aplicação usa bibliotecas do OIDC para gerenciar os tokens do usuário e manter uma sessão persistente para esse usuário.

A API de grupo de usuários e as páginas de login gerenciado são compatíveis com uma variedade de cenários descritos neste guia. As seções a seguir examinam como a API de grupos de usuários se divide ainda mais em classes que atendem aos seus requisitos de inscrição, login e gerenciamento de recursos.

Lista de operações de API agrupadas por modelo de autorização

A API de grupos de usuários do Amazon Cognito, tanto uma interface de gerenciamento de recursos quanto uma interface de autenticação e autorização voltada para o usuário, combina os modelos de autorização a seguir nas respectivas operações. Dependendo da operação da API, talvez seja necessário fornecer autorização com credenciais do IAM, um token de acesso, um token de sessão, um segredo do cliente ou uma combinação deles. Para muitas operações de autenticação e autorização de usuários, você pode escolher entre versões autenticadas e não autenticadas da solicitação. Operações não autenticadas são a prática recomendada de segurança para aplicações que você distribui para os usuários, como aplicações móveis; não é necessário incluir nenhum segredo no código.

Você pode atribuir permissões nas políticas do IAM somente para Operações de gerenciamento autorizadas pelo IAM e Operações de usuário autorizadas pelo IAM.

Operações de gerenciamento autorizadas pelo IAM

As operações de gerenciamento autorizadas pelo IAM permitem modificar e exibir a configuração do grupo de usuários e do cliente da aplicação, da mesma forma que você faria no Console de gerenciamento da AWS.

Por exemplo, para modificar seu grupo de usuários em uma solicitação de UpdateUserPoolAPI, você deve apresentar AWS credenciais e permissões do IAM para atualizar o recurso.

Para autorizar essas solicitações no AWS Command Line Interface (AWS CLI) ou em um AWS SDK, configure seu ambiente com variáveis de ambiente ou configuração de cliente que adicionem credenciais do IAM à sua solicitação. Para obter mais informações, consulte Acessando AWS usando suas AWS credenciais no Referência geral da AWS. Você também pode enviar solicitações diretamente aos endpoints de serviço da API de grupos de usuários do Amazon Cognito. Você deve autorizar ou assinar essas solicitações com AWS as credenciais que você incorpora no cabeçalho da sua solicitação. Para obter mais informações, consulte Assinatura de solicitações AWS da API.

Operações de gerenciamento autorizadas pelo IAM

AddCustomAttributes

CreateGroup

CreateIdentityProvider

CreateResourceServer

CreateUserImportJob

CreateUserPool

CreateUserPoolClient

CreateUserPoolDomain

DeleteGroup

DeleteIdentityProvider

DeleteResourceServer

DeleteUserPool

DeleteUserPoolClient

DeleteUserPoolDomain

DescribeIdentityProvider

DescribeResourceServer

DescribeRiskConfiguration

DescribeUserImportJob

DescribeUserPool

DescribeUserPoolClient

DescribeUserPoolDomain

Operações de gerenciamento autorizadas pelo IAM

Obtenha CSVHeader

GetGroup

GetIdentityProviderByIdentifier

GetSigningCertificate

Obtenha UICustomization

GetUserPoolMfaConfig

ListGroups

ListIdentityProviders

ListResourceServers

ListTagsForResource

ListUserImportJobs

ListUserPoolClients

ListUserPools

ListUsers

ListUsersInGroup

SetRiskConfiguration

Conjunto UICustomization

SetUserPoolMfaConfig

StartUserImportJob

StopUserImportJob

TagResource

Operações de gerenciamento autorizadas pelo IAM

UntagResource

UpdateGroup

UpdateIdentityProvider

UpdateResourceServer

UpdateUserPool

UpdateUserPoolClient

UpdateUserPoolDomain

Operações de usuário autorizadas pelo IAM

As operações de usuário autorizadas pelo IAM permitem o cadastro, o login, o gerenciamento de credenciais, a modificação e a exibição dos usuários.

Por exemplo, você pode ter um nível de aplicação do lado do servidor que oferece suporte a um front-end da Web. Seu aplicativo do lado do servidor é um cliente OAuth confidencial no qual você confia com acesso privilegiado aos recursos do Amazon Cognito. Para registrar um usuário no aplicativo, seu servidor pode incluir AWS credenciais em uma solicitação de AdminCreateUserAPI. Para obter mais informações sobre os tipos de OAuth clientes, consulte Tipos de clientes na Estrutura de Autorização OAuth 2.0.

Para autorizar essas solicitações no AWS CLI ou em um AWS SDK, configure seu ambiente de aplicativo do lado do servidor com variáveis de ambiente ou configuração de cliente que adicionem credenciais do IAM à sua solicitação. Para obter mais informações, consulte Acessando AWS usando suas AWS credenciais no Referência geral da AWS. Você também pode enviar solicitações diretamente aos endpoints de serviço da API de grupos de usuários do Amazon Cognito. Você deve autorizar ou assinar essas solicitações com AWS as credenciais que você incorpora no cabeçalho da sua solicitação. Para obter mais informações, consulte Assinatura de solicitações AWS da API.

Se o cliente da aplicação tiver um segredo de cliente, você deverá fornecer suas credenciais do IAM e, dependendo da operação, o parâmetro `SecretHash` ou o valor `SECRET_HASH` em `AuthParameters`. Para obter mais informações, consulte Computar valores de hash de segredo.

Operações de usuário autorizadas pelo IAM

AdminAddUserToGroup

AdminConfirmSignUp

AdminCreateUser

AdminDeleteUser

AdminDeleteUserAttributes

AdminDisableProviderForUser

AdminDisableUser

AdminEnableUser

AdminForgetDevice

AdminGetDevice

AdminGetUser

AdminInitiateAuth

AdminLinkProviderForUser

AdminListDevices

AdminListGroupsForUser

AdminListUserAuthEvents

AdminRemoveUserFromGroup

AdminResetUserPassword

AdminRespondToAuthChallenge

AdminSetUserMFAPreference

AdminSetUserPassword

Operações de usuário autorizadas pelo IAM

AdminSetUserSettings

AdminUpdateAuthEventFeedback

AdminUpdateDeviceStatus

AdminUpdateUserAttributes

AdminUserGlobalSignOut

Operações de usuário não autenticadas

Operações de usuário não autenticadas para se inscrever, fazer login e iniciar redefinições de senha para os usuários. Use operações de API não autenticadas ou públicas quando quiser que qualquer pessoa na internet se inscreva e faça login na aplicação.

Por exemplo, para registrar um usuário em seu aplicativo, você pode distribuir um cliente OAuth público que não forneça acesso privilegiado aos segredos. Você pode registrar esse usuário com a operação de API não autenticada. SignUp

Para enviar essas solicitações em um cliente público que você desenvolveu com um AWS SDK, você não precisa configurar nenhuma credencial. Você também pode enviar solicitações diretamente aos endpoints de serviço da API de grupos de usuários do Amazon Cognito sem autorização adicional.

Se o cliente da aplicação tiver um segredo de cliente, você deverá fornecer, dependendo da operação, o parâmetro `SecretHash` ou o valor `SECRET_HASH` em `AuthParameters`. Para obter mais informações, consulte Computar valores de hash de segredo.

Operações de usuário não autenticadas

SignUp

ConfirmSignUp

ResendConfirmationCode

ForgotPassword

Operações de usuário não autenticadas

ConfirmForgotPassword

InitiateAuth

Operações de usuário autorizadas por token

As operações de usuário autorizadas por token terminam a sessão, gerenciam as credenciais, modificam e visualizam os usuários após eles fazerem login ou iniciarem o processo de login. Use operações de API autorizadas por token quando não quiser distribuir segredos na aplicação e quiser autorizar solicitações com as credenciais do seu próprio usuário. Se o usuário tiver concluído o login, você deverá autorizar a solicitação de API autorizada por token com um token de acesso. Se o usuário estiver no meio de um processo de login, você deverá autorizar a solicitação de API autorizada por token com um token de sessão que o Amazon Cognito retornou em resposta à solicitação anterior.

Por exemplo, em um cliente público, talvez você queira atualizar o perfil de um usuário de uma forma que restrinja o acesso de gravação somente ao próprio perfil do usuário. Para fazer essa atualização, seu cliente pode incluir o token de acesso do usuário em uma solicitação de UpdateUserAttributesAPI.

Para enviar essas solicitações em um cliente público que você desenvolveu com um AWS SDK, você não precisa configurar nenhuma credencial. Inclua um parâmetro `AccessToken` ou `Session` na solicitação. Você também pode enviar solicitações diretamente aos endpoints de serviço da API de grupos de usuários do Amazon Cognito. Para autorizar uma solicitação para um endpoint de serviço, inclua o token de acesso ou sessão no corpo POST da solicitação.

Para assinar uma solicitação de API para uma operação autorizada por token, inclua o token de acesso como cabeçalho `Authorization` na solicitação, no formato `Bearer` *`<Base64-encoded access token>`*.

Operações de usuário autorizadas por token	AccessToken	Sessão
RespondToAuthChallenge		✓

Operações de usuário autorizadas por token	AccessToken	Sessão
ChangePassword	✓	
GetUser	✓	
StartWebAuthnRegistration	✓	
CompleteWebAuthnRegistration	✓	
DeleteWebAuthnCredential	✓	
ListWebAuthnCredentials	✓	
UpdateUserAttributes	✓	
DeleteUserAttributes	✓	
DeleteUser	✓	
ConfirmDevice	✓	
ForgetDevice	✓	
GetDevice	✓	
ListDevices	✓	
UpdateDeviceStatus	✓	
GetUserAttributeVerificationCode	✓	
VerifyUserAttribute	✓	
SetUserSettings	✓	

Operações de usuário autorizadas por token	AccessToken	Sessão
SetUserMFAPreference	✓	
GlobalSignOut	✓	
UpdateAuthEventFeedback		✓
AssociateSoftwareToken	✓	✓
VerifySoftwareToken	✓	✓
RevokeToken[1]		
GetTokensFromRefreshToken[1]		

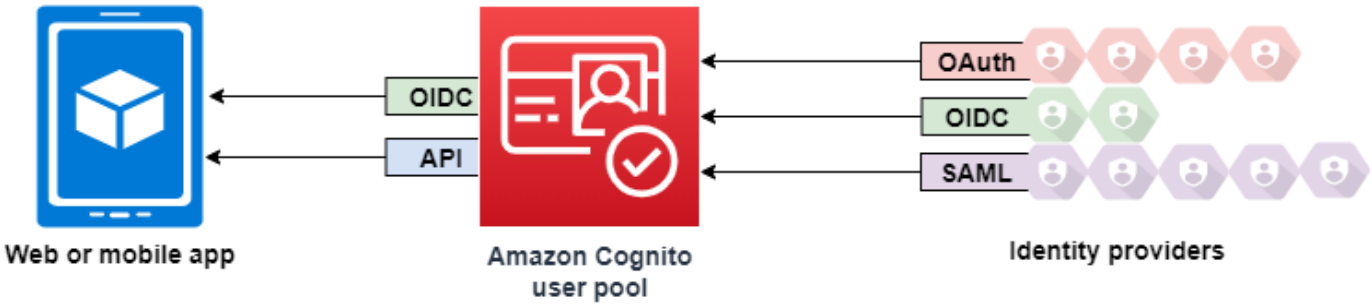
[1] `RevokeToken` e `GetTokensFromRefreshToken` usam tokens de atualização como parâmetro de autorização. O token de atualização serve como token de autorização e como recurso de destino.

Login do grupo de usuários com provedores de identidades de terceiros

Os usuários da aplicação podem fazer login diretamente por meio de um grupo de usuários ou federar por meio de um provedor de identidades (IdP) de terceiros. O grupo de usuários gerencia a sobrecarga de lidar com os tokens que são retornados do login social por meio do Facebook, Google, Amazon e Apple, e do OpenID Connect (OIDC) e SAML. IdPs Com a interface web hospedada integrada, o Amazon Cognito fornece gerenciamento e gerenciamento de tokens para usuários autenticados de todos. IdPs Dessa forma, os sistemas de backend podem realizar a padronização com base em um conjunto de tokens do grupo de usuários.

Como o login federado funciona em grupos de usuários do Amazon Cognito

O login por meio de um terceiro (federação) está disponível em grupos de usuários do Amazon Cognito. Esse recurso é independente da federação nos grupos de identidades do Amazon Cognito (identidades federadas).



O Amazon Cognito é um diretório de usuários e um provedor de identidade (IdP) OAuth 2.0. Quando você faz login de usuários locais no diretório do Amazon Cognito, seu grupo de usuários é um IdP para sua aplicação. Um usuário local existe exclusivamente em seu diretório de grupo de usuários sem federação por meio de um IdP externo.

Quando você conecta o Amazon Cognito às redes sociais, SAML ou OpenID Connect (OIDC IdPs), seu grupo de usuários atua como uma ponte entre vários provedores de serviços e seu aplicativo. Para o IdP, o Amazon Cognito é um provedor de serviços (SP). Você IdPs passa um token de ID do OIDC ou uma declaração SAML para o Amazon Cognito. O Amazon Cognito lê as reivindicações sobre seu usuário no token ou na afirmação e mapeia essas reivindicações para um novo perfil de usuário no diretório do grupo de usuários.

Depois, o Amazon Cognito cria um perfil para o usuário federado em seu próprio diretório. O Amazon Cognito adiciona atributos ao usuário com base nas reivindicações do seu IdP e, no caso do OIDC e de provedores de identidades sociais, um endpoint `userinfo` público operado pelo IdP. Os atributos do usuário mudam em seu grupo de usuários quando um atributo do IdP mapeado é alterado. Você também pode adicionar mais atributos independentes dos do IdP.

Depois que o Amazon Cognito cria um perfil para seu usuário federado, ele altera sua função e se apresenta como o IdP para sua aplicação, que agora é o SP. O Amazon Cognito é uma combinação de OIDC e 2.0 IdP. OAuth Ele gera tokens de acesso, tokens de ID e tokens de atualização. Para mais informações sobre tokens, consulte Compreendendo os tokens web JSON do grupo de usuários () JWTs.

É necessário criar uma aplicação que se integre ao Amazon Cognito para autenticar e autorizar os usuários, sejam eles federados ou locais.

As responsabilidades de uma aplicação como provedor de serviços do Amazon Cognito

Confirmar e processar as informações nos tokens

Na maioria dos casos, o Amazon Cognito redireciona seu usuário autenticado para um URL de aplicação que ele anexa com um código de autorização. Sua aplicação troca o código por tokens de acesso, ID e atualização. Depois, ela precisa conferir a validade dos tokens e fornecer informações ao usuário com base nas reivindicações contidas nos tokens.

Responder a eventos de autenticação com solicitações da API do Amazon Cognito

Sua aplicação precisa se integrar à API de grupos de usuários do Amazon Cognito e aos endpoints de API de autenticação. A API de autenticação conecta e desconecta o usuário e gerencia tokens. A API de grupos de usuários tem uma variedade de operações que gerenciam seu grupo de usuários, seus usuários e a segurança do ambiente de autenticação. Sua aplicação precisa saber o que fazer em seguida ao receber uma resposta do Amazon Cognito.

Fatos a saber sobre o login de terceiro dos grupos de usuários do Amazon Cognito

- Se quiser que seus usuários façam login com provedores federados, você deve escolher um domínio. Isso configura as páginas para login gerenciado. Para obter mais informações, consulte Usar o próprio domínio para fazer login gerenciado.
- Você não pode cadastrar usuários federados com operações de API como InitiateAuthe. AdminInitiateAuth Os usuários federados só podem fazer login com o Endpoint de login ou o Autorizar endpoint.
- O Autorizar endpoint é um endpoint de redirecionamento. Se você fornecer um parâmetro `idp_identifier` ou `identity_provider` na solicitação, ela será redirecionada silenciosamente para o IdP, ignorando o login gerenciado. Caso contrário, ela será redirecionada para o Endpoint de login do login gerenciado.
- Quando o login gerenciado redireciona uma sessão para um IdP federado, o Amazon Cognito inclui o cabeçalho `user-agent` `Amazon/Cognito` na solicitação.

- O Amazon Cognito gera o atributo `username` para um perfil de usuário federado usando a combinação de um identificador fixo com o nome de seu IdP. Para gerar um nome de usuário que corresponda aos seus requisitos personalizados, crie um mapeamento para o atributo `preferred_username`. Para obter mais informações, consulte Coisas a saber sobre mapeamentos.

  Exemplo: `MyIDP_bob@example.com`
- O Amazon Cognito cria um grupo de usuários para cada OIDC SAMl e IdP social que você adiciona ao seu grupo de usuários. O nome do grupo está no formato `[user pool ID]_[IdP name]`, por exemplo, `us-east-1_EXAMPLE_MYSSO` ou `us-east-1_EXAMPLE_Google`. Cada perfil de usuário exclusivo do IdP gerado automaticamente é adicionado automaticamente a esse grupo. Os usuários vinculados não são adicionados automaticamente a esse grupo, mas você pode adicionar seus perfis ao grupo em um processo separado.
- O Amazon Cognito registra informações sobre a identidade de seu usuário federado em um atributo e uma reivindicação no token de ID, chamada `identities`. Essa reivindicação contém o provedor do usuário e o ID exclusivo do provedor. Não é possível alterar o atributo `identities` em um perfil de usuário diretamente. Para obter mais informações sobre como vincular um usuário federado, consulte Vincular usuários federados a um perfil de usuário existente.
- Quando você atualiza o IdP em uma solicitação de API UpdateIdentityProvider, as alterações podem levar até 1 minuto para aparecerem no login gerenciado.
- O Amazon Cognito é compatível com até 20 redirecionamentos HTTP entre ele e o IdP.
- Quando o usuário faz login com o login gerenciado, o navegador armazena um cookie de sessão de login criptografado que registra o cliente e o provedor com os quais ele fez login. Se ele tentar fazer login novamente com os mesmos parâmetros, o login gerenciado reutilizará qualquer sessão existente não expirada, e o usuário se autenticará sem fornecer as credenciais novamente. Se o usuário fizer login novamente com um IdP diferente, incluindo uma mudança para ou do login do grupo de usuários local, ele deverá fornecer credenciais e gerar uma sessão de login.

  Você pode atribuir qualquer parte do seu grupo de usuários IdPs a qualquer cliente de aplicativo, e os usuários só podem entrar com um IdP que você atribuiu ao cliente do aplicativo.

Tópicos

- Como configurar provedores de identidade para seu grupo de usuários
- Como usar provedores de identidade social com um grupo de usuários
- Como usar provedores de identidade SAML com um grupo de usuários

- Como usar provedores de identidade OIDC com um grupo de usuários
- Mapeamento de atributos de IdP para perfis e tokens
- Vincular usuários federados a um perfil de usuário existente

Como configurar provedores de identidade para seu grupo de usuários

Com grupos de usuários, você pode implementar o login por meio de uma variedade de provedores de identidade externos ()IdPs. Esta seção do guia tem instruções para configurar esses provedores de identidade com seu grupo de usuários no console do Amazon Cognito. Como alternativa, você pode usar a API de grupos de usuários e um AWS SDK para adicionar programaticamente provedores de identidade de grupos de usuários. Para obter mais informações, consulte CreateIdentityProvider.

As opções de provedores de identidade compatíveis incluem provedores sociais, como Facebook, Google e Amazon, e os provedores OpenID Connect (OIDC) e SAML 2.0. Antes de começar, configure suas credenciais administrativas para seu IdP. Para cada tipo de provedor, você precisará registrar a aplicação, obter as credenciais necessárias e, em seguida, configurar os detalhes do provedor em seu grupo de usuários. Seus usuários podem então se inscrever e acessar a aplicação com as contas existentes dos provedores de identidade conectados.

O menu Provedores sociais e externos em Autenticação adiciona e atualiza o grupo de usuários IdPs. Para obter mais informações, consulte Login do grupo de usuários com provedores de identidades de terceiros.

Tópicos

- Configurar o acesso do usuário com um IdP social
- Configurar o login do usuário com um IdP OIDC
- Configurar o login do usuário com um IdP SAML

Configurar o acesso do usuário com um IdP social

É possível usar a federação para integrar grupos de usuários do Amazon Cognito com provedores de identidade social, como o Facebook, o Google e o Login with Amazon.

Para adicionar um provedor de identidade social, primeiro é necessário criar uma conta de desenvolvedor com o provedor de identidade. Depois de criar sua conta de desenvolvedor, registre a aplicação no provedor de identidade. O provedor de identidade cria um ID da aplicação e um

segredo para a aplicação e esses valores são configurados no grupo de usuários do Amazon Cognito.

- Plataforma de identidade Google
- Facebook para desenvolvedores
- Login da Amazon
- Fazer login com a Apple

Como integrar o login do usuário a um IdP social

1. Faça login no console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. No painel de navegação, escolha User Pools (Grupos de usuários) e escolha o grupo de usuários que deseja editar.
3. Clique no menu Provedores sociais e externos.
4. Escolha Add an identity provider (Adicionar um provedor de identidade) ou o provedor de identidade Facebook, Google, Amazon ou Apple que você configurou, localize Identity provider information (Informações do provedor de identidade) e escolha Edit (Editar). Para obter mais informações sobre como adicionar provedores de identidade social, consulte Como usar provedores de identidade social com um grupo de usuários.
5. Insira as informações do provedor de identidade social concluindo uma das etapas a seguir, com base em sua escolha de IdP:

   Facebook, Google e Login with Amazon

   Insira o ID e o segredo da aplicação recebidos ao criar a aplicação cliente.

   Sign in with Apple

   Insira o ID de serviço fornecido à Apple, bem como o ID de equipe, o ID de chave e a chave privada recebidos ao criar o cliente da aplicação.
6. Para Authorized scopes (Escopos autorizados), insira os nomes dos escopos do provedor de identidade social que deseja mapear aos atributos do grupo de usuários. Os escopos definem quais atributos do usuário, como nome e e-mail, você deseja acessar com a aplicação. Ao inserir escopos, use as seguintes diretrizes com base em sua escolha de IdP:

   - Facebook: separe os escopos com vírgulas. Por exemplo:

     ```
     public_profile, email
     ```

- Google, Login with Amazon e Sign in with Apple: separe os escopos com espaços. Por exemplo:
  - Google: `profile email openid`
  - Login with Amazon: `profile postal_code`
  - Sign in with Apple: `name email`

> **Note**
>
> Para Sign In with Apple (console), use as caixas de seleção para selecionar os escopos.

7. Escolha Salvar alterações.
8. No menu Clientes da aplicação, escolha um cliente da aplicação na lista e clique em Editar. Adicione o novo provedor de identidade social ao cliente da aplicação em Identity providers (Provedores de identidade).
9. Escolha Salvar alterações.

Para obter mais informações sobre redes sociais IdPs, consulteComo usar provedores de identidade social com um grupo de usuários.

Configurar o login do usuário com um IdP OIDC

É possível integrar o login do usuário a um provedor de identidade OpenID Connect (OIDC), como Salesforce ou Ping Identity.

Como adicionar um provedor OIDC a um grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários) no menu de navegação.
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Provedores sociais e externos e selecione Adicionar um provedor de identidade.
5. Escolha um provedor de identidade OpenID Connect.
6. Insira um nome exclusivo em Provider name (Nome do provedor).
7. Insira o ID do cliente que você recebeu do provedor em Client ID (ID do cliente).
8. Insira o segredo do cliente que você recebeu do provedor em Client secret (Segredo do cliente).

9. Insira os Authorized scopes (Escopos autorizados) para esse provedor. Os escopos definem quais grupos de atributos do usuário (como `name` e `email`) sua aplicação solicitará ao seu provedor. Os escopos devem ser separados por espaços, seguindo a especificação OAuth 2.0.

   O usuário deve receber consentimento para o fornecimento desses atributos à aplicação.

10. Escolha um Attribute request method (Método de solicitação de atributos) para fornecer ao Amazon Cognito o método HTTP (GET ou POST) que ele usa para buscar os detalhes do usuário no endpoint userInfo operado pelo provedor.

11. Escolha um Setup method (Método de configuração) para recuperar endpoints OpenID Connect por meio de Auto fill through issuer URL (Preenchimento automático por meio do URL do emissor) ou Manual input (Entrada manual). Use o preenchimento automático do URL do emissor quando seu provedor tiver um `.well-known/openid-configuration` endpoint público em que o Amazon Cognito possa recuperar URLs os endpoints`token`,, e. `authorization` `userInfo` `jwks_uri`

12. Insira o URL do emissor ou`authorization`,, `token``userInfo`, e o `jwks_uri` endpoint URLs do seu IdP.

    > **Note**
    >
    > Você pode usar somente os números de porta 443 e 80 com descoberta, preenchida automaticamente e inserida manualmente. URLs Os logins de usuários falharão se o provedor OIDC usar qualquer porta TCP não padrão.
    > O URL do emissor deve começar com `https://` e não pode terminar com o caractere `/`. Por exemplo, Salesforce usa este URL: `https://login.salesforce.com`
    > O `openid-configuration` documento associado ao URL do emissor deve fornecer HTTPS URLs para os seguintes valores: `authorization_endpoint``token_endpoint`,`userinfo_endpoint`, e. `jwks_uri`
    > Da mesma forma, ao escolher Entrada manual, você só pode inserir HTTPS URLs.

13. Por padrão, a declaração OIDC sub é mapeada para o atributo de grupo de usuários Username (Nome de usuário). Você pode mapear outras solicitações OIDC para atributos de grupo de usuários. Insira a solicitação OIDC e selecione o atributo de grupo de usuários correspondente na lista suspensa. Por exemplo, a solicitação email geralmente é mapeada para o atributo de grupo de usuários E-mail.

14. Mapeie atributos adicionais do provedor de identidade ao seu grupo de usuários. Para mais informações, consulte Especificar mapeamentos de atributos do provedor de identidade para o grupo de usuários.
15. Escolha Criar.
16. No menu Clientes da aplicação, selecione um cliente da aplicação na lista. Para adicionar o novo provedor de identidades SAML ao cliente da aplicação, navegue até a guia Páginas de login e selecione Editar em Configuração gerenciada de páginas de login.
17. Escolha Salvar alterações.

Para obter mais informações sobre o OIDC IdPs, consulte. Como usar provedores de identidade OIDC com um grupo de usuários

Configurar o login do usuário com um IdP SAML

Você pode usar a federação para que os grupos de usuários do Amazon Cognito se integrem a um provedor de identidade (IdP) SAML. Forneça um documento de metadados, fazendo upload do arquivo ou inserindo um URL do endpoint de documento de metadados. Para obter informações sobre como obter documentos de metadados para SAML de terceiros IdPs, consulte. Como configurar seu provedor de identidades SAML de terceiros

Para configurar um provedor de identidade SAML 2.0 no seu grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Provedores sociais e externos e selecione Adicionar um provedor de identidade.
5. Escolha um provedor de identidade SAML.
6. Insira Identifiers (Identificadores) separados por vírgulas. Um identificador direciona o Amazon Cognito para que ele confira o endereço de e-mail de login do usuário e, depois, direciona o usuário para o provedor que corresponde ao domínio dele.
7. Escolha Add sign-out flow (Adicionar fluxo de desconexão) se quiser que o Amazon Cognito envie solicitações de desconexão assinadas ao seu provedor quando um usuário se desconectar. Configure o provedor de identidades SAML 2.0 para enviar respostas de logout para o endpoint `https://mydomain.auth.us-east-1.amazoncognito.com/saml2/logout` que o Amazon Cognito cria quando você configura o login gerenciado. O endpoint `saml2/logout` usa uma associação POST.

> **Note**
>
> Se você selecionar essa opção e seu provedor de identidade SAML esperar uma solicitação de logout assinada, você também precisará configurar o certificado de assinatura fornecido pelo Amazon Cognito com seu IdP SAML.
> O IdP SAML processará a solicitação de logout assinada e fará logout do seu usuário da sessão do Amazon Cognito.

8. Selecione uma Metadata document source (Fonte de documento de metadados). Se seu provedor de identidade oferecer metadados SAML em um URL público, você pode escolher Metadata document URL (URL do documento de metadados) e inserir esse URL público. Do contrário, escolha Upload metadata document (Carregar documento de metadados) e, em seguida, um arquivo de metadados que você tenha baixado de seu provedor anteriormente.

   > **Note**
   >
   > Se seu provedor tiver um endpoint público, recomendamos que você insira um URL do documento de metadados em vez de carregar um arquivo. Se você usar o URL, o Amazon Cognito atualizará os metadados automaticamente. Normalmente, a atualização de metadados ocorre a cada seis horas ou antes de os metadados expirarem, o que ocorrer primeiro.

9. Escolha Map attributes between your SAML provider and your app (Mapear atributos entre seu provedor SAML e sua aplicação) para mapear atributos do provedor SAML para o perfil de usuário em seu grupo de usuários. Inclua os atributos obrigatórios do grupo de usuários no mapa de atributos.

   Por exemplo, quando escolher o User pool attribute (Atributo do grupo de usuários) `email`, insira o nome do atributo SAML como ele aparece na afirmação SAML de seu provedor de identidade. Seu provedor de identidade pode oferecer exemplos de afirmações SAML como referência. Alguns provedores de identidade usam nomes simples, como `email`, enquanto outros usam nomes de atributos formatados com URL, semelhantes a:

   ```
   http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
   ```

10. Escolha Criar.

> **Note**
>
> Se você vir `InvalidParameterException` durante a criação de um IdP SAML com um URL do endpoint de metadados HTTPS, verifique se o endpoint de metadados está com o SSL configurado corretamente e se há um certificado SSL válido associado a ele. Um exemplo dessa exceção seria “Erro ao recuperar metadados de*<metadata endpoint>*”.

Para configurar o IdP SAML para adicionar um certificado de assinatura

- Para obter o certificado contendo a chave pública que o IdP usa para verificar a solicitação de logout assinada, faça o seguinte:
  1. Navegue até o menu Provedores sociais e externos do grupo de usuários.
  2. Selecione seu provedor SAML.
  3. Clique em Ver certificado de assinatura.

Para obter mais informações sobre SAML, IdPs consulte[Como usar provedores de identidade SAML com um grupo de usuários](#).

Como usar provedores de identidade social com um grupo de usuários

Os usuários de aplicativos web e móveis podem fazer login por meio de provedores de identidade social (IdP), como o Facebook, o Google, a Amazon e a Apple. Com a interface do usuário da Web hospedada integrada, o Amazon Cognito fornece manuseio e gerenciamento de tokens para todos os usuários autenticados. Dessa forma, os sistemas de backend podem realizar a padronização com base em um conjunto de tokens do grupo de usuários. Você deve habilitar o login gerenciado para se integrar com provedores de identidades social compatíveis. Quando o Amazon Cognito cria suas páginas de login gerenciadas, ele cria endpoints OAuth 2.0 que o Amazon Cognito, seu OIDC e redes sociais usam para trocar informações. IdPs Para mais informações, consulte [Referência da API de autenticação dos grupos de usuários do Amazon Cognito](#).

Você pode adicionar um IdP social no Console de gerenciamento da AWS, ou você pode usar a AWS CLI ou a API do Amazon Cognito.

> **Note**
>
> O login por meio de um terceiro (federação) está disponível em grupos de usuários do Amazon Cognito. Esse recurso é independente da federação nos grupos de identidades do Amazon Cognito (identidades federadas).

Tópicos

Configurar uma conta de desenvolvedor e uma aplicação de IdP social

Antes de criar um IdP social com o Amazon Cognito, é necessário registrar sua aplicação no IdP social para receber um ID do cliente e a chave secreta do cliente.

Facebook

Para obter as informações mais recentes sobre configuração de contas de desenvolvedor e autenticação da Meta, consulte Desenvolvimento de apps com a Meta.

Como registrar uma aplicação com o Facebook/Meta

1. Crie uma conta de desenvolvedor com o Facebook.
2. Faça login com as credenciais do Facebook.
3. No menu My Apps (Meus aplicativos), escolha Create New App (Criar novo aplicativo).
4. Insira um nome para sua aplicação do Facebook e, em seguida, escolha Create App ID (Criar ID da aplicação).
5. Na barra de navegação à esquerda, escolha Settings (Configurações) e, em seguida, Basic (Básico).
6. Anote o App ID (ID do aplicativo) e a App Secret (Chave secreta do aplicativo). Você poderá usá-los na próxima seção.
7. Escolha + Add Platform (Adicionar plataforma) na parte inferior da página.
8. Escolha Website.

9. Em Website (Site da Web), insira o caminho para a página de acesso da aplicação em Site URL (URL do site).

```
https://mydomain.auth.us-east-1.amazoncognito.com/login?
response_type=code&client_id=1example23456789&redirect_uri=https://
www.example.com
```

10. Escolha Salvar alterações.
11. Insira o caminho para a raiz do domínio do grupo de usuários em App Domains (Domínios da aplicação).

```
https://mydomain.auth.us-east-1.amazoncognito.com
```

12. Escolha Salvar alterações.
13. Na barra de navegação, escolha Add Product (Adicionar produto) e escolha Set up (Configurar) para o produto Facebook Login (Login do Facebook).
14. Na barra de navegação, escolha Facebook Login (Login do Facebook) e Settings (Configurações).

    Insira o caminho para o `/oauth2/idpresponse` endpoint do seu domínio do grupo de usuários em Valid OAuth URIs Redirect.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/idpresponse
```

15. Escolha Salvar alterações.

Login with Amazon

Para obter as informações mais recentes sobre a configuração das contas de desenvolvedor e autenticação do Login with Amazon, consulte Login with Amazon Documentation.

Como registrar uma aplicação com o Login with Amazon

1. Crie uma conta de desenvolvedor com a Amazon.
2. Faça login com as credenciais da Amazon.
3. Você precisa criar um perfil de segurança da Amazon para receber o ID do cliente e a chave secreta do cliente da Amazon.

Selecione Apps and Services (Aplicativos e serviços) na barra de navegação na parte superior da página e, em seguida, selecione Login with Amazon (Login com a Amazon).

4. Escolha Create a Security Profile (Criar um perfil de segurança).
5. Insira o Security Profile Name (Nome do perfil de segurança), Security Profile Description (Descrição do perfil de segurança) e um Consent Privacy Notice URL (URL de notificação de consentimento de privacidade).
6. Escolha Save (Salvar).
7. Selecione Client ID (ID de cliente) e Client Secret (Segredo de cliente) para mostrar o ID e o segredo do cliente. Você poderá usá-los na próxima seção.
8. Passe o cursor sobre o ícone de engrenagem e escolha Web Settings (Configurações da Web) e, em seguida, escolha Edit (Editar).
9. Insira o domínio do grupo de usuários em Allowed Origins (Origens permitidas).

```
https://mydomain.auth.us-east-1.amazoncognito.com
```

10. Insira seu domínio do grupo de usuários com o `/oauth2/idpresponse` endpoint em Allowed Return URLs.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/idpresponse
```

11. Escolha Salvar.

Google

Para obter mais informações sobre OAuth 2.0 na plataforma Google Cloud, consulte Saiba mais sobre autenticação e autorização na documentação do Google Workspace for Developers.

Como registrar uma aplicação com o Google

1. Crie uma conta de desenvolvedor com o Google.
2. Faça login no Console do Google Cloud Platform.
3. Na barra de navegação superior, escolha Select a project (Selecionar um projeto). Se você já tiver um projeto na plataforma do Google, esse menu exibirá seu projeto padrão.
4. Selecione NEW PROJECT (Novo projeto).
5. Insira um nome para o produto e, depois, escolha CREATE (Criar).

6. Na barra de navegação esquerda, escolha Serviços APIs e, em seguida, tela de consentimento do Oauth.
7. Insira as informações da aplicação, um App domain (Domínio da aplicação), Authorized domains (Domínios autorizados) e Developer contact information (Informações de contato do desenvolvedor). Seus Authorized domains (Domínios autorizados) devem incluir `amazoncognito.com` e a raiz de seu domínio personalizado; por exemplo, `example.com`. Escolha SAVE AND CONTINUE (Salvar e continuar).
8. 1. Em Escopos, escolha Adicionar ou remover escopos e escolha, no mínimo, os seguintes OAuth escopos.

    1. `.../auth/userinfo.email`
    2. `.../auth/userinfo.profile`
    3. OpenID
9. Em Test users (Testar usuários), escolha Add Users (Adicionar usuários). Insira seu e-mail e todos os outros usuários de teste autorizados e escolha SAVE AND CONTINUE (Salvar e continuar).
10. Expanda a barra de navegação esquerda novamente e escolha Serviços APIs e, em seguida, Credenciais.
11. Escolha CRIAR CREDENCIAIS e, em seguida, ID OAuth do cliente.
12. Escolha um Application type (Tipo de aplicação) e forneça ao seu cliente um Name (Nome).
13. Em JavaScript Origens autorizadas, escolha ADICIONAR URI. Insira o domínio de seu grupo de usuários.

    ```
    https://mydomain.auth.us-east-1.amazoncognito.com
    ```
14. Em Redirecionamento autorizado URIs, escolha ADICIONAR URI. Insira o caminho para o endpoint `/oauth2/idpresponse` do domínio de seu grupo de usuários.

    ```
    https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/idpresponse
    ```
15. Selecione CREATE (Criar).
16. Armazene com segurança os valores que o Google exibe em Your client ID (Seu ID de cliente) e Your client secret (Seu segredo do cliente). Forneça esses valores ao Amazon Cognito quando você adicionar um IdP do Google.

Sign in with Apple

Para up-to-date obter mais informações sobre como configurar o Login com a Apple, consulte Configurando seu ambiente para fazer login com a Apple na documentação do desenvolvedor da Apple.

Como registrar uma aplicação com o Sign in with Apple (SIWA)

1. Crie uma conta de desenvolvedor com a Apple.
2. Faça login com as credenciais da Apple.
3. Na barra de navegação à esquerda, escolha Certificates, Identifiers & Profiles (Certificados, identificadores e perfis).
4. Na barra de navegação à esquerda, escolha Identifiers (Identificadores).
5. Na página Identifiers (Identificadores), escolha o ícone +.
6. Na página Registrar um novo identificador, escolha Aplicativo e IDs, em seguida, escolha Continuar.
7. Na página Select a type (Selecionar um tipo), escolha App (Aplicação) e, depois, Continue (Continuar).
8. Na página Register an App ID (Registrar ID de uma aplicação), faça o seguinte:
    1. Em Description (Descrição), insira uma descrição.
    2. Em App ID Prefix (Prefixo do ID da aplicação), insira um Bundle ID (ID do pacote). Anote o valor em App ID Prefix (Prefixo do ID da aplicação). Você usará esse valor após escolher a Apple como seu provedor de identidade em Configurar o grupo de usuários com um IdP social.
    3. Em Capabilities (Recursos), escolha Sign In with Apple (Fazer login com a Apple) e, depois, selecione Edit (Editar).
    4. Na página Entrar com a Apple: Configuração do ID do aplicativo, escolha configurar o aplicativo como principal ou agrupado com outro aplicativo IDs e escolha Salvar.
    5. Escolha Continue (Continuar).
9. Na página Confirm your App ID (Confirmar ID do seu app), escolha Register (Registrar).
10. Na página Identifiers (Identificadores), escolha o ícone +.
11. Na página Registrar um novo identificador, escolha Serviços e IDs, em seguida, escolha Continuar.
12. Na página Register a Services ID (Registrar um ID de serviços), faça o seguinte:

1. Em Description (Descrição), digite uma descrição.
2. Em Identifier (Identificador), digite um identificador. Anote esse ID de serviços, pois você precisará desse valor depois de escolher a Apple como provedor de identidades em Configurar o grupo de usuários com um IdP social.
3. Escolha Continue (Continuar) e, depois, Register (Registrar).

13. Escolha o ID de serviços que você acabou de criar na página Identifiers (Identificadores).
    1. Selecione Sign In with Apple (Fazer login com a Apple) e escolha Configure (Configurar).
    2. Na página Web Authentication Configuration (Configuração da autenticação web), selecione o ID da aplicação que você criou anteriormente como o Primary App ID (ID da aplicação principal).
    3. Escolha o ícone + ao lado do site URLs.
    4. Em Domains and subdomains (Domínios e subdomínios), insira o domínio do grupo de usuários sem um prefixo `https://`.

       ```
       mydomain.auth.us-east-1.amazoncognito.com
       ```

    5. Em Return URLs, insira o caminho para o `/oauth2/idpresponse` endpoint do seu domínio do grupo de usuários.

       ```
       https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/idpresponse
       ```

    6. Escolha Next (Próximo) e, depois, selecione Done (Concluído). Não é necessário verificar o domínio.
    7. Escolha Continue (Continuar) e, depois, Save (Salvar).
14. No painel de navegação à esquerda, selecione Keys (Chaves).
15. Na página Keys (Chaves), escolha o ícone +.
16. Na página Register a New Key (Registrar uma chave nova), faça o seguinte:
    1. Em Key Name (Nome da chave), insira um nome de chave.
    2. Escolha Sign In with Apple (Fazer login com a Apple) e escolha Configure (Configurar).
    3. Na página Configure Key (Configurar chave), selecione o ID da aplicação que você criou anteriormente como o Primary App ID (ID da aplicação principal). Escolha Salvar.
    4. Escolha Continue (Continuar) e, depois, Register (Registrar).

17. Na página Download Your Key (Baixe sua chave), escolha Download para baixar a chave privada e anote a Key ID (ID da chave). Em seguida, escolha Done (Concluído). Você precisará dessa chave privada e do valor de Key ID (ID da chave) mostrado nesta página depois de escolher a Apple como provedor de identidade no Configurar o grupo de usuários com um IdP social.

Configurar o grupo de usuários com um IdP social

Para configurar um IdP social do grupo de usuários com o Console de gerenciamento da AWS

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha Grupos de usuários.
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Provedores sociais e externos e selecione Adicionar um provedor de identidade.
5. Escolha um IdP social: Facebook, Google, Login with Amazon ou Sign in with Apple.
6. Escolha entre as seguintes etapas, com base em sua opção de IdP social:

   - Google e Login with Amazon: insira o app client ID (ID do cliente da aplicação) e o app client secret (o segredo do cliente da aplicação) gerado na seção anterior.
   - Facebook: insira o app client ID (ID do cliente da aplicação) e o app client secret (segredo do cliente da aplicação) gerado na seção anterior e, em seguida, escolha uma versão da API (por exemplo, versão 2.12). Recomendamos escolher a versão mais recente possível, já que cada API do Facebook tem um ciclo de vida e uma data de suspensão. Os escopos e atributos do Facebook podem variar entre as versões da API. Recomendamos que você teste seu login de identidade social com o Facebook para confirmar se a federação funciona como pretendido.
   - Sign In with Apple (Fazer login com a Apple): insira o Services ID (ID de serviços), o Team ID (ID de equipe), o Key ID (ID da chave) e a private key (chave privada) gerados na seção anterior.
7. Insira os nomes dos Authorized scopes (Escopos autorizados) que deseja utilizar. Os escopos definem quais atributos do usuário (como `name` e `email`) você deseja acessar com a aplicação. Para o Facebook, eles devem estar separados por vírgulas. Para o Google e o Login with Amazon, eles devem estar separados por espaços. Para Sign in with Apple, marque a caixa de seleção dos escopos que deseja acessar.

Provedor de identidade social	Escopos de exemplo
Facebook	`public_profile, email`
Google	`profile email openid`
Login da Amazon	`profile postal_code`
Fazer login com a Apple	`email name`

O consentimento do usuário da aplicação é solicitado para o fornecimento desses atributos à sua aplicação. Para mais informações sobre os escopos de provedores sociais, consulte a documentação do Google, Facebook, Login with Amazon ou do Sign in with Apple.

Em caso de acesso com Sign in with Apple, a seguir apresentamos os cenários de usuário cujos escopos talvez não sejam retornados:

- Um usuário final encontra falhas depois de sair da página de login com a Apple (elas podem ter origem de falhas internas dentro do Amazon Cognito ou de qualquer elemento escrito pelo desenvolvedor)
- O identificador de ID do serviço é usado em grupos de usuários e and/or outros serviços de autenticação.
- Um desenvolvedor inclui escopos adicionais depois que o usuário final tiver feito o login (sem recuperar novas informações)
- Um desenvolvedor exclui o usuário e, a seguir, o usuário faz login novamente sem remover a aplicação de seu perfil de ID da Apple

8. Mapeie atributos do IdP para o grupo de usuários. Para obter mais informações, consulte Especificar mapeamentos de atributos do provedor de identidade para seu grupo de usuários.
9. Escolha Criar.
10. No menu Clientes da aplicação, selecione um cliente da aplicação na lista. Para adicionar o novo provedor de identidades social ao cliente da aplicação, navegue até a guia Páginas de login e selecione Editar em Configuração gerenciada de páginas de login.
11. Escolha Salvar alterações.

Testar a configuração do IdP social

Na aplicação, você deve invocar um navegador no cliente do usuário para que ele possa fazer login com seu provedor social. Teste o login com seu provedor social após concluir os procedimentos de configuração nas seções anteriores. O exemplo de URL a seguir carrega a página de login do grupo de usuários com um domínio de prefixo.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Esse link é a página para a qual o Amazon Cognito direcionará você ao acessar o menu Clientes da aplicação, selecionar um cliente da aplicação, navegar até a guia Páginas de login e selecionar Visualizar página de login. Para obter mais informações sobre domínios do grupo de usuários, consulte Como configurar um domínio de grupo de usuários. Para obter mais informações sobre clientes de aplicativos, incluindo cliente IDs e retorno de chamada URLs, consulteConfigurações específicas da aplicação com clientes de aplicação.

O link de exemplo a seguir configura o redirecionamento silencioso para um provedor social por meio do Autorizar endpoint com um parâmetro de consulta `identity_provider`. Esse URL ignora o login interativo do grupo de usuários com login gerenciado e leva diretamente à página de login do IdP.

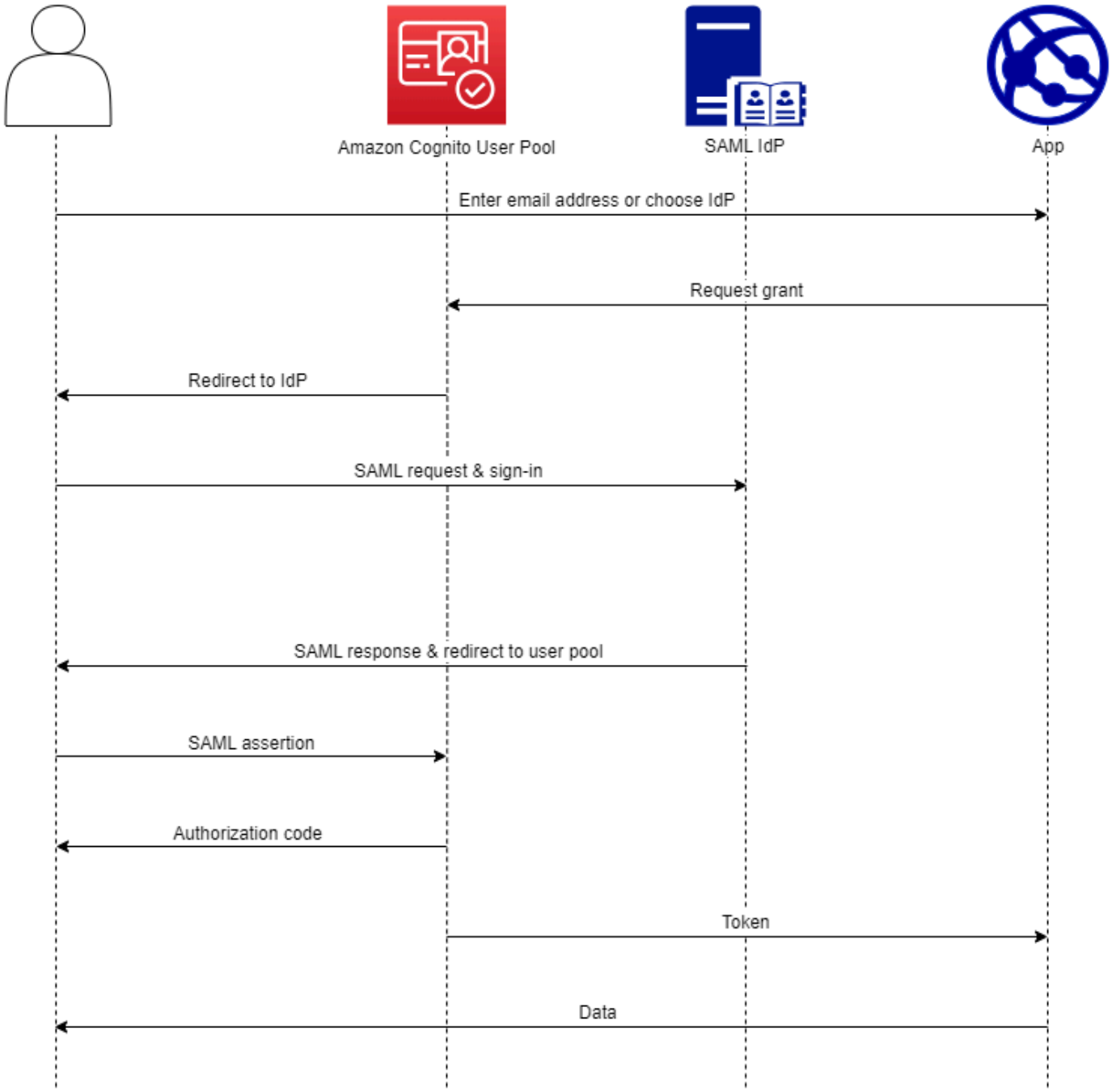
```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/
authorize?identity_provider=Facebook|Google|LoginWithAmazon|
SignInWithApple&response_type=code&client_id=1example23456789&redirect_uri=https://
www.example.com
```

Como usar provedores de identidade SAML com um grupo de usuários

Você pode determinar que os usuários da aplicação web e do aplicativo móvel façam login por meio de um provedor de identidades (IdP) SAML, como o Microsoft Active Directory Federation Services (ADFS) ou o Shibboleth. Você deve escolher um IdP SAML compatível com o padrão SAML 2.0.

Com o login gerenciado, o Amazon Cognito autentica usuários de IdP locais e terceirizados e emite tokens web JSON (). JWTs Com os tokens que o Amazon Cognito emite, você pode consolidar várias fontes de identidade em um padrão universal do OpenID Connect (OIDC) em todas as aplicações. O Amazon Cognito pode processar declarações SAML de seus fornecedores

terceirizados nesse padrão de SSO. Você pode criar e gerenciar um SAML IdP na, por meio da ou com Console de gerenciamento da AWS a API de AWS CLI grupos de usuários do Amazon Cognito. Para criar seu primeiro IdP SAML no Console de gerenciamento da AWS, consulte. Como adicionar e gerenciar provedores de identidade SAML em um grupo de usuários



> **Note**
>
> A federação com login por meio de um IdP de terceiros é um recurso dos grupos de usuários do Amazon Cognito. Os bancos de identidades do Amazon Cognito, às vezes chamados de identidades federadas do Amazon Cognito, são uma implementação da federação que você deve configurar separadamente em cada banco. Um grupo de usuários pode ser um IdP de terceiros para um banco de identidades. Para obter mais informações, consulte Banco de identidades do Amazon Cognito.

Referência rápida para configuração do IdP

É necessário configurar o IdP SAML para aceitar solicitação e enviar respostas ao grupo de usuários. A documentação de seu para IdP SAML conterá informações sobre como adicionar o grupo de usuários como uma aplicação ou terceira parte confiável para o IdP SAML 2.0. A documentação a seguir contém os valores que você deve fornecer para o ID da entidade SP e o URL do Serviço do consumidor de Declaração (ACS).

Referência rápida de valores SAML do grupo de usuários

ID de entidade SP

```
urn:amazon:cognito:sp:us-east-1_EXAMPLE
```

URL DO ACS

```
https://Your user pool domain/saml2/idpresponse
```

Você deve configurar seu grupo de usuários para apoiar o provedor de identidades. As etapas detalhadas para adicionar um IdP SAML externo são as seguintes:

1. Faça o download dos metadados SAML do seu IdP ou recupere o URL para o seu endpoint de metadados. Consulte Como configurar seu provedor de identidades SAML de terceiros.
2. Adicione um novo IdP ao seu grupo de usuários. Faça o upload dos metadados do SAML ou forneça o URL dos metadados. Consulte Como adicionar e gerenciar provedores de identidade SAML em um grupo de usuários.

3. Atribua o IdP aos seus clientes de aplicações. Consulte Configurações específicas da aplicação com clientes de aplicação.

Tópicos

Coisas que você deve saber sobre o SAML IdPs nos grupos de usuários do Amazon Cognito

A implementação de um IdP SAML 2.0 vem com alguns requisitos e restrições. Consulte esta seção ao implementar seu IdP. Você também encontrará informações úteis para solucionar erros durante a federação do SAML com um grupo de usuários.

O Amazon Cognito processa declarações do SAML para você

> Os grupos de usuários do Amazon Cognito são compatíveis com a federação SAML 2.0 com endpoints de pós-vinculação. Isso elimina a necessidade de a aplicação recuperar ou analisar as respostas de afirmação do SAML, pois o grupo de usuários recebe diretamente a resposta do SAML de seu IdP por meio de um agente de usuário. Seu grupo de usuários atua como um provedor de serviços (SP) em nome da aplicação. O Amazon Cognito é compatível com a autenticação única (SSO) iniciada por SP e por IdP, conforme descrito nas seções 5.1.2 e 5.1.4 da Visão geral técnica do SAML V2.0.

Fornecer um certificado de assinatura de IdP válido

> O certificado de assinatura nos metadados do seu provedor SAML não deve estar vencido quando você for configurar o IdP do SAML em seu grupo de usuários.

Grupos de usuários aceitam vários certificados de assinatura

Quando o IdP SAML inclui mais de um certificado de assinatura nos metadados do SAML, no login, o grupo de usuários determina que a declaração do SAML é válida se corresponder a qualquer certificado nos metadados do SAML. Cada certificado de assinatura deve ter no máximo 4.096 caracteres.

Manter o parâmetro do estado de retransmissão

O Amazon Cognito e o IdP SAML mantêm as informações da sessão com um parâmetro `relayState`.

1. O Amazon Cognito é compatível com valores de `relayState` maiores do que 80 bytes. Embora as especificações do SAML afirmem que o valor de `relayState` "não deve exceder 80 bytes de comprimento", a prática atual do setor geralmente diverge desse comportamento. Como consequência, rejeitar valores de `relayState` maiores que 80 bytes quebrará muitas integrações padrão de provedor de SAML.
2. O token `relayState` é uma referência invisível às informações de estado mantidas pelo Amazon Cognito. O Amazon Cognito não garante o conteúdo do parâmetro `relayState`. Não analise o respectivo conteúdo de forma que sua aplicação dependa do resultado. Para obter mais informações, consulte a SAML 2.0 specification (Especificação do SAML 2.0).

Identificar o endpoint do ACS

Seu provedor de identidade SAML exige que você defina um endpoint de consumidor de declaração. Seu IdP redireciona seus usuários para esse endpoint com sua declaração SAML. Configure o endpoint a seguir no domínio do grupo de usuários para a vinculação POST SAML 2.0 no provedor de identidades SAML.

```
https://Your user pool domain/saml2/idpresponse
With an Amazon Cognito domain:
https://mydomain.auth.us-east-1.amazoncognito.com/saml2/idpresponse
With a custom domain:
https://auth.example.com/saml2/idpresponse
```

Consulte Como configurar um domínio de grupo de usuários para obter mais informações sobre domínios do grupo de usuários.

Sem declarações reproduzidas

Você não pode repetir nem reproduzir uma declaração de SAML em seu `saml2/idpresponse` endpoint do Amazon Cognito. Uma declaração de SAML reproduzida tem um ID que duplica o ID de uma resposta anterior do IdP.

O ID do grupo de usuários é o ID da entidade SP

É necessário fornecer o ID do grupo de usuários ao IdP no (SP) `urn` do provedor de serviços, também chamado de URI de público ou ID da entidade SP. O URI de público do grupo de usuários tem o formato a seguir.

```
urn:amazon:cognito:sp:us-east-1_EXAMPLE
```

Você pode encontrar o ID do grupo de usuários na Visão geral do grupo de usuários, no console do Amazon Cognito.

Mapear todos os atributos obrigatórios

Configure seu IdP SAML para que forneça valores para todos os atributos definidos como necessários em seu grupo de usuários. Por exemplo, `email` é um atributo obrigatório comum para grupos de usuários. Para que seus usuários possam fazer login, suas declarações do IdP SAML devem incluir uma declaração a ser mapeada para o atributo do grupo de usuários `email`. Para ter mais informações sobre mapeamento de atributos, consulte Mapeamento de atributos de IdP para perfis e tokens.

O formato de declaração tem requisitos específicos

O IdP do SAML deve incluir as seguintes reivindicações na declaração do SAML:

- Uma reivindicação de `NameID`. O Amazon Cognito associa uma declaração de SAML ao usuário de destino por `NameID`. Se houver alterações de `NameID`, o Amazon Cognito considerará que a declaração é para um novo usuário. O atributo definido em `NameID` na configuração do IdP deve ter um valor persistente. Para atribuir usuários do SAML a um perfil de usuário consistente no grupo de usuários, atribua sua declaração `NameID` a partir de um atributo com um valor que não mude.

```
<saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:persistent">
  carlos
</saml2:NameID>
```

O `Format` na declaração de `NameID` de `urn:oasis:names:tc:SAML:1.1:nameid-format:persistent` indica que seu IdP está transmitindo um valor imutável. O Amazon Cognito não exige essa declaração de formato. Ele atribui um formato de `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified` quando o IdP não especifica um formato da declaração `NameID`. Esse comportamento está em conformidade com a seção 2.2.2 Nome do tipo complexo IDType, da especificação SAML 2.0.

- Uma reivindicação `AudienceRestriction` com um valor de `Audience` que define o ID da entidade SP do grupo de usuários como o destino da resposta.

```
<saml:AudienceRestriction>
  <saml:Audience> urn:amazon:cognito:sp:us-east-1_EXAMPLE
</saml:AudienceRestriction>
```

- Para login único iniciado pelo SP, um elemento `Response` com valor `InResponseTo` do ID da solicitação SAML original.

```
<saml2p:Response Destination="https://mydomain.auth.us-east-1.amazoncognito.com/
saml2/idpresponse" ID="id123" InResponseTo="_dd0a3436-bc64-4679-
a0c2-cb4454f04184" IssueInstant="Date-time stamp" Version="2.0"
 xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol" xmlns:xs="http://
www.w3.org/2001/XMLSchema">
```

> **Note**
>
> As declarações de SAML iniciadas pelo IdP não devem conter um valor `InResponseTo`.

- Um elemento `SubjectConfirmationData` com um valor de `Recipient` do endpoint `saml2/idpresponse` do grupo de usuários e, para SAML iniciado pelo SP, um valor de `InResponseTo` que corresponde ao ID da solicitação SAML original.

```
<saml2:SubjectConfirmationData InResponseTo="_dd0a3436-bc64-4679-a0c2-
cb4454f04184" NotOnOrAfter="Date-time stamp" Recipient="https://mydomain.auth.us-
east-1.amazoncognito.com/saml2/idpresponse"/>
```

Solicitações de login iniciadas pelo SP

Quando o Autorizar endpoint direciona o usuário para a página de login do IdP, o Amazon Cognito inclui uma solicitação SAML em um parâmetro de URL da solicitação `HTTP GET`. A solicitação SAML contém informações sobre seu grupo de usuários, incluindo endpoint do ACS. Como opção, você pode aplicar uma assinatura criptográfica a essas solicitações.

Assinar solicitações e criptografar respostas

Cada grupo de usuários com um provedor de SAML gera um par de chaves assimétrico e um certificado de assinatura para uma assinatura digital que o Amazon Cognito atribui às solicitações de SAML. Cada IdP de SAML externo que você configura para aceitar uma resposta de SAML criptografada faz com que o Amazon Cognito gere um novo par de chaves e um certificado de criptografia para esse provedor. Para visualizar e baixar os certificados com a chave pública, escolha seu IdP no menu Provedores sociais e externos no console do Amazon Cognito.

Para estabelecer confiança com as solicitações de SAML do seu grupo de usuários, forneça ao IdP uma cópia do certificado de assinatura SAML 2.0 de seu grupo de usuários. Seu IdP pode ignorar as solicitações de SAML que seu grupo de usuários assinou se você não configurar o IdP para aceitar solicitações assinadas.

1. O Amazon Cognito aplica uma assinatura digital às solicitações de SAML que o usuário transmite ao IdP. Seu grupo de usuários assina todas as solicitações de logout único (SLO), e você pode configurar seu grupo de usuários para assinar solicitações de autenticação única (SSO) para qualquer IdP externo do SAML. Quando você fornece uma cópia do certificado, o IdP consegue verificar a integridade das solicitações SAML de seus usuários.
2. Seu IdP SAML pode criptografar respostas do SAML com o certificado de criptografia. Quando você configura um IdP com criptografia SAML, seu IdP só deve enviar respostas criptografadas.

Codificar caracteres não alfanuméricos

O Amazon Cognito não aceita caracteres UTF-8 de 4 bytes (como # ou #) que o IdP transmite como um valor de atributo. É possível codificar o caractere em Base64, transmiti-lo como texto e, então, decodificá-lo na aplicação.

No exemplo a seguir, a declaração de atributo não será aceita:

```
<saml2:Attribute Name="Name" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:unspecified">
```

```
  <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="xsd:string">#</saml2:AttributeValue>
</saml2:Attribute>
```

Ao contrário do exemplo anterior, a seguinte declaração de atributo será aceita:

```
<saml2:Attribute Name="Name" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
format:unspecified">
  <saml2:AttributeValue xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:type="xsd:string">8J+YkA==</saml2:AttributeValue>
</saml2:Attribute>
```

O endpoint de metadados deve ter uma segurança de camada de transporte válida

Se você ver, `InvalidParameterException` ao criar um IdP SAML com uma URL de endpoint de metadados HTTPS, por exemplo, “Erro ao recuperar metadados de”, certifique-se *<metadata endpoint>* de que o endpoint de metadados tenha o SSL configurado corretamente e que haja um certificado SSL válido associado a ele. Para obter mais informações sobre a validação de certificados, consulte O que é um SSL/TLS certificado? .

O endpoint de metadados deve estar em uma porta TCP padrão para HTTP ou HTTPS

O Amazon Cognito só aceita metadados URLs para provedores de SAML nas portas TCP padrão 80 para HTTP e 443 para HTTPS. Como prática recomendada de segurança, hospede os metadados SAML em um URL criptografado por TLS com o prefixo `https://`. Insira os metadados URLs no formato *`http://www.example.com/saml2/metadata.xml`* ou*`https://www.example.com/saml2/metadata.xml`*. O console do Amazon Cognito aceita metadados URLs somente com o prefixo. `https://` Você também pode configurar os metadados do IdP com e. CreateIdentityProviderUpdateIdentityProvider

Clientes de aplicações com SAML iniciado pelo IdP só podem fazer login com SAML

Ao ativar o suporte para um IdP do SAML 2.0 que oferece suporte ao login iniciado pelo IdP em um cliente de aplicativo, você só pode adicionar outro SAML IdPs 2.0 a esse cliente de aplicativo. Você não consegue adicionar o diretório de usuários ao grupo de usuários e todos os provedores de identidade externos que não sejam SAML a um cliente de aplicação configurado dessa forma.

As respostas de logout devem usar a vinculação POST

O endpoint `/saml2/logout` aceita `LogoutResponse` como solicitações `HTTP POST`. Os grupos de usuários não aceitam respostas de logout com vinculação `HTTP GET`.

Alternância de certificados de assinatura de metadados

O Amazon Cognito armazena em cache os metadados SAML por até 6 horas quando você fornece metadados com um URL. Ao realizar qualquer alternância de certificados de assinatura de metadados, configure sua fonte de metadados para publicar os certificados originais e novos por pelo menos 6 horas. Quando o Amazon Cognito atualiza o cache do URL de metadados, ele trata cada certificado como válido e o IdP SAML pode começar a assinar as declarações SAML com o novo certificado. Após esse período, você pode remover o certificado original dos metadados publicados.

Diferenciação entre maiúsculas e minúsculas dos nomes de usuário SAML

Quando um usuário federado tenta fazer login, o provedor de identidades (IdP) SAML passa um `NameId` para o Amazon Cognito na declaração SAML do usuário. O Amazon Cognito identifica um usuário federado SAML por meio da respectiva declaração `NameId`. Independentemente das configurações de diferenciação entre maiúsculas e minúsculas do grupo de usuários, o Amazon Cognito reconhece que um usuário federado voltou de um IdP SAML quando passa uma declaração `NameId` exclusiva e que diferencie maiúsculas e minúsculas. Se você mapear um atributo como `email` para `NameId` e seu usuário alterar o endereço de e-mail, ele não conseguirá fazer login na aplicação.

Mapeie `NameId` em suas declarações SAML de um atributo do IdP que tenha valores que não se alteram.

Por exemplo, Carlos tem um perfil de usuário em seu grupo de usuários que não diferencia maiúsculas e minúsculas de uma declaração SAML dos Serviços de Federação do Active Directory (ADFS) que passou um valor `NameId` de `Carlos@example.com`. Na próxima vez em que Carlos tentar fazer login, seu IdP ADFS passará um valor `NameId` de `carlos@example.com`. Como `NameId` deve apresentar uma correspondência exata de maiúsculas e minúsculas, o login não é bem-sucedido.

Se seus usuários não conseguirem fazer login depois que o respectivo `NameID` mudar, exclua o perfil desses usuários do grupo de usuários. O Amazon Cognito criará novos perfis de usuário na próxima vez em que eles fizerem login.

Tópicos

- Como configurar seu provedor de identidades SAML de terceiros
- Como adicionar e gerenciar provedores de identidade SAML em um grupo de usuários

- Iniciação de sessão SAML em grupos de usuários do Amazon Cognito
- Como desconectar usuários do SAML com logout único
- Assinatura e criptografia SAML
- Nomes e identificadores do provedor de identidades SAML

Como configurar seu provedor de identidades SAML de terceiros

Para adicionar um provedor de identidades (IdP) SAML ao seu grupo de usuários, você deve fazer algumas atualizações de configuração na interface de gerenciamento do IdP. Esta seção descreve como formatar os valores que você deve fornecer ao IdP. Você também aprenderá a recuperar o documento de metadados de URL estático ou ativo que identifica o IdP e suas declarações SAML para o grupo de usuários.

Para configurar soluções do provedor de identidades (IdP) SAML 2.0 de terceiros para funcionar com federação para grupos de usuários do Amazon Cognito, é necessário configurar o IdP SAML para redirecionar ao seguinte URL do Serviço do Consumidor de Declaração (ACS): `https://mydomain.auth.us-east-1.amazoncognito.com/saml2/idpresponse`. Se o grupo de usuários tiver um domínio do Amazon Cognito, você poderá encontrar o caminho do domínio do grupo de usuários no menu Domínio do grupo de usuários no console do Amazon Cognito.

Alguns SAML IdPs exigem que você forneça o `urn`, também chamado de URI do público ou ID da entidade SP, no formulário `urn:amazon:cognito:sp:us-east-1_EXAMPLE`. Você pode encontrar o ID do grupo de usuários na Visão geral do grupo de usuários, no console do Amazon Cognito.

Você também precisa configurar o IdP SAML para que forneça valores de todos os atributos obrigatórios em seu grupo de usuários. Normalmente, `email` é um atributo obrigatório para grupos de usuários. Nesse caso, o IdP SAML precisa fornecer alguma forma de reivindicação `email` em sua declaração SAML, e você precisa mapear a declaração para esse provedor.

As seguintes informações de configuração para soluções IdP SAML 2.0 de terceiros são um bom ponto de partida para configurar a federação com grupos de usuários do Amazon Cognito: Para obter as informações mais atuais, consulte diretamente a documentação do seu provedor.

Para assinar solicitações SAML, você deve configurar seu IdP para confiar nas solicitações assinadas pelo certificado de assinatura do grupo de usuários. Para aceitar respostas SAML criptografadas, configure seu IdP para criptografar todas as respostas SAML para seu grupo de

usuários. Seu provedor terá a documentação sobre a configuração desses recursos. Para ver um exemplo da Microsoft, consulte Configurar a criptografia de token SAML do Microsoft Entra.

> **Note**
>
> O Amazon Cognito exige apenas o documento de metadados do seu provedor de identidades. O provedor também pode oferecer informações de configuração personalizadas para a federação SAML 2.0 com o IAM ou o Centro de Identidade do AWS IAM. Para saber como configurar a integração com o Amazon Cognito, procure instruções gerais para recuperar o documento de metadados e gerenciar o restante da configuração em seu grupo de usuários.

Solução	Mais informações
Microsoft Entra ID	Metadados da federação
Okta	Como baixar os metadados do IdP e os certificados de assinatura SAML para uma integração de aplicações SAML
Auth0	Configurar Auth0 como provedor de identidades SAML
Identidade de ping (PingFederate)	Exportação de metadados SAML de PingFederate
JumpCloud	Notas de configuração do SAML
SecureAuth	Integração de aplicações SAML

Como adicionar e gerenciar provedores de identidade SAML em um grupo de usuários

Depois de configurar seu provedor de identidades para trabalhar com o Amazon Cognito, você pode adicioná-lo aos seus grupos de usuários e clientes de aplicações. Os procedimentos a seguir demonstram como criar, modificar e excluir provedores SAML em um grupo de usuários do Amazon Cognito.

Console de gerenciamento da AWS

Você pode usar o Console de gerenciamento da AWS para criar e excluir provedores de identidade SAML (IdPs).

Antes de criar um IdP SAML, é necessário ter o documento de metadados do SAML, que você obtém do IdP de terceiros. Para obter instruções sobre como obter ou gerar o documento de metadados do SAML necessário, consulte Como configurar seu provedor de identidades SAML de terceiros.

Para configurar um IdP SAML 2.0 em seu grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas credenciais da AWS .
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Provedores sociais e externos e selecione Adicionar um provedor de identidade.
5. Escolha um IdP SAML.
6. Insira um Nome de provedor. Você pode passar esse nome amigável em um parâmetro de solicitação `identity_provider` para Autorizar endpoint.
7. Insira Identifiers (Identificadores) separados por vírgulas. Um identificador diz ao Amazon Cognito que ele deve conferir o endereço de e-mail que um usuário insere quando faz o acesso e, em seguida, direcioná-lo ao provedor que corresponde ao domínio dele.
8. Escolha Add sign-out flow (Adicionar fluxo de desconexão) se quiser que o Amazon Cognito envie solicitações de desconexão assinadas ao seu provedor quando um usuário se desconectar. Você deve configurar o IdP SAML 2.0 para enviar respostas de logout ao endpoint `https://`*`mydomain.auth.us-east-1.amazoncognito.com`*`/saml2/logout` que é criado quando você configura o login gerenciado. O endpoint `saml2/logout` usa uma associação POST.

> **Note**
>
> Se você selecionar essa opção e seu IdP SAML esperar uma solicitação de logout assinada, você também precisará configurar o IdP SAML com um certificado de assinatura do seu grupo de usuários.

O IdP SAML processará a solicitação de logout assinada e desconectará seu usuário da sessão do Amazon Cognito.

9. Escolha sua configuração de login SAML iniciada pelo IdP. Como prática recomendada de segurança, escolha Aceitar somente declarações SAML iniciadas pelo SP. Se você preparou seu ambiente para aceitar com segurança sessões de login SAML não solicitadas, escolha Aceitar declarações de SAML iniciadas pelo SP e iniciadas pelo IdP. Para obter mais informações, consulte Iniciação de sessão SAML em grupos de usuários do Amazon Cognito.
10. Escolha uma Metadata document source (Fonte de documento de metadados). Se seu IdP oferecer metadados SAML em um URL público, você poderá escolher Metadata document URL (URL do documento de metadados) e inserir esse URL público. Do contrário, escolha Upload metadata document (Carregar documento de metadados) e, em seguida, um arquivo de metadados que você tenha baixado de seu provedor anteriormente.

   > **Note**
   >
   > Se seu provedor tiver um endpoint público, recomendamos que você insira um URL do documento de metadados em vez de carregar um arquivo. O Amazon Cognito atualiza os metadados automaticamente a partir do URL de metadados. Normalmente, a atualização de metadados ocorre a cada seis horas ou antes de os metadados expirarem, o que ocorrer primeiro.

11. Escolha Mapear atributos entre seu provedor SAML e sua aplicação para mapear atributos do provedor SAML para o perfil de usuário em seu grupo de usuários. Inclua os atributos obrigatórios do grupo de usuários no mapa de atributos.

    Por exemplo, quando escolher o User pool attribute (Atributo do grupo de usuários) `email`, insira o nome do atributo SAML como ele aparece na declaração SAML de seu IdP. Se o IdP oferecer exemplos de declarações SAML, é possível usá-los para ajudar você a encontrar o nome. Alguns IdPs usam nomes simples, como`email`, enquanto outros usam nomes como os seguintes.

    ```
    http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
    ```

12. Escolha Criar.

API/CLI

Use os comandos a seguir para criar e gerenciar um provedor de identidade (IdP) SAML.

Para criar um IdP e carregar um documento de metadados

- AWS CLI: `aws cognito-idp create-identity-provider`

  Exemplo com arquivo de metadados: `aws cognito-idp create-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-type SAML --provider-details file:///details.json --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

  Onde `details.json` contém:

  ```
  "ProviderDetails": {
        "MetadataFile": "<SAML metadata XML>",
        "IDPSignout" : "true",
        "RequestSigningAlgorithm" : "rsa-sha256",
        "EncryptedResponses" : "true",
        "IDPInit" : "true"
  }
  ```

  > **Note**
  >
  > Se *<SAML metadata XML>* contiver alguma instância do personagem", você deve adicionar \ como caractere de escape:\".

  Exemplo com URL de metadados: `aws cognito-idp create-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-type SAML --provider-details MetadataURL=https://myidp.example.com/sso/saml/metadata --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

- AWS API: CreateIdentityProvider

Para fazer upload de um novo documento de metadados para um IdP

- AWS CLI: `aws cognito-idp update-identity-provider`

  Exemplo com arquivo de metadados: `aws cognito-idp update-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-details file:///details.json --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

  Onde `details.json` contém:

```
"ProviderDetails": {
      "MetadataFile": "<SAML metadata XML>",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

  > **Note**
  >
  > Se *<SAML metadata XML>* contiver alguma instância do personagem", você deve adicionar \ como caractere de escape:\".

  Exemplo com URL de metadados: `aws cognito-idp update-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1 --provider-details MetadataURL=https://myidp.example.com/sso/saml/metadata --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

- AWS API: UpdateIdentityProvider

Para obter informações sobre um IdP específico

- AWS CLI: `aws cognito-idp describe-identity-provider`

```
aws cognito-idp describe-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1
```

- AWS API: DescribeIdentityProvider

Para listar informações sobre todos IdPs

- AWS CLI: `aws cognito-idp list-identity-providers`

  Exemplo: `aws cognito-idp list-identity-providers --user-pool-id us-east-1_EXAMPLE --max-results 3`

- AWS API: ListIdentityProviders

Para excluir um IdP

- AWS CLI: `aws cognito-idp delete-identity-provider`

  ```
  aws cognito-idp delete-identity-provider --user-pool-id us-east-1_EXAMPLE --provider-name=SAML_provider_1
  ```

- AWS API: DeleteIdentityProvider

Para configurar o IdP SAML para adicionar um grupo de usuários como uma parte dependente

- O URN do provedor de serviço dos grupos de usuários é `urn:amazon:cognito:sp:us-east-1_EXAMPLE`. O Amazon Cognito exige um valor de restrição de público que corresponda a esse URN na resposta do SAML. Configure seu IdP para usar o seguinte endpoint de vinculação POST para a IdP-to-SP mensagem de resposta.

  ```
  https://mydomain.auth.us-east-1.amazoncognito.com/saml2/idpresponse
  ```

- O IdP SAML deve preencher o `NameID` e quaisquer atributos necessários para o grupo de usuários na declaração do SAML. `NameID` é usado para identificar exclusivamente o usuário federado do SAML no grupo de usuários. O IdP deve transmitir o ID de nome do SAML de cada usuário em um formato consistente com distinção entre maiúsculas e minúsculas. Qualquer variação no valor do ID do nome de um usuário cria um novo perfil de usuário.

Como fornecer um certificado de assinatura ao IdP SAML 2.0

- Para baixar uma cópia da chave pública do Amazon Cognito que o IdP pode usar para validar solicitações de logout SAML, clique no menu Provedores sociais e externos do grupo de usuários, selecione seu IdP e, em Ver certificado de assinatura, selecione Baixar como .crt.

É possível excluir qualquer provedor SAML configurado em seu grupo de usuários com o console do Amazon Cognito.

Como excluir um provedor SAML

1. Faça login no console do Amazon Cognito.
2. No painel de navegação, escolha User Pools (Grupos de usuários) e escolha o grupo de usuários que deseja editar.
3. Clique no menu Provedores sociais e externos.
4. Selecione o botão de rádio ao lado do SAML IdPs que você deseja excluir.
5. Quando receber a solicitação de Delete identity provider (Excluir provedor de identidade), insira o nome do provedor SAML para confirmar a exclusão e escolha Delete (Excluir).

Iniciação de sessão SAML em grupos de usuários do Amazon Cognito

O Amazon Cognito é compatível com autenticação única (SSO) iniciada pelo provedor de serviços (iniciada pelo SP) e pelo IdP. Como prática recomendada de segurança, implemente o SSO iniciado pelo SP em seu grupo de usuários. A Seção 5.1.2 de Visão geral técnica do SAML V2.0 descreve a SSO iniciada pelo SP. O Amazon Cognito é o provedor de identidade (IdP) de sua aplicação. A aplicação é o provedor de serviços (SP) que recupera tokens para usuários autenticados. No entanto, quando você usa um IdP de terceiro para autenticar usuários, o Amazon Cognito é o SP. Quando os usuários do SAML 2.0 fazem a autenticação com um fluxo iniciado por SP, sempre devem fazer uma solicitação ao Amazon Cognito e redirecionar para o IdP para autenticação.

Para alguns casos de uso empresariais, o acesso a aplicações internas começa em um marcador em um painel hospedado pelo IdP empresarial. Quando um usuário seleciona um marcador, o IdP gera uma resposta SAML e a envia ao SP para autenticar o usuário na aplicação.

Você pode configurar um IdP SAML em seu grupo de usuários para oferecer suporte ao SSO iniciado pelo IdP. Com a autenticação iniciada por IdP, o Amazon Cognito não consegue verificar se ele solicitou a resposta SAML recebida, pois ele não inicia a autenticação com uma solicitação

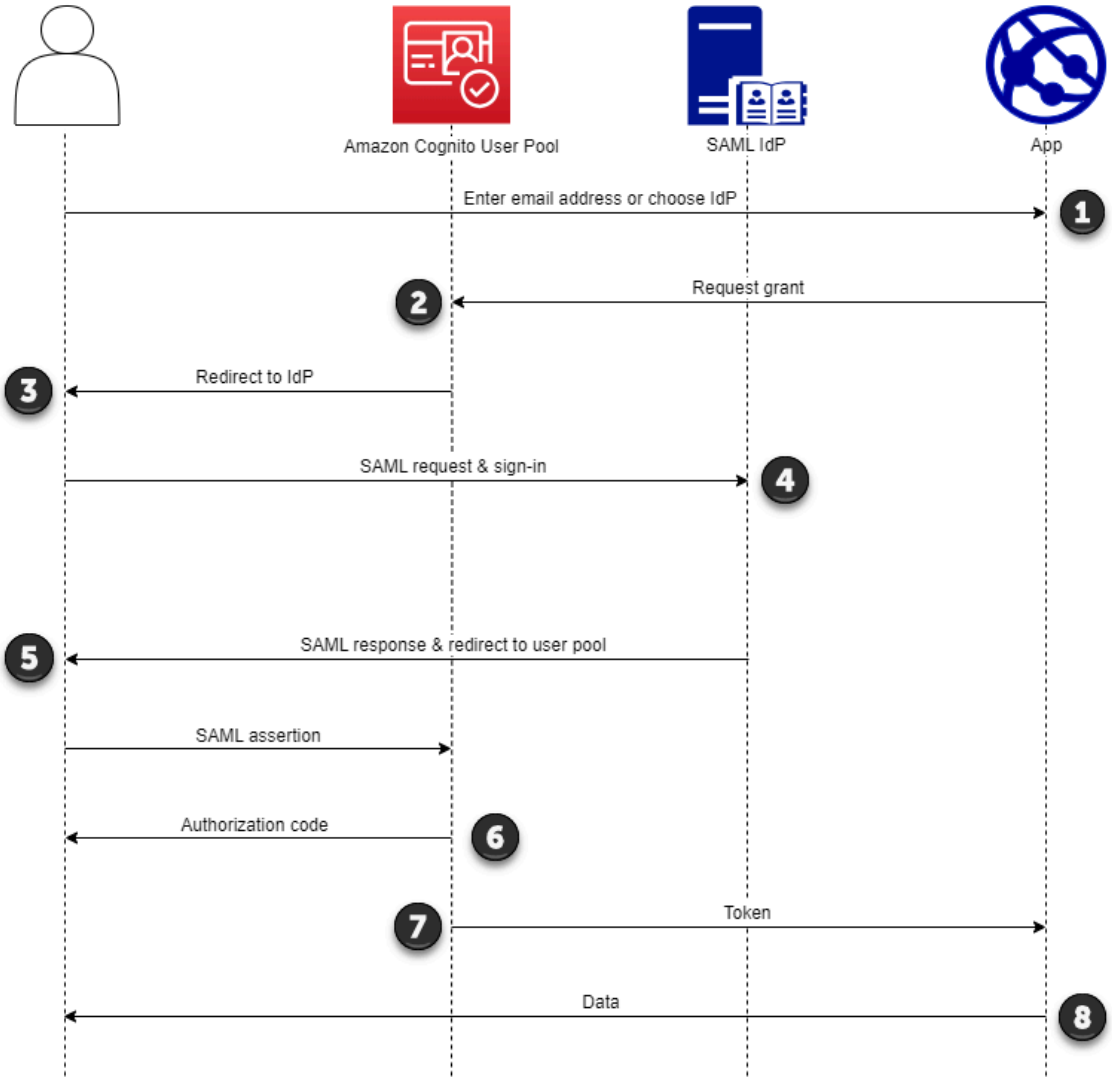
SAML. Com SSO iniciado pelo SP, o Amazon Cognito define parâmetros de estado que validam uma resposta SAML em relação à solicitação original. Com o login iniciado pelo SP, você também pode se proteger contra falsificação de solicitação entre sites (CSRF).

Tópicos

Implementar o login SAML iniciado pelo SP

Como prática recomendada, implemente o login service-provider-initiated (iniciado pelo SP) em seu grupo de usuários. O Amazon Cognito inicia a sessão do usuário e a redireciona para o seu IdP. Com esse método, você tem mais controle sobre quem apresenta as solicitações de login. Você também pode permitir o login iniciado pelo IdP em alguns casos.

O processo a seguir mostra como os usuários fazem o login iniciado pelo SP em seu grupo de usuários por meio de um provedor SAML.



1. Seu usuário insere o endereço de e-mail em uma página de login. Para determinar o redirecionamento do usuário para o IdP, você pode coletar o endereço de e-mail em uma aplicação personalizada ou invocar o login gerenciado na visualização da web.

   Você pode configurar suas páginas de login gerenciadas para exibir uma lista IdPs ou solicitar um endereço de e-mail e combiná-lo com o identificador do seu IdP SAML. Para solicitar um endereço de e-mail, edite o estilo de identidade visual do login gerenciado e, em Fundação, localize Comportamento de autenticação e, em Exibição do provedor, defina Estilo de exibição como Entrada de pesquisa de domínio.

2. Sua aplicação invoca o endpoint de redirecionamento do grupo de usuários e solicita uma sessão com o ID do cliente que corresponde à aplicação e o ID do IdP que corresponde ao usuário.

3. O Amazon Cognito redireciona seu usuário para o IdP com uma solicitação SAML, opcionalmente assinada, em um elemento `AuthnRequest`.
4. O IdP autentica o usuário de forma interativa ou com uma sessão memorizada por um cookie do navegador.
5. O IdP redireciona seu usuário para o endpoint de resposta SAML do grupo de usuários com a declaração SAML opcionalmente criptografada em sua carga útil POST.

> **Note**
>
> O Amazon Cognito cancela sessões que não recebem uma resposta em 5 minutos e redireciona o usuário para o login gerenciado. Quando seu usuário encontra esse resultado, ele recebe a mensagem de erro `Something went wrong`.

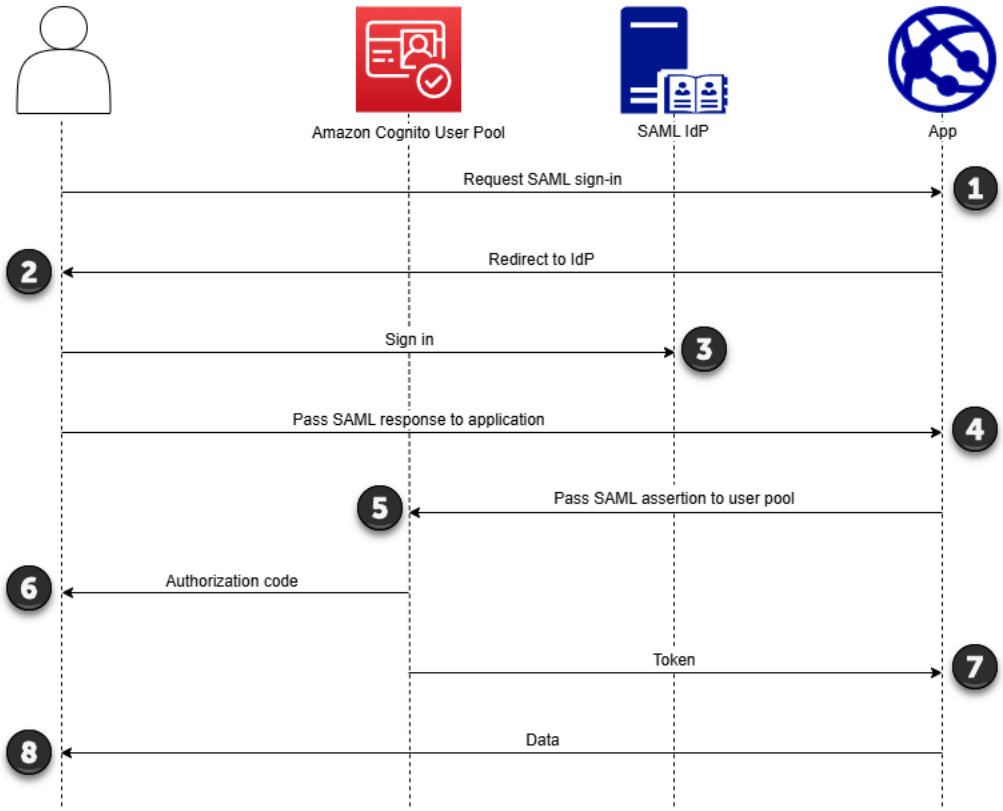
6. Após verificar a declaração do SAML e mapear atributos do usuário das declarações na resposta, o Amazon Cognito cria ou atualiza internamente o perfil do usuário no grupo de usuários. Normalmente, seu grupo de usuários retorna um código de autorização para a sessão do navegador do usuário.
7. Seu usuário apresenta o código de autorização ao seu aplicativo, que troca o código por tokens web JSON (JWTs).
8. A aplicação aceita e processa o token de ID do usuário como autenticação, gera solicitações autorizadas aos recursos com o token de acesso e armazena o token de atualização.

Quando um usuário se autentica e recebe uma concessão de código de autorização, o grupo de usuários retorna tokens de ID, acesso e atualização. O token de ID é um objeto de autenticação para gerenciamento de identidade baseado em OIDC. O token de acesso é um objeto de autorização com escopos OAuth 2.0. O token de atualização é um objeto que gera novos tokens de ID e acesso quando os tokens atuais do usuário expiram. Você pode configurar a duração dos tokens dos usuários em seu cliente de aplicação do grupo de usuários.

Você também pode escolher a duração dos tokens de atualização. Depois que o token de atualização do usuário expirar, ele deverá fazer login novamente. Se eles fizerem a autenticação por meio de um IdP SAML, a duração da sessão de seus usuários será definida pela expiração dos tokens, não pela expiração da sessão com o IdP. A aplicação precisa armazenar o token de atualização de cada usuário e renovar a sessão quando ela expirar. O login gerenciado mantém as sessões do usuário em um cookie do navegador válido por 1 hora.

Implementar o login SAML iniciado pelo IdP

Ao configurar seu provedor de identidades para fazer login no SAML 2.0 iniciado pelo IdP, você pode apresentar declarações de SAML ao endpoint `saml2/idpresponse` em seu domínio de grupo de usuários sem a necessidade de iniciar a sessão no Autorizar endpoint. Um grupo de usuários com essa configuração aceita declarações SAML iniciadas pelo IdP de um provedor de identidades externo do grupo de usuários compatível com o cliente de aplicação solicitado.



1. Um usuário solicita o login SAML em sua aplicação.
2. A aplicação invoca um navegador ou redireciona o usuário para a página de login do provedor SAML.
3. O IdP autentica o usuário de forma interativa ou com uma sessão memorizada por um cookie do navegador.
4. O IdP redireciona o usuário para sua aplicação com a declaração SAML, ou resposta, no corpo POST.
5. A aplicação adiciona a declaração SAML ao corpo POST de uma solicitação para o endpoint `saml2/idpresponse` do grupo de usuários.
6. O Amazon Cognito emite um código de autorização para o usuário.

7. Seu usuário apresenta o código de autorização ao seu aplicativo, que troca o código por tokens web JSON (JWTs).
8. A aplicação aceita e processa o token de ID do usuário como autenticação, gera solicitações autorizadas aos recursos com o token de acesso e armazena o token de atualização.

As etapas a seguir descrevem o processo geral de configuração e login com um provedor SAML 2.0 iniciado pelo IdP.

1. Crie ou atribua um grupo de usuários e um cliente de aplicação.
2. Crie um IdP SAML 2.0 em seu grupo de usuários.
3. Configure seu IdP para aceitar a iniciação do IdP. O SAML iniciado pelo IdP apresenta considerações de segurança às quais outros provedores de SSO não estão sujeitos. Por esse motivo, você não pode adicionar algo que não seja SAML IdPs, incluindo o próprio grupo de usuários, a nenhum cliente de aplicativo que use um provedor de SAML com login iniciado pelo IdP.
4. Associe seu provedor SAML iniciado pelo IdP a um cliente de aplicação em seu grupo de usuários.
5. Direcione seu usuário para a página de login do seu IdP SAML e recupere uma declaração de SAML.
6. Direcione o usuário ao endpoint `saml2/idpresponse` do grupo de usuários com sua declaração SAML.
7. Receba tokens web JSON (JWTs).

Para aceitar declarações de SAML não solicitadas em seu grupo de usuários, pense no impacto sobre a segurança da sua aplicação. É provável que ocorram tentativas de falsificação de solicitações e CSRF quando você aceita solicitações iniciadas pelo IdP. Embora seu grupo de usuários não possa verificar uma sessão de login iniciada pelo IdP, o Amazon Cognito valida seus parâmetros de solicitação e declarações de SAML.

Além disso, sua declaração de SAML não deve conter uma reivindicação `InResponseTo` e deve ter sido emitida nos últimos 6 minutos.

Você deve enviar solicitações com SAML iniciado pelo IdP para o `/saml2/idpresponse`. Para solicitações de autorização de login gerenciado e iniciadas pelo SP, forneça parâmetros que identifiquem o cliente da aplicação solicitado, os escopos, o URI de redirecionamento e outros

detalhes, como parâmetros da string de consulta nas solicitações `HTTP GET`. Entretanto, para declarações de SAML iniciadas pelo IdP, os detalhes da solicitação devem ser formatados como um parâmetro `RelayState` no corpo de uma solicitação `HTTP POST`. O corpo da solicitação também deve conter sua declaração de SAML como um parâmetro `SAMLResponse`.

Veja a seguir um exemplo de solicitação e resposta para um provedor SAML iniciado pelo IdP.

```
POST /saml2/idpresponse HTTP/1.1
User-Agent: USER_AGENT
Accept: */*
Host: example.auth.us-east-1.amazoncognito.com
Content-Type: application/x-www-form-urlencoded

SAMLResponse=[Base64-encoded SAML assertion]&RelayState=identity_provider
%3DMySAMLIdP%26client_id%3D1example23456789%26redirect_uri%3Dhttps%3A%2F
%2Fwww.example.com%26response_type%3Dcode%26scope%3Demail%2Bopenid%2Bphone

HTTP/1.1 302 Found
Date: Wed, 06 Dec 2023 00:15:29 GMT
Content-Length: 0
x-amz-cognito-request-id: 8aba6eb5-fb54-4bc6-9368-c3878434f0fb
Location: https://www.example.com?code=[Authorization code]
```

Console de gerenciamento da AWS

Para configurar um IdP para SAML iniciado pelo IdP

1. Crie um grupo de usuários, um cliente de aplicação e um provedor de identidades SAML.
2. Desassocie todos os provedores de identidade social e OIDC do seu cliente de aplicação, se houver algum associado.
3. Navegue até o menu Provedores sociais e externos do grupo de usuários.
4. Edite ou adicione um provedor SAML.
5. Em Login de SAML iniciado por IdP, escolha Aceitar declarações de SAML iniciadas pelo SP e iniciadas pelo IdP.
6. Escolha Salvar alterações.

API/CLI

Para configurar um IdP para SAML iniciado pelo IdP

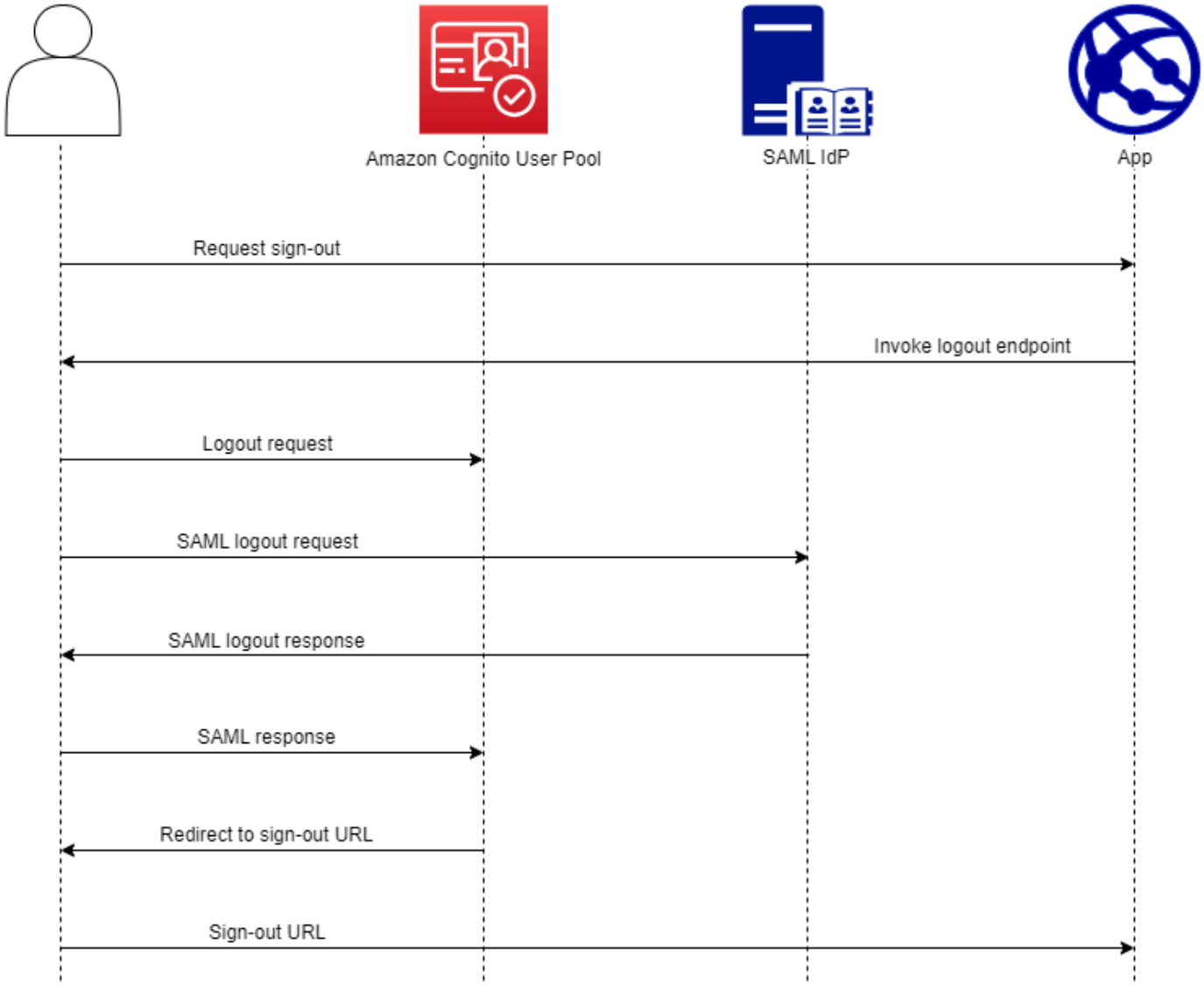
Configure o SAML iniciado pelo IdP com o `IDPInit` parâmetro em uma solicitação de API CreateIdentityProviderou UpdateIdentityProviderAPI. Veja a seguir um exemplo de `ProviderDetails` de um IdP que oferece suporte ao SAML iniciado pelo IdP.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

Como desconectar usuários do SAML com logout único

O Amazon Cognito oferece suporte ao logout único (SLO) do SAML 2.0. Com o SLO, seu aplicativo pode desconectar usuários de seus provedores de identidade SAML (IdPs) quando eles se desconectam do seu grupo de usuários. Dessa forma, quando os usuários quiserem entrar na aplicação novamente, precisam fazer a autenticação com o IdP SAML. Caso contrário, eles podem ter cookies do navegador do IdP ou do grupo de usuários que os transmitem à aplicação sem a necessidade de inserir credenciais.

Quando você configura seu IdP SAML para aceitar o Fluxo de logout, o Amazon Cognito redireciona seu usuário com uma solicitação de logout de SAML assinada para seu IdP. O Amazon Cognito determina o local de redirecionamento a partir do URL `SingleLogoutService` nos metadados do seu IdP. O Amazon Cognito assina a solicitação de desconexão com seu certificado de assinatura do grupo de usuários.



Quando você direciona um usuário com uma sessão de SAML para o endpoint `/logout` do grupo de usuários, o Amazon Cognito redireciona seu usuário do SAML com a seguinte solicitação para o endpoint de SLO especificado nos metadados do IdP.

```
https://[SingleLogoutService endpoint]?
SAMLRequest=[encoded SAML request]&
RelayState=[RelayState]&
SigAlg=http://www.w3.org/2001/04/xmldsig-more#rsa-sha256&
Signature=[User pool RSA signature]
```

Em seguida, seu usuário retorna ao seu endpoint `saml2/logout` com um `LogoutResponse` de seu IdP. Seu IdP deve enviar a `LogoutResponse` em uma solicitação `HTTP POST`. Em seguida, o Amazon Cognito faz o redirecionamento para o destino de redirecionamento a partir da solicitação inicial de desconexão.

Seu provedor de SAML pode enviar um `LogoutResponse` com mais de um `AuthnStatement`. O `sessionIndex` no primeiro `AuthnStatement` em uma resposta desse tipo deve corresponder ao `sessionIndex` na resposta SAML que autenticou originalmente o usuário. Se `sessionIndex` estiver em qualquer outro `AuthnStatement`, o Amazon Cognito não reconhecerá a sessão e seu usuário não será desconectado.

Console de gerenciamento da AWS

Para configurar a desconexão do SAML

1. Crie um grupo de usuários, um cliente de aplicação e um IdP SAML.
2. Ao criar ou editar seu provedor de identidades SAML, em Informações do provedor de identidades, marque a caixa com o título Adicionar fluxo de saída.
3. No menu Provedores sociais e externos do grupo de usuários, selecione seu IdP e localize o Certificado de assinatura.
4. Escolha Baixar como .crt.
5. Configure seu provedor SAML para oferecer suporte ao logout único e à assinatura de solicitações do SAML, além de carregar o certificado de assinatura do grupo de usuários. Seu IdP deve redirecionar para `/saml2/logout` no domínio do grupo de usuários.

API/CLI

Para configurar a desconexão do SAML

Configure o logout único com o `IDPSignout` parâmetro de uma solicitação de UpdateIdentityProviderAPI CreateIdentityProviderou. Veja a seguir um exemplo de `ProviderDetails` de um IdP compatível com o logout único do SAML.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",,
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
```

```
      "IDPInit" : "true"
}
```

Assinatura e criptografia SAML

O login do SAML 2.0 acontece com base no usuário de uma aplicação como portador de solicitações e respostas em seu fluxo de autenticação. Talvez você queira que os usuários não leiam nem modifiquem esses documentos SAML em trânsito. Para fazer isso, adicione assinatura e criptografia SAML aos provedores de identidade SAML (IdPs) em seu grupo de usuários. Com a assinatura SAML, seus grupos de usuários adicionam uma assinatura às solicitações de login e saída do SAML. Com a chave pública do grupo de usuários, seu IdP pode verificar se está recebendo solicitações SAML não modificadas. Então, quando seu IdP responde e transmite as declarações de SAML para as sessões do navegador dos usuários, o IdP pode criptografar essa resposta para que o usuário não inspecione seus próprios atributos e direitos.

Com a assinatura e criptografia do SAML, todas as operações criptográficas durante as operações do SAML do grupo de usuários devem gerar assinaturas e texto cifrado com as chaves geradas pelo Amazon Cognito. user-pool-provided Atualmente, você não pode configurar um grupo de usuários para assinar solicitações ou aceitar declarações criptografadas usando uma chave externa.

> **Note**
>
> Seus certificados de grupo de usuários são válidos por 10 anos. Uma vez por ano, o Amazon Cognito gera novos certificados de assinatura e criptografia para seu grupo de usuários. O Amazon Cognito retorna o certificado mais recente quando você solicita o certificado de assinatura e assina as solicitações com o certificado de assinatura mais recente. Seu IdP pode criptografar declarações SAML com qualquer certificado de criptografia de grupo de usuários que não tenha expirado. Seus certificados anteriores continuam válidos até o fim do prazo, e a chave pública não muda entre os certificados. Como prática recomendada, atualize anualmente o certificado na configuração do seu provedor.

Tópicos

- Aceitar respostas SAML criptografadas do seu IdP
- Assinatura de solicitações SAML

Aceitar respostas SAML criptografadas do seu IdP

O Amazon Cognito e seu IdP podem estabelecer confidencialidade nas respostas do SAML quando os usuários fazem login e logout. O Amazon Cognito atribui um par de chaves RSA público-privado e um certificado a cada provedor externo de SAML que você configura no seu grupo de usuários. Ao ativar a criptografia de resposta para seu provedor de SAML do grupo de usuários, carregue o certificado em um IdP que aceite respostas SAML criptografadas. A conexão do grupo de usuários com o IdP SAML só funciona quando o IdP começa a criptografar todas as declarações do SAML com a chave fornecida.

Veja a seguir uma visão geral do fluxo de login com SAML criptografado.

1. Seu usuário inicia o login e escolhe o IdP SAML.
2. Seu grupo de usuários Autorizar endpoint redireciona o usuário para o IdP SAML com uma solicitação de login do SAML. Seu grupo de usuários pode, como opção, colocar nessa solicitação uma assinatura que permita a verificação da integridade pelo IdP. Quando quiser assinar solicitações SAML, você deve configurar seu IdP para aceitar solicitações que seu grupo de usuários tenha assinado com a chave pública no certificado de assinatura.
3. O IdP SAML faz login com seu usuário e gera uma resposta SAML. O IdP criptografa a resposta com a chave pública e redireciona o usuário para o endpoint `/saml2/idpresponse` do grupo de usuários. O IdP deve criptografar a resposta conforme definido pela especificação SAML 2.0. Para obter mais informações, consulte `Element <EncryptedAssertion>` em Declarações e protocolos para o OASIS Security Assertion Markup Language (SAML) V2.0.
4. Seu grupo de usuários decifra o texto cifrado na resposta SAML com a chave privada e faz login com seu usuário.

> **Important**
>
> Quando você ativa a criptografia de resposta para um IdP SAML em seu grupo de usuários, o IdP deve criptografar todas as respostas com uma chave pública específica do provedor. O Amazon Cognito não aceita respostas SAML não criptografadas de um IdP externo SAML que você configura para aceitar a criptografia.

Qualquer IdP SAML externo no grupo de usuários pode aceitar criptografia de resposta, e cada IdP recebe seu próprio par de chaves.

Console de gerenciamento da AWS

Para configurar a criptografia de resposta SAML

1. Crie um grupo de usuários, um cliente de aplicação e um IdP SAML.
2. Ao criar ou editar seu provedor de identidades SAML, em Assinar solicitações e criptografar respostas, marque a caixa com o título Exigir declarações de SAML criptografadas desse provedor.
3. No menu Provedores sociais e externos do grupo de usuários, selecione o IdP SAML e clique em Exibir certificado de criptografia.
4. Escolha Baixar como .crt e envie o arquivo baixado ao seu IdP SAML. Configure o IdP SAML para criptografar as respostas do SAML com a chave no certificado.

API/CLI

Para configurar a criptografia de resposta SAML

Configure a criptografia de resposta com o `EncryptedResponses` parâmetro de uma solicitação CreateIdentityProviderou de UpdateIdentityProviderAPI. Veja a seguir um exemplo de `ProviderDetails` de um IdP compatível com a assinatura da solicitação.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

Para obter o certificado de criptografia do seu grupo de usuários, faça uma solicitação de DescribeIdentityProviderAPI e recupere o valor de `ActiveEncryptionCertificate` no parâmetro `ProviderDetails` de resposta. Salve esse certificado e envie-o ao seu IdP como certificado de criptografia para solicitações de login do seu grupo de usuários.

Assinatura de solicitações SAML

A possibilidade de provar a integridade das solicitações do SAML 2.0 ao seu IdP é uma vantagem de segurança do login do SAML iniciado pelo SP do Amazon Cognito. Cada grupo de usuários com um domínio recebe um certificado de assinatura X.509. Com a chave pública nesse certificado, os

grupos de usuários aplicam uma assinatura criptográfica às solicitações de saída que seu grupo de usuários gera quando os usuários selecionam um IdP SAML. Como opção, você pode configurar seu cliente de aplicação para assinar solicitações de login SAML. Quando você assina a solicitações SAML, seu IdP pode verificar se a assinatura nos metadados XML de suas solicitações corresponde à chave pública no certificado do grupo de usuários que você fornece.

Console de gerenciamento da AWS

Para configurar a assinatura da solicitação SAML

1. Crie um grupo de usuários, um cliente de aplicação e um IdP SAML.
2. Ao criar ou editar seu provedor de identidades SAML, em Assinar solicitações e criptografar respostas, marque a caixa com o título Assinar solicitações SAML neste provedor.
3. No menu Provedores sociais e externos do grupo de usuários, selecione Ver certificado de assinatura.
4. Escolha Baixar como .crt e envie o arquivo baixado ao seu IdP SAML. Configure seu IdP SAML para verificar a assinatura das solicitações SAML recebidas.

API/CLI

Para configurar a assinatura da solicitação SAML

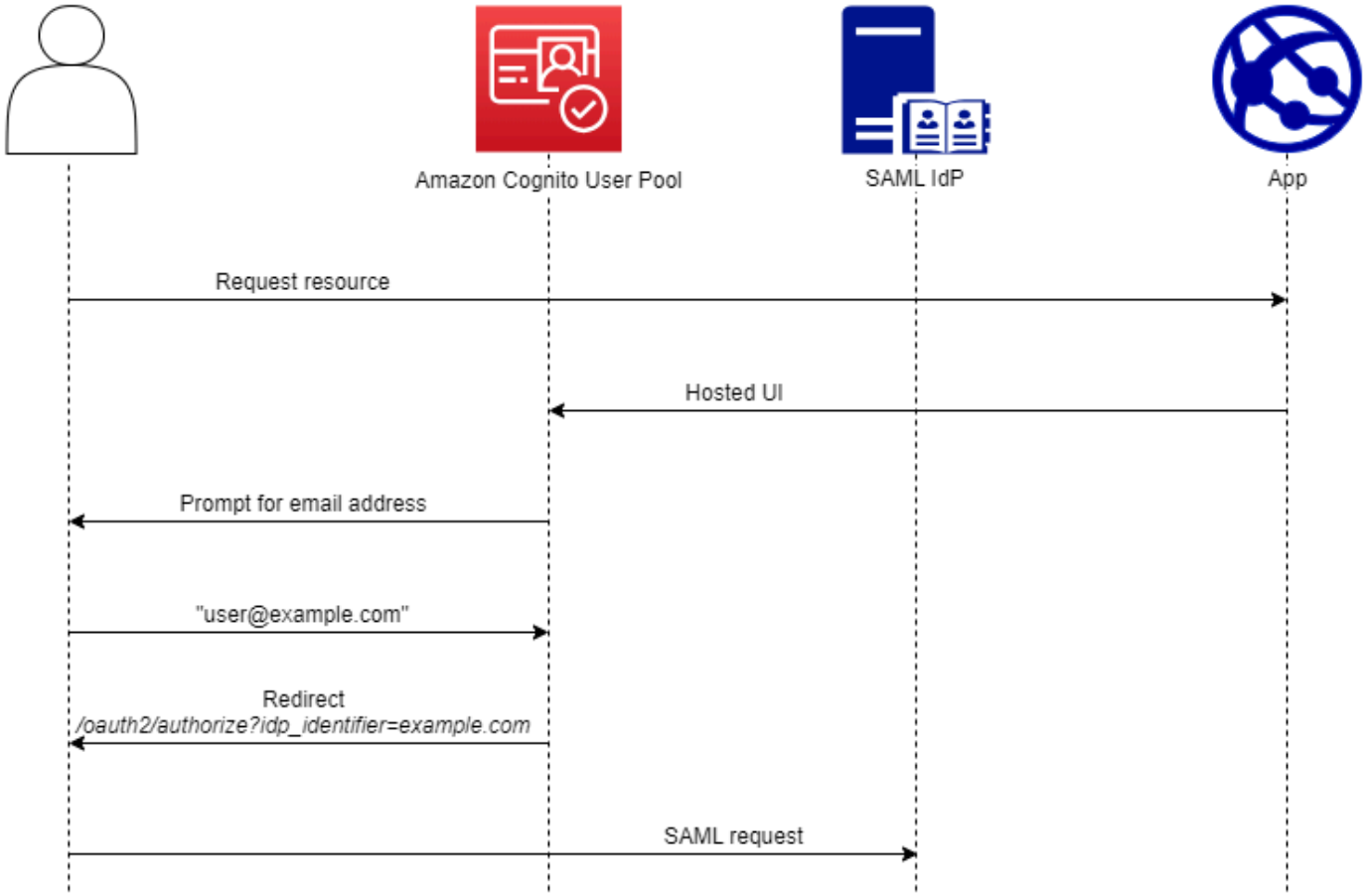
Configure a assinatura da solicitação com o `RequestSigningAlgorithm` parâmetro de uma solicitação CreateIdentityProviderou de UpdateIdentityProviderAPI. Veja a seguir um exemplo de `ProviderDetails` de um IdP compatível com a assinatura da solicitação.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

Nomes e identificadores do provedor de identidades SAML

Ao nomear seus provedores de identidade SAML (IdPs) e atribuir identificadores de IdP, você pode automatizar o fluxo de solicitações de entrada e saída iniciadas pelo SP para esse provedor. Para

obter informações sobre restrições de string ao nome do provedor, consulte a `ProviderName` propriedade de. CreateIdentityProvider



Você também pode escolher até 50 identificadores para os provedores SAML. Identificador é um nome amigável para um IdP em seu grupo de usuários e deve ser exclusivo dentro do grupo. Se os identificadores SAML corresponderem aos domínios de e-mail dos usuários, o login gerenciado solicitará o endereço de e-mail de cada usuário, avaliará o domínio em seu endereço de e-mail e os redirecionará para o IdP que corresponde ao domínio. Como a mesma organização pode ter vários domínios, um único IdP pode ter vários identificadores.

Independentemente de você usar identificadores de domínio de e-mail, é possível usar identificadores em uma aplicação multilocatário para redirecionar os usuários para o IdP correto. Quando quiser ignorar totalmente o login gerenciado, você pode personalizar os links que apresenta aos usuários para que eles sejam redirecionados por Autorizar endpoint diretamente para o IdP. Para cadastrar usuários com um identificador e redirecionar para o IdP, inclua o identificador no formato `idp_identifier=`*`myidp.example.com`* nos parâmetros de solicitação da solicitação de autorização inicial.

Outro método para passar um usuário para o seu IdP é preencher o parâmetro `identity_provider` com o nome do IdP no seguinte formato de URL.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
identity_provider=MySAMLIdP&
client_id=1example23456789&
redirect_uri=https://www.example.com
```

Depois que o usuário faz login com seu IdP SAML, ele o redireciona com uma resposta SAML no corpo `HTTP POST` para o endpoint `/saml2/idpresponse`. O Amazon Cognito processa a declaração do SAML e, se as declarações na resposta atenderem às expectativas, redireciona para a URL de retorno de chamada do cliente de aplicação. Depois que seu usuário tiver concluído a autenticação dessa forma, ele poderá interagir com páginas da Web somente para seu IdP e aplicação.

Com identificadores de IdP em formato de domínio, o login gerenciado solicita endereços de e-mail no login e, quando o domínio de e-mail corresponde a um identificador de IdP, redireciona os usuários para a página de login do IdP. Por exemplo, você cria uma aplicação que exige o login de funcionários de duas empresas diferentes. A primeira empresa, AnyCompany A, possui `exampleA.com` `exampleA.co.uk` e. A segunda empresa, AnyCompany B, possui`exampleB.com`. Neste exemplo, você configurou dois IdPs, um para cada empresa, da seguinte forma:

- Para o IdP A, você define os identificadores `exampleA.com` e `exampleA.co.uk`.
- Para o IdP B, você define o identificador `exampleB.com`.

Na aplicação, invoque o login gerenciado para seu cliente da aplicação para solicitar que cada usuário insira seu endereço de e-mail. O Amazon Cognito traz o domínio do endereço de e-mail, correlaciona o domínio a um IdP com um identificador de domínio e redireciona seu usuário para o IdP correto com uma solicitação para Autorizar endpoint que contém um parâmetro de solicitação `idp_identifier`. Por exemplo, se um usuário inserir `bob@exampleA.co.uk`, a próxima página com a qual ele vai interagir é a página de login do IdP em `https://auth.exampleA.co.uk/sso/saml`.

Também é possível implementar a mesma lógica de forma independente. Na aplicação, você pode criar um formulário personalizado que coleta as entradas do usuário e as correlaciona com o IdP correto de acordo com sua própria lógica. Você pode gerar portais personalizados para cada um dos

locatários da aplicação, em que cada um é vinculado ao endpoint de autorização com o identificador do locatário nos parâmetros da solicitação.

Para coletar um endereço de e-mail e analisar o domínio no login gerenciado, atribua pelo menos um identificador a cada IdP SAML que você atribuiu ao cliente da aplicação. Por padrão, a tela de login gerenciado exibe um botão para cada um dos IdPs que você atribuiu ao seu cliente de aplicativo. Mas se tiver atribuído os identificadores com êxito, a página de login da IU hospedada clássica será semelhante à imagem a seguir.
Uma página de login do login gerenciado no Amazon Cognito exibindo um login de usuário local e uma solicitação para que um usuário federado insira um endereço de e-mail.

> **Note**
>
> Na interface de usuário hospedada clássica, a página de login do seu cliente de aplicativo solicita automaticamente um endereço de e-mail quando você atribui identificadores ao seu. IdPs Na experiência de login gerenciado, você deve habilitar esse comportamento no editor de identidade visual. Na categoria de configurações Comportamento de autenticação, selecione Entrada de pesquisa de domínio sob o cabeçalho Exibição do provedor.

A análise de domínio no login gerenciado exige que você use domínios como identificadores de IdP. Se você atribuir um identificador de qualquer tipo a cada SAML IdPs de um cliente de aplicativo, o login gerenciado desse aplicativo não exibirá mais os botões de seleção de IDP. Adicione identificadores de IdP para SAML quando você quiser usar análise de e-mail ou lógica personalizada para gerar redirecionamentos. Quando você quiser gerar redirecionamentos silenciosos e também quiser que suas páginas de login gerenciadas exibam uma lista de IdPs, não atribua identificadores e use o parâmetro de `identity_provider` solicitação em suas solicitações de autorização.

- Se você atribuir somente um IdP SAML ao cliente da aplicação, a página de login do login gerenciado exibirá um botão para fazer login com esse IdP.
- Se você atribuir um identificador a cada IdP SAML ativado para o cliente da aplicação, na página de login do login gerenciado, será exibido um prompt de entrada para inserção de um endereço de e-mail.
- Se você tiver vários IdPs e não atribuir um identificador a todos eles, a página de login gerenciado exibirá um botão para entrar com cada IdP atribuído.
- Se você atribuiu identificadores à sua IdPs e deseja que suas páginas de login gerenciadas exibam uma seleção de botões de IdP, adicione um novo IdP que não tenha identificador no seu

cliente de aplicativo ou crie um novo cliente de aplicativo. Também é possível excluir um IdP existente e adicioná-lo novamente sem um identificador. Se você criar um IdP, seus usuários do SAML criarão novos perfis de usuário. Essa duplicação de usuários ativos pode afetar a cobrança no mês em que você altera a configuração do IdP.

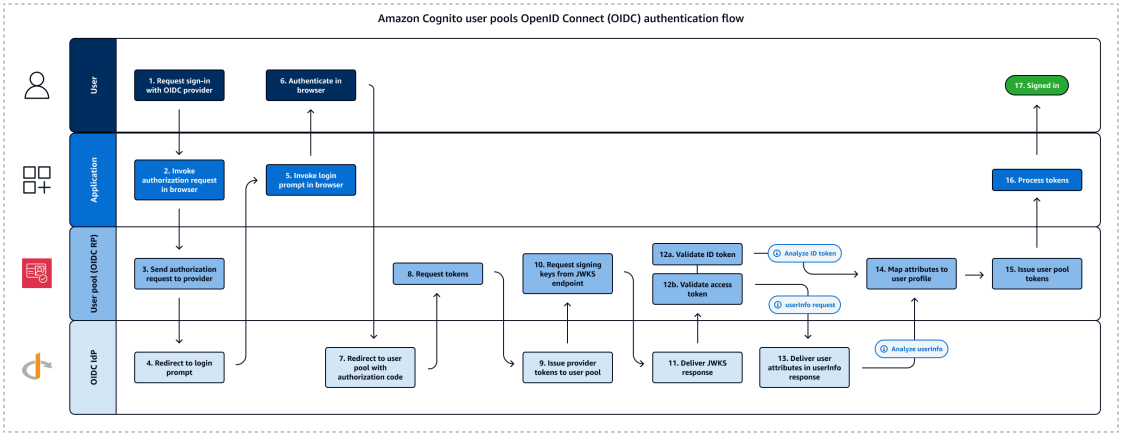
Para obter mais informações sobre a configuração do IdP, consulte Como configurar provedores de identidade para seu grupo de usuários.

Como usar provedores de identidade OIDC com um grupo de usuários

Os usuários podem entrar no seu aplicativo usando suas contas existentes dos provedores de identidade do OpenID Connect (OIDC) (). IdPs Com os provedores do OIDC, os usuários de sistemas independentes de login único podem fornecer as credenciais existentes enquanto a aplicação recebe tokens do OIDC no formato compartilhado dos grupos de usuários. Para configurar um IdP OIDC, configure seu IdP para gerenciar o grupo de usuários como a RP e configure sua aplicação para gerenciar o grupo de usuários como o IdP. O Amazon Cognito serve como uma etapa intermediária entre vários OIDC IdPs e seus aplicativos. O grupo de usuários aplica regras de mapeamento de atributos às declarações nos tokens de ID e acesso que o provedor transmite diretamente para o grupo de usuários. O Amazon Cognito então emite novos tokens com base nos atributos de usuário mapeados e em quaisquer ajustes adicionais que você tenha feito no fluxo de autenticação com acionadores do Lambda.

Os usuários que fazem login com um IdP do OIDC não precisam fornecer novas credenciais ou informações para acessar a aplicação de grupo de usuários. Sua aplicação pode redirecioná-los silenciosamente para seu IdP para login, com um grupo de usuários como uma ferramenta em segundo plano que padroniza o formato do token da aplicação. Para saber mais sobre o redirecionamento de IdP, consulte Autorizar endpoint.

Assim como com outros provedores de identidade de terceiros, você deve registrar a aplicação no provedor OIDC e obter informações sobre a aplicação IdP que deseja conectar ao seu grupo de usuários. Um IdP de grupo de usuários OIDC exige um ID do cliente, segredo do cliente, escopos que você deseja solicitar e informações sobre endpoints de serviços do provedor. Seu grupo de usuários pode descobrir os endpoints OIDC do provedor a partir de um endpoint de descoberta. Ou você pode inseri-los manualmente. Você também deve examinar os tokens de ID do provedor e criar mapeamentos de atributos entre o IdP e os atributos em seu grupo de usuários.



Consulte Fluxo de autenticação do IdP do grupo de usuários do OIDC para obter mais detalhes sobre esse fluxo de autenticação.

> **Note**
>
> O login por meio de um terceiro (federação) está disponível em grupos de usuários do Amazon Cognito. Esse recurso é independente da federação OIDC com bancos de identidades do Amazon Cognito.

Você pode adicionar um IdP OIDC ao seu grupo de usuários no Console de gerenciamento da AWS, por meio do ou com o método da AWS CLI API do grupo de usuários. CreateIdentityProvider

Tópicos

- Pré-requisitos
- Registrar uma aplicação com um IdP OIDC
- Adicionar um IdP OIDC ao seu grupo de usuários
- Testar a configuração de IdP OIDC
- Fluxo de autenticação do IdP do grupo de usuários do OIDC

Pré-requisitos

Antes de começar, você precisará fazer o seguinte:

- Um grupo de usuários com um cliente da aplicação e um domínio do grupo de usuários. Para obter mais informações, consulte Criar um grupo de usuários.

- Um IdP OIDC com a seguinte configuração:
  - Comporta a autenticação de cliente `client_secret_post`. O Amazon Cognito não verifica a declaração `token_endpoint_auth_methods_supported` no endpoint de descoberta OIDC para seu IdP. O Amazon Cognito não comporta a autenticação de cliente `client_secret_basic`. Para obter mais informações sobre a autenticação do cliente, consulte Autenticação de cliente na documentação do OpenID Connect.
  - Só usa HTTPS para endpoints OIDC, como `openid_configuration`, `userInfo` e `jwks_uri`.
  - Só usa as portas TCP 80 e 443 para endpoints OIDC.
  - Só assina tokens de ID com algoritmos HMAC-SHA, ECDSA ou RSA.
  - Publica uma reivindicação de ID de chave `kid` no `jwks_uri` e inclui uma reivindicação `kid` nos respectivos tokens.
  - Apresenta uma chave pública não expirada com uma cadeia de confiança de CA raiz válida.

Registrar uma aplicação com um IdP OIDC

Antes de adicionar um IdP OIDC à configuração do grupo de usuários e atribuí-lo aos clientes da aplicação, você configura uma aplicação cliente do OIDC no IdP. Seu grupo de usuários é a aplicação de parte confiável que gerenciará a autenticação com o IdP.

Para registrar com um IdP OIDC

1. Crie uma conta de desenvolvedor com o IdP OIDC.

   Links para o OIDC IdPs

IdP OIDC	Como instalar	URL de descoberta OIDC
Salesforce	Salesforce as an OpenID Connect Identity Provider	`https://MyDomainName.my.salesforce.com/.well-known/openid-configuration`
OneLogin	Connect an OIDC enabled app	`https://your-domain.onelogin.com/oidc/2/.well-known/openid-configuration`

IdP OIDC	Como instalar	URL de descoberta OIDC
JumpCloud	SSO with OIDC	`https://oauth.id.jumpcloud.com/.well-known/openid-configuration`
Okta	Instale um provedor de identidade Okta	`https://`*`Your Okta subdomain`*`.okta.com/.well-known/openid-configuration`
Microsoft Entra ID	OpenID Connect on the Microsoft identity platform	`https://login.microsoftonline.com/`*`{tenant}`*`/v2.0` Os valores de `tenant` podem incluir um ID de locatário, `common`, `organizations` ou `consumers`.

2. Inscreva o URL do domínio do grupo de usuários com o endpoint `/oauth2/idpresponse` com o IdP OIDC. Isso garante que o IdP OIDC o aceite posteriormente no Amazon Cognito quando autenticar os usuários.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/idpresponse
```

3. Selecione os escopos que você deseja que seu diretório de usuários compartilhe com o grupo de usuários. O escopo openid é necessário para que o OIDC ofereça qualquer IdPs informação do usuário. O escopo `email` é necessário para conceder acesso às declarações `email` e `email_verified`. Escopos adicionais na especificação do OIDC são `profile` para todos os atributos do usuário e phone para `phone_number` e `phone_number_verified`.
4. O IdP OIDC fornece um ID e uma chave secreta do cliente. Anote esses valores e adicione-os à configuração do IdP OIDC que você adicionará posteriormente ao grupo de usuários.

Exemplo: usar o Salesforce como um IdP OIDC com o grupo de usuários

Você usa um IdP OIDC quando deseja estabelecer confiança entre um IdP compatível com OIDC, como o Salesforce e seu grupo de usuários.

1. Crie uma conta no site de desenvolvedores do Salesforce.
2. Faça login na conta de desenvolvedor que você criou na etapa anterior.
3. Na página do Salesforce, execute um dos seguintes procedimentos:

   - Se você estiver usando o Lightning Experience, escolha o ícone de engrenagem da configuração e, depois, Setup Home (Página inicial de configuração).
   - Se você estiver usando o Salesforce Classic e você visualizar Setup (Configuração) no cabeçalho da interface do usuário, selecione-o.
   - Se você estiver usando o Salesforce Classic e você não visualizar Setup (Configuração) no cabeçalho, selecione seu nome na barra de navegação superior e selecione Setup (Configuração) na lista suspensa.
4. Na barra de navegação à esquerda, escolha Company Settings (Configurações da empresa).
5. Na barra de navegação, escolha Domain (Domínio), insira um domínio e escolha Create (Criar).
6. Na barra de navegação à esquerda, em Platform Tools (Ferramentas de plataforma), escolha Apps (Aplicações).
7. Escolha App Manager (Gerenciador de aplicativos).
8. a. Escolha New connected app (Nova aplicação conectada).

   b. Preencha os campos necessários.

      Em Start URL (URL de início), insira um URL no endpoint `/authorize` para o domínio do grupo de usuários que faz login em seu IdP Salesforce. Quando seus usuários acessam sua aplicação conectada, o Salesforce os direciona para esse URL para concluir o login. Em seguida, o Salesforce redireciona os usuários para o URL de retorno de chamada que você associou ao cliente de aplicação.

      ```
      https://mydomain.auth.us-east-1.amazoncognito.com/authorize?
      response_type=code&client_id=<your_client_id>&redirect_uri=https://
      www.example.com&identity_provider=CorpSalesforce
      ```

   c. Ative OAuth as configurações e insira a URL do `/oauth2/idpresponse` endpoint do seu domínio do grupo de usuários em URL de retorno de chamada. Esse é o URL em que o Salesforce emite o código de autorização que o Amazon Cognito troca por um token. OAuth

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/idpresponse
```

9. Selecione seus escopos. Você deve incluir o escopo openid. Para conceder acesso às declarações email e email_verified, adicione o escopo email. Escopos separados por espaços.
10. Escolha Criar.

    No Salesforce, o ID do cliente é chamado de Consumer Key (Chave do consumidor) e a chave secreta do cliente é uma Consumer Secret (Chave secreta do consumidor). Anote o ID e a chave secreta do cliente. Você poderá usá-los na próxima seção.

Adicionar um IdP OIDC ao seu grupo de usuários

Após configurar o IdP, você pode configurar o grupo de usuários para lidar com solicitações de autenticação com um IdP OIDC.

Amazon Cognito console

Adicionar um IdP OIDC no console

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas credenciais da AWS .
2. Escolha User Pools (Grupos de usuários) no menu de navegação.
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Provedores sociais e externos e selecione Adicionar um provedor de identidade.
5. Escolha um IdP OpenID Connect.
6. Insira um Nome do provedor exclusivo.
7. Insira o ID do cliente do IdP. Esse é o ID do cliente da aplicação que você criou no IdP OIDC. O ID do cliente fornecido deve ser um provedor OIDC que você configurou com um URL de retorno de chamada de `https://[your user pool domain]/oauth2/idpresponse`.
8. Insira o Segredo do cliente do IdP. Esse deve ser o segredo do cliente para o mesmo cliente da aplicação da etapa anterior.
9. Insira os Authorized scopes (Escopos autorizados) para esse provedor. Os escopos definem quais grupos de atributos do usuário (como `name` e `email`) sua aplicação solicitará ao seu provedor. Os escopos devem ser separados por espaços, seguindo a especificação OAuth2.0.

O IdP pode solicitar que os usuários consintam em fornecer esses atributos à aplicação quando fizerem login.

10. Escolha um método de solicitação de atributo. IdPspode exigir que as solicitações para seus `userInfo` endpoints sejam formatadas como ou. `GET` `POST` O endpoint `userInfo` do Amazon Cognito exige solicitações `HTTP GET`, por exemplo.
11. Selecione um Método de configuração para definir como o grupo de usuários determinará o caminho para os principais endpoints de federação OIDC no IdP. Normalmente, IdPs hospeda um `/well-known/openid-configuration` endpoint em um URL base do emissor. Se esse for o caso do seu provedor, a opção Preenchimento automático por meio do URL do emissor solicitará esse URL base, tentará acessar o caminho `/well-known/openid-configuration` com base nele e lerá os endpoints listados. Você pode ter caminhos de endpoint não típicos ou deseja encaminhar solicitações para um ou mais endpoints por meio de um proxy alternativo. Nesse caso, selecione Entrada manual e especifique caminhos para os endpoints `authorization`, `token`, `userInfo` e `jwks_uri`.

    > **Note**
    >
    > O URL deve começar com `https://` e não deve terminar com uma barra `/`. Somente os números de porta 443 e 80 podem ser usados com esse URL. Por exemplo, Salesforce usa este URL: `https://login.salesforce.com`
    > Se você escolher preenchimento automático, o documento de descoberta deverá usar HTTPS para os seguintes valores: `authorization_endpoint`, `token_endpoint`, `userinfo_endpoint` e `jwks_uri`. Caso contrário, o login falhará.

12. Configure as regras de mapeamento de atributos em Mapear atributos entre o provedor OpenID Connect e o grupo de usuários. Atributo do grupo de usuários é o atributo de destino no perfil de usuário do Amazon Cognito e o atributo OpenID Connect é o atributo de origem que você deseja que o Amazon Cognito encontre em uma declaração de token de ID ou resposta `userInfo`. O Amazon Cognito mapeia automaticamente a declaração OIDC sub para `username` no perfil do usuário de destino.

    Para obter mais informações, consulte Mapeamento de atributos de IdP para perfis e tokens.
13. Selecione Adicionar provedor de identidade.

14. No menu Clientes da aplicação, selecione um cliente da aplicação na lista. Navegue até a guia Páginas de login e, em Configuração gerenciada de páginas de login, clique em Editar. Localize provedores de identidades e adicione o novo IdP OIDC.
15. Escolha Salvar alterações.

API/CLI

Veja a configuração do OIDC no exemplo dois em. CreateIdentityProvider Você pode modificar essa sintaxe e usá-la como o corpo da `CreateIdentityProvider` solicitação ou `UpdateIdentityProvider` o arquivo `--cli-input-json` de entrada para create-identity-provider.

Testar a configuração de IdP OIDC

Na aplicação, você deve invocar um navegador no cliente do usuário para que ele possa fazer login com o provedor OIDC. Teste o login com seu provedor após concluir os procedimentos de configuração nas seções anteriores. O exemplo de URL a seguir carrega a página de login do grupo de usuários com um domínio de prefixo.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Esse link é a página para a qual o Amazon Cognito direcionará você ao acessar o menu Clientes da aplicação, selecionar um cliente da aplicação, navegar até a guia Páginas de login e selecionar Visualizar página de login. Para obter mais informações sobre domínios do grupo de usuários, consulte Como configurar um domínio de grupo de usuários. Para obter mais informações sobre clientes de aplicativos, incluindo cliente IDs e retorno de chamada URLs, consulteConfigurações específicas da aplicação com clientes de aplicação.

O link de exemplo a seguir configura o redirecionamento silencioso para o provedor `MyOIDCIdP` por meio do Autorizar endpoint com um parâmetro de consulta `identity_provider`. Esse URL ignora o login interativo do grupo de usuários com login gerenciado e leva diretamente à página de login do IdP.

```
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
identity_provider=MyOIDCIdP&response_type=code&client_id=1example23456789&redirect_uri=https://
www.example.com
```

Fluxo de autenticação do IdP do grupo de usuários do OIDC

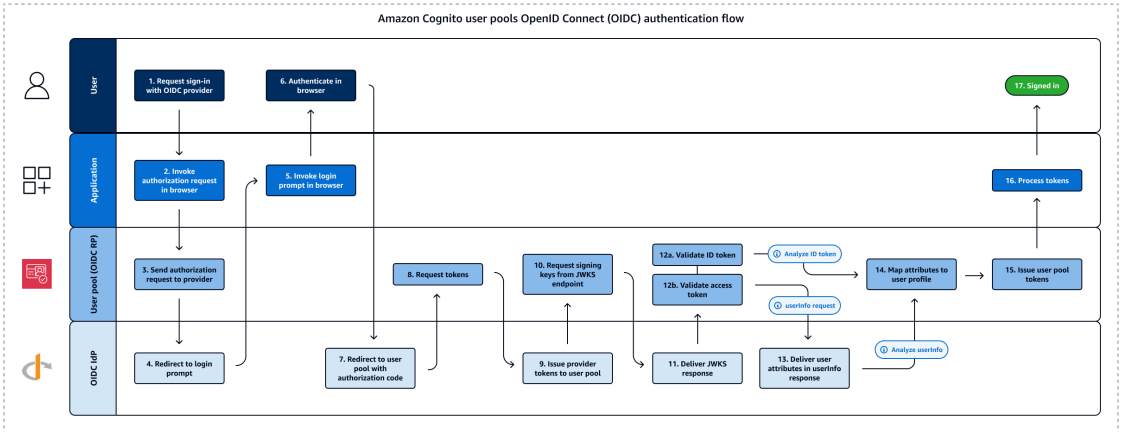
Com o login do OpenID Connect (OIDC), seu grupo de usuários automatiza um fluxo de login com código de autorização em seu provedor de identidades (IdP). Depois que o usuário conclui o login com o IdP, o Amazon Cognito coleta o código no endpoint `oauth2/idpresponse` do provedor externo. Com o token de acesso resultante, seu grupo de usuários consulta o endpoint `userInfo` do IdP para recuperar os atributos do usuário. Em seguida, seu grupo de usuários compara os atributos recebidos com as regras de mapeamento de atributos que você configurou e preenche o perfil do usuário e o token de ID adequadamente.

Os escopos OAuth 2.0 que você solicita na configuração do seu provedor OIDC definem os atributos do usuário que o IdP fornece ao Amazon Cognito. Como prática recomendada de segurança, solicite somente os escopos que correspondem aos atributos que você deseja mapear para seu grupo de usuários. Por exemplo, se seu grupo de usuários solicitar `openid profile`, você receberá todos os atributos possíveis, mas se solicitar `openid email phone_number`, você receberá apenas o endereço de e-mail e o número de telefone do usuário. Você pode configurar os escopos que você solicita do OIDC IdPs para serem diferentes daqueles que você autoriza e solicita na solicitação de autenticação do cliente do aplicativo e do grupo de usuários.

Quando o usuário faz login em sua aplicação usando um IdP OIDC, seu grupo de usuários conduz o seguinte fluxo de autenticação.

1. Um usuário acessa a página de login do login gerenciado e opta por fazer login com seu IdP OIDC.
2. Sua aplicação direciona o navegador do usuário para o endpoint de autorização do grupo de usuários.
3. O grupo de usuários redireciona a solicitação para o endpoint de autorização do IdP OIDC.
4. O IdP exibe uma solicitação de login.
5. Na aplicação, a sessão do usuário exibe uma solicitação de login para o IdP OIDC.
6. O usuário insere suas credenciais para o IdP ou apresenta um cookie para uma sessão já autenticada.
7. Depois que o usuário é autenticado, o IdP OIDC é redirecionado para o Amazon Cognito com um código de autorização.

8. O grupo de usuários troca o código de autorização por tokens de ID e acesso. O Amazon Cognito recebe tokens de acesso quando você configura o IdP com os escopos `openid`. As declarações no token de ID e na resposta `userInfo` são determinadas por escopos adicionais da configuração do IdP, por exemplo, `profile` e `email`.
9. O IdP emite os tokens solicitados.
10. Seu grupo de usuários determina o caminho do emissor para o `jwks_uri` endpoint do IdP na configuração do IdP e solicita URLs as chaves de assinatura do token do endpoint JSON web key set (JWKS).
11. O IdP retorna as chaves de assinatura do endpoint do JWKS.
12. O grupo de usuários valida os tokens do IdP com base nos dados de assinatura e expiração nos tokens.
13. O grupo de usuários autoriza uma solicitação para o endpoint `userInfo` do IdP com o token de acesso. O IdP responde com dados do usuário com base nos escopos do token de acesso.
14. O grupo de usuários compara o token de ID e a resposta `userInfo` do IdP com as regras de mapeamento de atributos no grupo de usuários. Ele grava atributos de IdP mapeados nos atributos do perfil do grupo de usuários.
15. O Amazon Cognito emite tokens do portador da aplicação, o que pode incluir tokens de identidade, acesso e atualização.
16. A aplicação processa os tokens do grupo de usuários e faz o login do usuário.



> **Note**
>
> O Amazon Cognito cancela solicitações de autenticação que não são concluídas em 5 minutos e redireciona o usuário para o login gerenciado. A página exibe a mensagem de erro `Something went wrong` (Ocorreu algum problema).

O OIDC é uma camada de identidade acima da OAuth 2.0, que especifica tokens de identidade formatados em JSON (JWT) que são emitidos pelos IdPs aplicativos clientes do OIDC (partes confiáveis). Consulte a documentação do IdP OIDC para obter informações sobre como adicionar o Amazon Cognito como uma parte dependente OIDC.

Quando um usuário se autentica com uma concessão de código de autorização, o grupo de usuários retorna tokens de ID, acesso e atualização. O token de ID é um token OIDC padrão para gerenciamento de identidade, e o token de acesso é um token OAuth 2.0 padrão. Para obter mais informações sobre os tipos de concessão que o cliente de aplicação do grupo de usuários pode comportar, consulte Autorizar endpoint.

Como um grupo de usuários processa declarações de um provedor de OIDC

Quando o usuário conclui o login com um provedor OIDC de terceiros, o login gerenciado recupera um código de autorização do IdP. O grupo de usuários troca o código de autorização por tokens de acesso e ID com o endpoint `token` do IdP. O grupo de usuários não transmite esses tokens ao usuário ou à aplicação, mas os utiliza para criar um perfil de usuário com dados que ele apresenta em declarações nos próprios tokens.

O Amazon Cognito não valida de forma independente o token de acesso. Em vez disso, ele solicita informações de atributos do usuário do endpoint `userInfo` do provedor e espera que a solicitação seja negada se o token não for válido.

O Amazon Cognito valida o token de ID do provedor com as seguintes verificações:

1. Confira se o provedor assinou o token com um algoritmo do seguinte conjunto: RSA, HMAC, Elliptic Curve.
2. Se o provedor assinou o token com um algoritmo de assinatura assimétrico, confira se o ID da chave de assinatura na declaração `kid` do token está listado no endpoint `jwks_uri` do provedor. O Amazon Cognito atualiza a chave de assinatura do endpoint do JWKS na configuração do IdP para cada token de ID de IdP que processa.

3. Compare a assinatura do token de ID com a assinatura que se espera com base nos metadados do provedor.
4. Compare a declaração `iss` com o emissor de OIDC configurado para o IdP.
5. Compare se a declaração `aud` corresponde ao ID do cliente configurado no IdP ou se ela contém o ID do cliente configurado se houver vários valores na declaração `aud`.
6. Confira se a data e a hora na declaração `exp` não é anterior à hora atual.

O grupo de usuários valida o token de ID e, depois, tenta fazer uma solicitação ao endpoint `userInfo` do provedor com o token de acesso do provedor. Ele recupera todas as informações do perfil do usuário que os escopos no token de acesso o autorizam a ler. Depois, o grupo de usuários procura os atributos do usuário definidos conforme necessário. É necessário criar mapeamentos para os atributos necessários na configuração do provedor. O grupo de usuários confere o token de ID do provedor e a resposta `userInfo`. O grupo de usuários grava todas as declarações que correlacionam regras de mapeamento e atributos do usuário no perfil do grupo de usuários. O grupo de usuários ignora atributos que, embora correspondam a uma regra de mapeamento, não são obrigatórios e não se encontram nas declarações do provedor.

Mapeamento de atributos de IdP para perfis e tokens

Os serviços de provedor de identidades (IdP), incluindo o Amazon Cognito, normalmente podem registrar mais informações sobre um usuário. Talvez você queira saber em qual empresa ele trabalha, como falar com ele e outras informações de identificação. Mas o formato que esses atributos assumem varia entre os provedores. Por exemplo, configure três IdPs de três fornecedores diferentes com seu grupo de usuários e examine um exemplo de declaração SAML, token de ID ou `userInfo` carga útil de cada um. Um deles representará o endereço de e-mail do usuário como `email`, outro como `emailaddress`, e o terceiro como `http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`.

Um grande benefício da consolidação IdPs com um grupo de usuários é a capacidade de mapear a variedade de nomes de atributos em um único esquema de token OIDC com nomes de atributos compartilhados, consistentes e previsíveis. Dessa forma, seus desenvolvedores não precisam manter a lógica para processar uma variedade complexa de eventos de autenticação única. Essa consolidação de formato é o mapeamento de atributos. O mapeamento de atributos do grupo de usuários adiciona nomes de atributos do IdP aos nomes de atributos do grupo de usuários correspondentes. Por exemplo, você pode configurar seu grupo de usuários para gravar o valor de uma reivindicação `emailaddress` no atributo padrão do grupo de usuários `email`.

Cada IdP do grupo de usuários tem um esquema de mapeamento de atributos separado. Para especificar mapeamentos de atributos para seu IdP, configure um provedor de identidades de grupo de usuários no console do Amazon Cognito, um SDK da AWS ou a API REST de grupos de usuários.

Coisas a saber sobre mapeamentos

Antes de começar a configurar o mapeamento de atributos de usuário, analise os seguintes detalhes importantes:

- Quando um usuário federado faz login em sua aplicação, deve haver um mapeamento para cada atributo que seu grupo de usuários exige. Por exemplo, se seu grupo de usuários exigir um atributo `email` para cadastro, mapeie esse atributo ao seu equivalente usando o IdP.
- Por padrão, os endereços de e-mail mapeados não são verificados. Não é possível verificar um endereço de e-mail mapeado usando um código único. Em vez disso, mapeie um atributo usando o IdP para obter o status de verificação. Por exemplo, o Google e a maioria dos provedores de OIDC incluem o atributo `email_verified`.
- É possível associar tokens do provedor de identidades (IdP) a atributos personalizados no grupo de usuários. Os provedores sociais apresentam um token de acesso e os provedores de OIDC apresentam um token de acesso e ID. Para associar um token, adicione um atributo personalizado com 2.048 caracteres, no máximo, conceda ao cliente da aplicação acesso de gravação ao atributo e associe `access_token` ou `id_token` do IdP ao atributo personalizado.
- Para cada atributo mapeado do grupo de usuários, a extensão máxima do valor de 2.048 caracteres deve ser suficiente para o valor que o Amazon Cognito obtém do IdP. Caso contrário, o Amazon Cognito vai relatar um erro quando os usuários acessarem sua aplicação. O Amazon Cognito não comporta mapeamento de tokens de IdP a atributos personalizados quando os tokens têm mais de 2.048 caracteres.
- O Amazon Cognito gera o atributo `username` no perfil de um usuário federado com base em reivindicações específicas transmitidas por seu IdP federado, conforme mostra a tabela a seguir. O Amazon Cognito anexa esse valor de atributo com o nome de seu IdP, por exemplo, `MyOIDCIdP_[sub]`. Quando você quiser que seus usuários federados tenham um atributo que corresponda exatamente a um atributo em seu diretório de usuários externo, mapeie esse atributo para um atributo de login do Amazon Cognito, como `preferred_username`.

Provedor de identidades	Atributo de origem **`username`**
Facebook	`id`
Google	`sub`
Login da Amazon	`user_id`
Fazer login com a Apple	`sub`
Provedores SAML	`NameID`
Provedores OpenID Connect (OIDC)	`sub`

- Quando um grupo de usuários não diferencia maiúsculas de minúsculas, o Amazon Cognito converte o atributo de origem do nome de usuário em minúsculas nos nomes de usuário gerados automaticamente pelos usuários federados. Veja a seguir um exemplo de nome de usuário para um grupo de usuários que diferencia maiúsculas de minúsculas: `MySAML_TestUser@example.com`. A seguir, veja o mesmo nome de usuário para um grupo de usuários que não diferencia maiúsculas de minúsculas: `MySAML_testuser@example.com`.

  Em grupos de usuários que não diferenciam maiúsculas de minúsculas, seus acionadores do Lambda que processam o nome de usuário devem considerar essa modificação em qualquer reivindicação de maiúsculas e minúsculas para atributos de origem de nome de usuário. Para vincular seu IdP a um grupo de usuários que tenha uma configuração de diferenciação de maiúsculas e minúsculas diferente do grupo de usuários atual, crie um grupo de usuários.
- O Amazon Cognito deve ser capaz de atualizar seus atributos mapeados do grupo de usuários quando os usuários fazem login na sua aplicação. Quando um usuário faz login por meio de um IdP, o Amazon Cognito atualiza os atributos mapeados com as informações mais recentes do IdP. O Amazon Cognito só atualiza os atributos mapeados quando seus valores mudam. Para garantir que o Amazon Cognito possa atualizar os atributos, verifique os seguintes requisitos:
  - Todos os atributos personalizados do grupo de usuários mapeados por meio do IdP devem ser mutáveis. É possível atualizar atributos personalizados mutáveis a qualquer momento. Entretanto, você só pode definir um valor para o atributo personalizado imutável ao criar o perfil de usuário pela primeira vez. Para criar um atributo personalizado mutável no console do Amazon Cognito, ative a caixa de seleção Mutável correspondente ao atributo adicionado ao selecionar Adicionar atributos personalizados no menu Cadastrar-se. Ou, se você criar

seu grupo de usuários usando a operação de CreateUserPoolAPI, poderá definir o `Mutable` parâmetro para cada um desses atributos como`true`. Se seu IdP enviar um valor para um atributo imutável mapeado, o Amazon Cognito retornará um erro, e o login falhará.

- Nas configurações do cliente de aplicativo para seu aplicativo, os atributos mapeados deve ser gravável. Você pode definir quais atributos são graváveis na página App clients (Clientes da aplicação) no console do Amazon Cognito. Ou, se você criar o aplicativo cliente usando a operação de API `CreateUserPoolClient`, você pode adicionar esses atributos à matriz `WriteAttributes`. Se o seu IdP enviar um valor para um atributo mapeado não gravável, o Amazon Cognito não definirá o valor do atributo e prosseguirá com a autenticação.

- Quando os atributos do IdP contêm vários valores, o Amazon Cognito transforma todos os valores em uma única string delimitada por vírgulas entre os caracteres de colchetes `[` e `]`. O Amazon Cognito codifica em forma de URL os valores que contêm caracteres não alfanuméricos, exceto para `.`, `-`, `*` e `_`. Você deve decodificar e analisar os valores individuais antes de usá-los em sua aplicação.
- O atributo de destino retém qualquer valor que as regras de mapeamento de atributos atribuam a ele, a menos que uma ação administrativa ou de login o altere. O Amazon Cognito não remove atributos dos usuários quando o atributo de origem não é mais enviado no token de provedor ou na declaração SAML. As ações a seguir removem o valor de um atributo de um perfil do grupo de usuários para um usuário federado:
  1. O IdP envia um valor em branco para o atributo de origem e uma regra de mapeamento aplica o valor em branco ao atributo de destino.
  2. Você limpa o valor do atributo mapeado com uma AdminDeleteUserAttributessolicitação DeleteUserAttributesor.

Como especificar mapeamentos de atributos do provedor de identidade para o grupo de usuários (Console de gerenciamento da AWS)

Você pode usar o Console de gerenciamento da AWS para especificar mapeamentos de atributos para o IdP, seu grupo de usuários.

> **Note**
>
> O Amazon Cognito mapeará solicitações de entrada para atributos do grupo de usuários somente se as solicitações existirem no token de entrada. Se uma reivindicação mapeada anteriormente não existir mais no token de entrada, ela não será excluída ou alterada. Se sua aplicação exigir o mapeamento de declarações excluídas, é possível usar o acionador do

Lambda de pré-autenticação para excluir o atributo personalizado durante a autenticação e permitir que esses atributos sejam preenchidos novamente com base no token de entrada.

Para especificar um mapeamento de atributo de IdP

1. Faça login no console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. No painel de navegação, escolha User Pools (Grupos de usuários) e escolha o grupo de usuários que deseja editar.
3. Clique no menu Provedores sociais e externos.
4. Clique em Adicionar um provedor de identidade ou selecione o IdP Facebook, Google, Amazon ou Apple que você configurou. Localize Attribute mapping (Mapeamento de atributos) e escolha Edit (Editar).

   Para obter mais informações sobre como adicionar um IdP social, consulte Como usar provedores de identidade social com um grupo de usuários.
5. Conclua as seguintes etapas para cada atributo que precisar mapear:

   a. Escolha um atributo da coluna User pool attribute (Atributo do grupo de usuários). Esse é o atributo que será atribuído ao perfil de usuário no grupo de usuários. Os atributos personalizados são listados depois dos atributos padrão.

   b. Selecione um atributo na coluna de ***<provider>***atributos. Esse será o atributo transmitido do diretório do provedor. Atributos conhecidos do provedor social são fornecidos em uma lista suspensa.

   c. Para mapear atributos adicionais entre seu IdP e o Amazon Cognito, escolha Add another attribute (Adicionar outro atributo).
6. Escolha Salvar alterações.

Para especificar um mapeamento de atributos do provedor SAML

1. Faça login no console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. No painel de navegação, escolha User Pools (Grupos de usuários) e escolha o grupo de usuários que deseja editar.
3. Clique no menu Provedores sociais e externos.
4. Escolha Add an identity provider (Adicionar um provedor de identidade) ou escolha o IdP que você configurou. Localize Attribute mapping (Mapeamento de atributos) e escolha Edit (Editar).

Para mais informações sobre como adicionar um IdP SAML, consulte Como usar provedores de identidade SAML com um grupo de usuários.

5. Conclua as seguintes etapas para cada atributo que precisar mapear:

   a. Escolha um atributo da coluna User pool attribute (Atributo do grupo de usuários). Esse é o atributo que será atribuído ao perfil de usuário no grupo de usuários. Os atributos personalizados são listados depois dos atributos padrão.

   b. Selecione um atributo da coluna SAML attribute (Atributo SAML). Esse será o atributo transmitido do diretório do provedor.

      Seu IdP pode oferecer exemplos de declarações SAML como referência. Alguns IdPs usam nomes simples, como`email`, enquanto outros usam nomes de atributos formatados em URL semelhantes a:

      ```
      http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress
      ```

   c. Para mapear atributos adicionais entre seu IdP e o Amazon Cognito, escolha Add another attribute (Adicionar outro atributo).

6. Escolha Salvar alterações.

Especificando mapeamentos de atributos do provedor de identidade para seu grupo de usuários (e API)AWS CLI AWS

O corpo da solicitação a seguir UpdateIdentityProvidermapeia CreateIdentityProviderou mapeia os atributos `emailaddress` “MyIdP” do provedor SAML e `phone` para os atributos do grupo de usuários e `email` `birthdate``phone_number`, nessa ordem. `birthdate` Esse é um corpo de solicitação completo para um provedor de SAML 2.0. Seu corpo de solicitação pode variar dependendo do tipo de IdP e dos detalhes específicos. O mapeamento de atributos está no parâmetro `AttributeMapping`.

```
{
   "AttributeMapping": {
      "email" : "emailaddress",
      "birthdate" : "birthdate",
      "phone_number" : "phone"
   },
   "IdpIdentifiers": [
      "IdP1",
```

```
      "pdxsaml"
   ],
   "ProviderDetails": {
      "IDPInit": "true",
      "IDPSignout": "true",
      "EncryptedResponses" : "true",
      "MetadataURL": "https://auth.example.com/sso/saml/metadata",
      "RequestSigningAlgorithm": "rsa-sha256"
   },
   "ProviderName": "MyIdP",
   "ProviderType": "SAML",
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Use os comandos a seguir para especificar os mapeamentos de atributos do IdP para o grupo de usuários.

Para especificar mapeamentos de atributos no momento da criação do provedor

- AWS CLI: `aws cognito-idp create-identity-provider`

  Exemplo com arquivo de metadados: `aws cognito-idp create-identity-provider --user-pool-id <user_pool_id> --provider-name=SAML_provider_1 --provider-type SAML --provider-details file:///details.json --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`

  Onde `details.json` contém:

  ```
  {
      "MetadataFile": "<SAML metadata XML>"
  }
  ```

  > **Note**
  >
  > Se *<SAML metadata XML>* contiver alguma citação ("), ela deverá ser escapada (\").

  Exemplo com URL de metadados:

  ```
  aws cognito-idp create-identity-provider \
  --user-pool-id us-east-1_EXAMPLE \
  ```

```
--provider-name=SAML_provider_1 \
--provider-type SAML \
--provider-details MetadataURL=https://myidp.example.com/saml/metadata \
--attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/
emailaddress
```

- API/SDK: CreateIdentityProvider

Para especificar mapeamentos de atributos para um IdP existente

- AWS CLI: `aws cognito-idp update-identity-provider`

  Example: `aws cognito-idp update-identity-provider --user-pool-id <user_pool_id> --provider-name <provider_name> --attribute-mapping email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/emailaddress`
- API/SDK: UpdateIdentityProvider

Para obter informações sobre o mapeamento de atributos para determinado IdP

- AWS CLI: `aws cognito-idp describe-identity-provider`

  Example: `aws cognito-idp describe-identity-provider --user-pool-id <user_pool_id> --provider-name <provider_name>`
- API/SDK: DescribeIdentityProvider

Vincular usuários federados a um perfil de usuário existente

Geralmente, o mesmo usuário tem um perfil com vários provedores de identidade (IdPs) que você conectou ao seu grupo de usuários. O Amazon Cognito pode vincular cada ocorrência de um usuário ao mesmo perfil em seu diretório. Dessa forma, uma pessoa com vários usuários de IdP pode ter uma experiência consistente em seu aplicativo. AdminLinkProviderForUserinstrui o Amazon Cognito a reconhecer o ID exclusivo de um usuário em seu diretório federado como um usuário no grupo de usuários. Um usuário em seu grupo de usuários é contabilizado como um usuário ativo mensal (MAU) para fins de faturamento quando você tem zero ou mais identidades federadas associadas ao perfil do usuário.

Quando um usuário federado faz login no grupo de usuários pela primeira vez, o Amazon Cognito procura um perfil local que você tenha vinculado à identidade dele. Se nenhum perfil

vinculado existir, o grupo de usuários cria um perfil. Você pode criar um perfil local e vinculá-lo ao usuário federado a qualquer momento antes do primeiro login, em uma solicitação de API `AdminLinkProviderForUser` ou em uma tarefa preliminar planejada ou em Acionador do Lambda de pré-cadastro. Depois que o usuário faz login e o Amazon Cognito detecta um perfil local vinculado, o grupo de usuários lê as reivindicações do usuário e as compara às regras de mapeamento do IdP. Depois, o grupo de usuários atualiza o perfil local vinculado com as reivindicações mapeadas pelo login. Dessa forma, você pode configurar o perfil local com declarações de acesso e manter suas declarações de identidade up-to-date com seu provedor. Depois que o Amazon Cognito associa o usuário federado a um perfil vinculado, ele sempre faz login nesse perfil. Depois, é possível vincular mais identidades de provedores do usuário ao mesmo perfil, oferecendo a um cliente uma experiência consistente na aplicação. Para vincular um usuário federado que já tenha feito login, você deve primeiro excluir o perfil existente. Você pode identificar perfis existentes por seu formato: *`[Provider name]`*`_identifier`. Por exemplo, .`LoginWithAmazon_amzn1.account.AFAEXAMPLE` Um usuário que você criou e depois vinculou a uma identidade de usuário de terceiros tem o nome de usuário com o qual ele foi criado e um atributo `identities` que contém os detalhes de suas identidades vinculadas.

> **Important**
>
> Como `AdminLinkProviderForUser` permite que um usuário com uma identidade federada externa faça login como um usuário existente no grupo de usuários, é fundamental que ela seja usada somente com atributos externos IdPs e de provedor nos quais o proprietário do aplicativo confie.

Por exemplo, se você for um provedor de serviços gerenciados (MSP) com uma aplicação compartilhada com vários clientes. Cada um dos clientes faz login em sua aplicação por meio dos Serviços de Federação do Active Directory (ADFS). Seu administrador de TI, Carlos, tem uma conta nos domínios de cada um de seus clientes. Você quer que Carlos seja reconhecido como administrador da aplicação toda vez em que fizer login, independentemente do IdP.

Seu ADFS IdPs apresenta o endereço de e-mail de Carlos `msp_carlos@example.com` na `email` reivindicação das declarações de SAML de Carlos para o Amazon Cognito. Você cria um usuário em seu grupo de usuários com o nome de usuário `Carlos`. Os comandos a seguir AWS Command Line Interface (AWS CLI) vinculam as identidades de Carlos de IdPs ADFS1, e. ADFS2 ADFS3

> **Note**
>
> É possível vincular um usuário com base em reivindicações de atributo específicas. Essa habilidade é exclusiva do OIDC e do SAML. IdPs Para outros tipos de provedor, é necessário realizar a vinculação com base em um atributo de origem fixo. Para obter mais informações, consulte AdminLinkProviderForUser. É necessário definir `ProviderAttributeName` como `Cognito_Subject` ao vincular um IdP social a um perfil de usuário. `ProviderAttributeValue` precisa ser o identificador exclusivo do usuário com seu IdP.

```
aws cognito-idp admin-link-provider-for-user \
--user-pool-id us-east-1_EXAMPLE \
--destination-user ProviderAttributeValue=Carlos,ProviderName=Cognito \
--source-user
 ProviderName=ADFS1,ProviderAttributeName=email,ProviderAttributeValue=msp_carlos@example.com

aws cognito-idp admin-link-provider-for-user \
--user-pool-id us-east-1_EXAMPLE \
--destination-user ProviderAttributeValue=Carlos,ProviderName=Cognito \
--source-user
 ProviderName=ADFS2,ProviderAttributeName=email,ProviderAttributeValue=msp_carlos@example.com

aws cognito-idp admin-link-provider-for-user \
--user-pool-id us-east-1_EXAMPLE \
--destination-user ProviderAttributeValue=Carlos,ProviderName=Cognito \
--source-user
 ProviderName=ADFS3,ProviderAttributeName=email,ProviderAttributeValue=msp_carlos@example.com
```

O perfil do usuário `Carlos` em seu grupo de usuários agora tem o atributo `identities` a seguir.

```
[{
    "userId": "msp_carlos@example.com",
    "providerName": "ADFS1",
    "providerType": "SAML",
    "issuer": "http://auth.example.com",
    "primary": false,
    "dateCreated": 111111111111111
}, {
    "userId": "msp_carlos@example.com",
    "providerName": "ADFS2",
```

```
    "providerType": "SAML",
    "issuer": "http://auth2.example.com",
    "primary": false,
    "dateCreated": 111111111111111
}, {
    "userId": "msp_carlos@example.com",
    "providerName": "ADFS3",
    "providerType": "SAML",
    "issuer": "http://auth3.example.com",
    "primary": false,
    "dateCreated": 111111111111111
}]
```

Fatos a saber sobre como vincular usuários federados

- Você pode vincular até cinco usuários federados a cada perfil de usuário.
- Você pode vincular usuários a cada IdP a partir de até cinco declarações de atributos do IdP, conforme definido pelo parâmetro `ProviderAttributeName` de `SourceUser` em uma solicitação de API `AdminLinkProviderForUser`. Por exemplo, se você vinculou pelo menos um usuário aos atributos de origem `email`, `phone`, `department`, `given_name` `elocation`, você só pode vincular usuários adicionais em um desses cinco atributos.
- É possível vincular usuários federados a um perfil de usuário federado existente ou a um usuário local.
- Você não pode vincular provedores a perfis de usuário no Console de gerenciamento da AWS.
- O token de ID do usuário contém todos os provedores associados na reivindicação `identities`.
- Você pode definir uma senha para o perfil de usuário federado criado automaticamente em uma solicitação de API. AdminSetUserPassword Depois, o status desse usuário é alterado de `EXTERNAL_PROVIDER` para `CONFIRMED`. Um usuário nesse estado pode fazer login como usuário federado e iniciar fluxos de autenticação na API como um usuário local vinculado. Eles também podem modificar suas senhas e atributos em solicitações de API autenticadas por token, como e. ChangePasswordUpdateUserAttributes Como prática de segurança recomendada e para manter os usuários sincronizados com seu IdP externo, não defina senhas em perfis de usuário federados. Em vez disso, vincule usuários a perfis locais com `AdminLinkProviderForUser`.
- O Amazon Cognito preenche os atributos do usuário em um perfil de usuário local vinculado quando o usuário faz login por meio de seu IdP. O Amazon Cognito processa declarações de identidade no token de ID de um IdP do OIDC e também verifica `userInfo` o endpoint dos provedores 2.0 e OIDC. OAuth O Amazon Cognito prioriza as informações em um token de ID em detrimento das informações de `userInfo`.

Ao descobrir que o usuário não está mais usando uma conta de usuário externa vinculada ao perfil dele, você pode desassociar essa conta de usuário do grupo de usuários. Ao vincular o usuário, você forneceu o nome do atributo, o valor do atributo e o nome do provedor do usuário na solicitação. Para remover um perfil que seu usuário não precisa mais, faça uma solicitação de AdminDisableProviderForUserAPI com parâmetros equivalentes.

Consulte AdminLinkProviderForUserpara obter mais exemplos e sintaxe de comando no AWS SDKs.

Login gerenciado do grupo de usuários

Você pode escolher um domínio da web para hospedar serviços para o grupo de usuários. Um grupo de usuários do Amazon Cognito ganha as funções a seguir quando você adiciona um domínio, coletivamente conhecido como login gerenciado.

- Um servidor de autorização que atua como um provedor de identidade (IdP) para aplicativos que funcionam com OAuth 2.0 e OpenID Connect (OIDC). O servidor de autorização roteia solicitações, emite e gerencia tokens web JSON (JWTs) e fornece informações de atributos do usuário.
- Uma interface de ready-to-use usuário (UI) para operações de autenticação, como login, saída e gerenciamento de senhas. As páginas de login gerenciado funcionam como um frontend da web para serviços de autenticação.
- Um provedor de serviços (SP) ou parte confiável (RP) para SAML 2.0, OIDC IdPs, Facebook, Login with Amazon IdPs, Sign in with Apple e Google.

Uma opção adicional que compartilha alguns recursos com o login gerenciado é a IU hospedada clássica. A IU hospedada clássica é uma versão de primeira geração dos serviços de login gerenciado. Os serviços de IdP e RP da IU hospedada geralmente têm as mesmas características do login gerenciado, mas as páginas de login têm um design mais simples e menos recursos. Por exemplo, o login por chave de acesso não está disponível na IU hospedada clássica. No plano de recursos Lite, a IU hospedada clássica é sua única opção para serviços de domínio do grupo de usuários.

As páginas de login gerenciado são uma coleção de interfaces da web para atividades básicas de cadastro, login, autenticação multifator e redefinição de senha no grupo de usuários. Eles também conectam usuários a um ou mais provedores de identidade terceirizados (IdPs) quando você deseja oferecer aos usuários a opção de login. Sua aplicação pode invocar as páginas de login gerenciado nos navegadores dos usuários quando você quiser autenticar e autorizar usuários.

Você pode personalizar a experiência do usuário do login gerenciado conforme a sua marca com logotipos, planos de fundo e estilos personalizados. Há duas opções de identidade visual que você pode aplicar à IU de login gerenciado: o editor de identidade visual para login gerenciado e a identidade visual da IU hospedada (clássica) para a IU hospedada.

Editor de identidade visual

Uma experiência de usuário atualizada com a maioria das opções de up-to-date autenticação e um editor visual no console do Amazon Cognito.

Identidade visual de IU hospedada

Uma experiência de usuário familiar para usuários anteriores dos grupos de usuários do Amazon Cognito. A identidade visual da IU hospedada é um sistema baseado em arquivos. Para aplicar a identidade visual às páginas de IU hospedada, você carrega um arquivo de imagem com o logotipo e um arquivo que define os valores de diversas opções de estilo CSS predefinidas.

O editor de identidade visual não está disponível em todos os planos de recursos para grupos de usuários. Para obter mais informações, consulte Planos de recursos de grupos de usuários.

Para obter mais informações sobre a criação de solicitações para login gerenciado e serviços de IU hospedada, consulte Referência de login gerenciado e endpoints do grupo de usuários.

> **Note**
>
> O login gerenciado do Amazon Cognito não é compatível com a autenticação personalizada com acionadores do Lambda de desafio de autenticação personalizada.

Tópicos

- Aplicar a identidade visual às páginas de login gerenciado

Managed login localization

O login gerenciado usa como padrão o idioma inglês nas páginas interativas com o usuário. Você pode exibir suas páginas de login gerenciado localizadas para o idioma de sua escolha. Os idiomas disponíveis são aqueles disponíveis no Console de gerenciamento da AWS. No link que você distribui aos usuários, adicione um parâmetro de consulta `lang`, conforme mostrado no exemplo a seguir.

```
https://<your domain>/oauth2/authorize?lang=es&response_type=code&client_id=<your app
 client id>&redirect_uri=<your relying-party url>
```

O Amazon Cognito define um cookie no navegador dos usuários com sua preferência de idioma após a solicitação inicial com um parâmetro `lang`. Depois que o cookie é definido, a seleção do idioma persiste sem exibir ou exigir que você inclua o parâmetro nas solicitações. Por exemplo, depois que um usuário faz uma solicitação de login com um parâmetro `lang=de`, as páginas de login gerenciado são exibidas em alemão até que ele limpe os cookies ou faça uma nova solicitação com um novo parâmetro de localização, como `lang=en`.

A localização está disponível somente para login gerenciado. Você precisa estar no plano de recursos Essentials ou Plus e ter atribuído seu domínio para usar a identidade visual de login gerenciado.

A seleção que seu usuário faz no login gerenciado não está disponível para acionadores personalizados de remetente de e-mail ou SMS. Ao implementar esses acionadores, você deve usar outros mecanismos para determinar o idioma preferencial do usuário. Nos fluxos de login, o atributo `locale` pode indicar o idioma preferencial do usuário com base na localização. Nos fluxos de cadastro, a região ou o ID do cliente da aplicação do grupo de usuários pode indicar uma preferência de idioma.

Os idiomas a seguir estão disponíveis.

Idiomas do login gerenciado

Linguagem	Código
Alemã	de

Linguagem	Código
Inglês	en
Espanhola	es
Francesa	fr
Bahasa Indonésia	id
Italiano	it
Japonês	ja
Coreano	ko
Português (Brasil)	pt-BR
Chinês (simplificado)	zh-CN
Chinês (tradicional)	zh-TW

Documentos de termos

Você pode configurar suas páginas de login gerenciado para exibir links para seus documentos de Termos de uso e Política de privacidade quando os usuários se cadastrarem. Ao configurar ambos documentos de termos no cliente da aplicação, os usuários verão o seguinte texto durante o cadastro: Ao se cadastrar, você concorda com nossos Termos de uso e Política de privacidade. As frases Termos de uso e Política de privacidade aparecem na página de login gerenciado, com hiperlinks para os documentos.

Os documentos de termos oferecem suporte a idiomas específicos URLs que se alinham à localização gerenciada de login. Quando os usuários selecionam um idioma com o parâmetro de consulta `lang`, o Amazon Cognito exibe links para seus documentos de termos nesse idioma. Se você não configurou um URL para um idioma específico, o Amazon Cognito usará o URL padrão configurado para o cliente da aplicação.

Para configurar documentos de termos para o cliente da aplicação, navegue até o menu Login gerenciado no grupo de usuários. Em Documentos de termos, selecione Criar documento de termos.

Amazon Cognito console

Como criar um documento de termos

1. Navegue até o grupo de usuários e clique no menu Login gerenciado. Localize Documentos de termos.
2. Selecione Criar documento de termos.
3. Selecione o cliente da aplicação ao qual deseja atribuir o documento de termos.
4. Insira um Nome dos termos. Isso identificará o documento no console.
5. Em Links, escolha um Idioma e insira o URL onde você hospeda seu documento de termos nesse idioma.
6. URLs Para adicionar outros idiomas, escolha Adicionar outro.
7. Escolha Criar.

Amazon Cognito user pools API

Veja a seguir um exemplo de corpo da solicitação CreateTerms. Ele faz com que a página de cadastro do cliente da aplicação `1example23456789` exiba links para uma versão em francês e uma versão em português (Brasil) da política de privacidade quando o login gerenciado estiver localizado para esse idioma. É necessário definir uma solicitação separada para que URLs o `terms-of-use` login gerenciado renderize os links na página de inscrição.

```
{
   "ClientId": "1example23456789",
   "Enforcement": "NONE",
   "Links": {
      "cognito:default" : "https://example.com/privacy/",
      "cognito:french" : "https://example.com/fr/privacy/",
      "cognito:portuguese-brazil" : "https://example.com/pt/privacy/"
   },
   "TermsName": "privacy-policy",
   "TermsSource": "LINK",
   "UserPoolId": "us-east-1_EXAMPLE"
}
```

> **Note**
>
> Você deve criar um documento de termos de uso e de política de privacidade para o cliente da aplicação antes que o Amazon Cognito exiba os documentos de termos nas páginas de login gerenciado.

Configurando o login gerenciado com AWS Amplify

Se você usa AWS Amplify para adicionar autenticação ao seu aplicativo web ou móvel, você pode configurar suas páginas de login gerenciadas na interface de linha de comando (CLI) do Amplify e bibliotecas na estrutura do Amplify. Para adicionar autenticação à sua aplicação, adicione a categoria `Auth` ao seu projeto. Em seguida, em sua aplicação, autentique os usuários do grupo de usuários com as bibliotecas de cliente do Amplify.

Você pode invocar páginas de login gerenciado para autenticação ou federar usuários por meio de um endpoint de autorização que redireciona para um IdP. Após um usuário se autenticar com êxito com o provedor, o Amplify criará um novo usuário no grupo de usuários e transmitirá os tokens do usuário para a aplicação.

Os exemplos a seguir mostram como usar AWS Amplify para configurar o login gerenciado com provedores sociais em seu aplicativo.

- React
- Swift
- Vibração
- Android

Configurar o login gerenciado com o console do Amazon Cognito

O primeiro requisito para login gerenciado e IU hospedada é um domínio do grupo de usuários. No console de grupos de usuários, navegue até a guia Domínio do grupo de usuários e adicione um domínio do Cognito ou um domínio personalizado. Você também pode escolher um domínio durante o processo de criação de um novo grupo de usuários. Para obter mais informações, consulte Como configurar um domínio de grupo de usuários. Quando um domínio está ativo no grupo de usuários, todos os clientes da aplicação veiculam páginas públicas de autenticação nesse domínio.

Ao criar ou modificar um domínio do grupo de usuários, você define a Versão de marca do seu domínio. Essa versão de marca é uma opção de login gerenciado ou IU hospedada (clássica). A versão de marca escolhida se aplica a todos os clientes da aplicação que usam os serviços de login em seu domínio.

A próxima etapa é criar um cliente da aplicação na guia Clientes da aplicação do grupo de usuários. No processo de criação de um cliente da aplicação, o Amazon Cognito solicitará informações sobre sua aplicação e, em seguida, solicitará que você selecione um URL de retorno. O URL de retorno também é chamado de URL de parte confiável (RP), o URI de redirecionamento e o URL de retorno de chamada. Esse é o URL no qual sua aplicação é executada, por exemplo, `https://www.example.com` ou `myapp://example`.

Após configurar um domínio e um cliente da aplicação com um estilo de identidade visual no grupo de usuários, suas páginas de login gerenciado ficarão disponíveis na Internet.

Visualizar a página de login

No console do Amazon Cognito, clique no botão Visualizar páginas de login na guia Páginas de login do cliente da aplicação, no menu Clientes da aplicação. Esse botão levará você a uma página de login no domínio do grupo de usuários com os parâmetros básicos a seguir.

- O ID do cliente da aplicação
- Uma solicitação de concessão de código de autorização
- Uma solicitação para todos os escopos que você ativou para o cliente da aplicação atual
- O primeiro URL de retorno de chamada na lista para o cliente da aplicação atual

O botão Visualizar página de login é útil quando você deseja testar as funções básicas das páginas de login gerenciado. Suas páginas de login corresponderão à Versão de marca que você atribuiu ao domínio do grupo de usuários. Você pode personalizar o URL de login com parâmetros adicionais e modificados. Na maioria dos casos, os parâmetros gerados automaticamente do link Visualizar página de login não atendem totalmente às necessidades da aplicação. Nesses casos, você precisa personalizar o URL que a aplicação invoca quando faz login dos usuários. Para obter mais informações sobre chaves e valores de parâmetros de login, consulte Referência de login gerenciado e endpoints do grupo de usuários.

A página da web de login usa o formato de URL a seguir. Este exemplo solicita uma concessão de código de autorização com o parâmetro `response_type=code`.

```
https://<your domain>/oauth2/authorize?response_type=code&client_id=<your app client
 id>&redirect_uri=<your relying-party url>
```

É possível pesquisar a string do domínio do grupo de usuários no menu Domínio do grupo de usuários. No menu Clientes do aplicativo, você pode identificar o cliente do aplicativo IDs, seu retorno de chamada URLs, seus escopos permitidos e outras configurações.

Ao navegar até o endpoint `/oauth2/authorize` com parâmetros personalizados, o Amazon Cognito redireciona você ao endpoint `/oauth2/login` ou, se tiver um parâmetro `identity_provider` ou `idp_identifier`, ele redireciona você silenciosamente para a página de login de seu IdP.

Exemplo de solicitação para uma concessão implícita

Você pode visualizar a página da web de login com o URL a seguir para a concessão de código implícita onde `response_type=token`. Depois de um login bem-sucedido, o Amazon Cognito retorna tokens do grupo de usuários para a barra de endereço do seu navegador da Web.

```
            https://mydomain.auth.us-east-1.amazoncognito.com/authorize?
response_type=token&client_id=1example23456789&redirect_uri=https://
mydomain.example.com
```

Os tokens de identidade e acesso aparecem como parâmetros anexados ao URL de redirecionamento.

O URL a seguir é um exemplo de resposta de uma solicitação de concessão implícita.

```
            https://auth.example.com/
#id_token=eyJraaBcDeF1234567890&access_token=eyJraGhIjKlM1112131415&expires_in=3600&token_type=
```

Personalizar páginas de autenticação

No passado, o Amazon Cognito hospedava somente páginas de login com a IU hospedada clássica, um design simples que proporciona uma aparência universal às páginas da web de autenticação. Era possível personalizar grupos de usuários do Amazon Cognito com uma imagem de logotipo e

ajustar alguns estilos com um arquivo que especificava valores de estilo CSS. Posteriormente, o Amazon Cognito introduziu o login gerenciado, um serviço de autenticação hospedado atualizado. O login gerenciado é atualizado look-and-feel com o editor de marca. O editor de identidade visual é um editor visual no-code e oferece um conjunto maior de opções do que a experiência de personalização da IU hospedada. O login gerenciado também introduziu imagens de fundo personalizadas e um tema de modo escuro.

É possível alternar entre as experiências de identidade visual do login gerenciado e da IU hospedada nos grupos de usuários. Para saber mais sobre como personalizar suas páginas de login gerenciado, consulte Aplicar a identidade visual às páginas de login gerenciado.

Informações importantes sobre o login gerenciado e a IU hospedada

O cookie de sessão de login gerenciado e de IU hospedada com duração de 1 hora

Quando um usuário faz login usando suas páginas de login ou um provedor de terceiros, o Amazon Cognito define um cookie no navegador dele. Com esse cookie, os usuários podem fazer login novamente com o mesmo método de autenticação por 1 hora. Ao fazer login com o cookie do navegador, eles recebem novos tokens que duram o período especificado na configuração do cliente da aplicação. Alterações nos atributos do usuário ou nos fatores de autenticação não afetam sua capacidade de fazer login novamente com o cookie do navegador.

A autenticação com o cookie de sessão não redefine a duração do cookie para mais 1 hora. Os usuários precisarão fazer login novamente se tentarem acessar as páginas de login mais de 1 hora após a última autenticação interativa bem-sucedida.

Confirmar contas de usuário e verificar atributos de usuário

Para usuários locais do grupo de usuários, o login gerenciado e a IU hospedada funcionam melhor quando você configura o grupo de usuários para Permitir que o Cognito envie mensagens automaticamente para verificar e confirmar. Quando você ativa essa configuração, o Amazon Cognito envia uma mensagem com um código de confirmação para os usuários que se cadastram. Quando você confirma os usuários como administrador do grupo de usuários, as páginas de login exibem uma mensagem de erro após o cadastro. Nesse estado, o Amazon Cognito criou o usuário, mas não conseguiu enviar uma mensagem de verificação. Você ainda pode confirmar os usuários como administradores, mas eles podem entrar em contato com a central de suporte após encontrarem um erro. Para receber mais informações sobre confirmação administrativa, consulte Permitir que os usuários se inscrevam na aplicação, mas mediante confirmação deles como administradores do grupo de usuários.

Escopo de operações de login gerenciado

O login gerenciado e a IU hospedada clássica são compatíveis com o cadastro, o login e o gerenciamento de senhas. Isso inclui concluir o login com autenticação multifator (MFA) e registrar autenticadores WebAuthn. O login gerenciado não é compatível com o gerenciamento de perfil de usuário por meio de autoatendimento, como alterações de atributos e configuração de preferências de MFA. Você deve implementar o gerenciamento de perfil no código da sua própria aplicação. O login gerenciado também não oferece a capacidade de confirmar alterações de atributos quando você atualiza endereços de e-mail e números de telefone como administrador com a operação da AdminUpdateUserAttributesAPI.

Visualizar as alterações na configuração

Se você fizer alterações de estilo em suas páginas e elas não aparecerem imediatamente, aguarde alguns minutos e atualize a página.

Decodificar tokens do grupo de usuários

Os tokens do grupo de usuários do Amazon Cognito são assinados usando um RS256 algoritmo. Você pode decodificar e verificar os tokens do grupo de usuários usando AWS Lambda. Consulte Decodificar e verificar os tokens JWT do Amazon Cognito em. GitHub

Versão do TLS

As páginas de login gerenciado e IU hospedada exigem criptografia em trânsito. Os domínios do grupo de usuários fornecidos pelo Amazon Cognito exigem que os navegadores dos usuários negociem uma versão mínima do TLS 1.2. Os domínios personalizados são compatíveis com conexões de navegador com TLS versão 1.2. A IU hospedada (clássica) não exige o TLS 1.2 para domínios personalizados, mas o login gerenciado mais recente exige o TLS versão 1.2 tanto para domínios personalizados quanto para domínios de prefixo. Como o Amazon Cognito gerencia a configuração dos serviços de domínio, você não pode modificar os requisitos de TLS do domínio do grupo de usuários.

Políticas de CORS

Nem o login gerenciado, nem a IU hospedada são compatíveis com as políticas de origem de compartilhamento de recursos de origem cruzada (CORS). Uma política de CORS impediria os usuários de transmitir parâmetros de autenticação em suas solicitações. Em vez disso, implemente uma política de CORS no frontend da aplicação. O Amazon Cognito retorna um cabeçalho de resposta `Access-Control-Allow-Origin: *` para as solicitações aos endpoints a seguir.

1. Endpoint de token
2. Revogar endpoint
3. endpoint userinfo

Cookies

O login gerenciado e a IU hospedada definem cookies nos navegadores dos usuários. Os cookies seguem os requisitos de alguns navegadores de que os sites não definam cookies de terceiros. Eles têm como escopo apenas os endpoints do seu grupo de usuários e incluem o seguinte:

- Um cookie `XSRF-TOKEN` para cada solicitação.
- Um cookie `csrf-state` para consistência da sessão quando um usuário é redirecionado.
- Um cookie `csrf-state-legacy` para consistência da sessão, lido pelo Amazon Cognito como uma alternativa quando seu navegador não é compatível com o atributo `SameSite`.
- Um cookie de sessão `cognito` que preserva as tentativas de login bem-sucedidas por uma hora.
- Um cookie `lang` que preserva a escolha de localização do idioma do usuário no login gerenciado.
- Um cookie `page-data` que mantém a persistência dos dados obrigatórios enquanto o usuário navega entre as páginas de login gerenciado.

No iOS, você pode bloquear todos os cookies. Essa configuração não é compatível com o login gerenciado ou com a IU hospedada. Para trabalhar com usuários que possam ativar essa configuração, crie a autenticação do grupo de usuários em um aplicativo iOS nativo com um AWS SDK. Nesse cenário, você pode criar seu próprio armazenamento de sessão que não seja baseado em cookies.

Efeitos da alteração da versão do login gerenciado

Considere os efeitos a seguir da adição de domínios e da configuração da versão de login gerenciado.

- Ao adicionar um domínio de prefixo, seja com identidade visual de login gerenciado ou de IU hospedada (clássica), pode levar até 60 segundos até que as páginas de login estejam disponíveis.
- Ao adicionar um domínio personalizado, seja com identidade visual de login gerenciado ou de IU hospedada (clássica), pode levar até 5 minutos até que as páginas de login estejam disponíveis.

- Ao alterar a versão de marca do domínio, pode levar até 4 minutos até que as páginas de login estejam disponíveis na nova versão de marca.
- Ao alternar entre a identidade visual de login gerenciado e de IU hospedada (clássica), o Amazon Cognito não mantém as sessões de usuário. É necessário fazer login novamente com a nova interface.

Estilo padrão

Quando você cria um cliente de aplicativo no Console de gerenciamento da AWS, o Amazon Cognito atribui automaticamente um estilo de marca ao seu cliente de aplicativo. Quando você cria programaticamente um cliente de aplicativo com a CreateUserPoolClientoperação, nenhum estilo de marca é criado. O login gerenciado não está disponível para um cliente de aplicativo criado com um AWS SDK até que você crie um com uma CreateManagedLoginBrandingsolicitação.

Tempo limite do prompt de autenticação de login gerenciado

O Amazon Cognito cancela solicitações de autenticação que não são concluídas em 5 minutos e redireciona o usuário para o login gerenciado. A página exibe a mensagem de erro `Something went wrong` (Ocorreu algum problema).

Como configurar um domínio de grupo de usuários

Configurar um domínio é opcional na configuração de um grupo de usuários. Um domínio de grupo de usuários hospeda recursos para autenticação de usuários, federação com provedores terceirizados e fluxos do OpenID Connect (OIDC). Ele tem o login gerenciado, uma interface pré-criada para operações importantes, como cadastro, login e recuperação de senha. Ele também hospeda os endpoints padrão do OpenID Connect (OIDC), como authorize, userInfo e token, para autorização machine-to-machine (M2M) e outros fluxos de autenticação e autorização do OIDC e 2.0. OAuth

Os usuários são autenticados com páginas de login gerenciado no domínio associado ao seu grupo de usuários. Você tem duas opções para configurar esse domínio: usar o domínio hospedado padrão do Amazon Cognito ou configurar um domínio personalizado de sua propriedade.

A opção de domínio personalizado tem mais opções de flexibilidade, segurança e controle. Por exemplo, um domínio familiar de propriedade da organização pode estimular a confiança do usuário e tornar o processo de login mais intuitivo. No entanto, a abordagem de domínio personalizado exige certa sobrecarga adicional, como gerenciar o certificado SSL e a configuração do DNS.

Os endpoints de descoberta do OIDC, `/.well-known/openid-configuration` para endpoints URLs e `/.well-known/jwks.json` chaves de assinatura de token, não estão hospedados em seu domínio. Para obter mais informações, consulte Provedor de identidades e endpoints de terceiros confiáveis.

Entender como configurar e gerenciar o domínio para seu grupo de usuários é uma etapa importante para integrar a autenticação na aplicação. Fazer login com a API de grupos de usuários e um AWS SDK pode ser uma alternativa à configuração de um domínio. O modelo baseado em API fornece tokens diretamente em uma resposta de API, mas para implementações que usam os recursos estendidos dos grupos de usuários, como um IdP do OIDC, você deve configurar um domínio. Para obter mais informações sobre modelos de autenticação disponíveis em grupos de usuários, consulteNoções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado.

Tópicos

- Informações importantes sobre domínios de grupo de usuários
- Usar o domínio de prefixo do Amazon Cognito para login gerenciado
- Usar o próprio domínio para fazer login gerenciado

Informações importantes sobre domínios de grupo de usuários

Os domínios de grupo de usuários são um ponto de serviço para as partes que dependem do OIDC em aplicações e para os elementos da interface do usuário. Considere os detalhes a seguir ao planejar a implementação de um domínio para o grupo de usuários.

Termos reservados

Não é possível usar o texto `aws`, `amazon` ou `cognito` no nome de um domínio com prefixo do Amazon Cognito.

Os endpoints de descoberta estão em um domínio diferente

Os endpoints de descoberta `.well-known/openid-configuration` e `.well-known/jwks.json` do grupo de usuários não estão no domínio personalizado ou do prefixo do grupo de usuários. O caminho para esses endpoints é o apresentado a seguir.

- `https://cognito-idp.`*`Region`*`.amazonaws.com/`*`your user pool ID`*`/.well-known/openid-configuration`
- `https://cognito-idp.`*`Region`*`.amazonaws.com/`*`your user pool ID`*`/.well-known/jwks.json`

Tempo efetivo de mudanças de domínio

O Amazon Cognito pode levar até 1 minuto para iniciar ou atualizar a versão de marca de um domínio de prefixo. As mudanças em um domínio personalizado podem levar até 5 minutos para serem propagadas. Os novos domínios personalizados podem levar até 1 hora para serem propagados.

Domínios personalizados e do prefixo ao mesmo tempo

Você pode configurar um grupo de usuários com um domínio personalizado e um domínio de prefixo de propriedade AWS da. Como os endpoints de descoberta do grupo de usuários estão hospedados em um domínio diferente, eles atendem apenas ao domínio personalizado. Por exemplo, o `openid-configuration` fornecerá um valor único para `"authorization_endpoint"` de `"https://auth.example.com/oauth2/authorize"`.

Quando você tem domínios personalizados e do prefixo em um grupo de usuários, pode usar o domínio personalizado com todos os recursos de um provedor OIDC. O domínio de prefixo em um grupo de usuários com essa configuração não tem descoberta nem token-signing-key endpoints e deve ser usado adequadamente.

Domínios personalizados preferenciais como ID de parte confiável para chave de acesso

Ao configurar a autenticação do grupo de usuários com chaves de acesso, você deve definir uma ID de parte confiável (RP). Quando você tem um domínio personalizado e um domínio de prefixo, pode definir o ID de RP somente como seu domínio personalizado. Para definir um domínio de prefixo como o ID de RP no console do Amazon Cognito, exclua seu domínio personalizado ou insira o nome de domínio totalmente qualificado (FQDN) do domínio de prefixo como um domínio de terceiros.

Não use domínios personalizados em diferentes níveis da sua hierarquia de domínios

Você pode configurar grupos de usuários separados para ter domínios personalizados no mesmo domínio de primeiro nível (TLD), por exemplo auth.example.com e auth2.example.com. O cookie de sessão do login gerenciado é válido para um domínio personalizado e todos os subdomínios, por exemplo, *.auth.example.com. Por esse motivo, nenhum usuário de suas aplicações deve acessar o login gerenciado de qualquer domínio principal ou subdomínio. Quando os domínios personalizados usarem o mesmo TLD, mantenha-os no mesmo nível de subdomínio.

Digamos que você tenha um grupo de usuários com o domínio personalizado auth.example.com. E então você cria outro grupo de usuários e atribui o domínio personalizado uk.auth.example.com. O usuário faz login com auth.example.com. e recebe um cookie que seu navegador apresenta

a qualquer site no caminho curinga *.auth.example.com. Em seguida, ele tenta fazer login em uk.auth.example.com.. Ele transmite um cookie inválido para o domínio do grupo de usuários e recebe um erro em vez de uma solicitação de login. Por outro lado, um usuário com um cookie para *.auth.example.com não tem problemas em iniciar uma sessão de login em auth2.example.com.

Versão de marca

Ao criar um domínio, você define uma Versão de marca. Suas opções são a nova experiência de login gerenciado e a experiência de IU hospedada clássica. Essa opção se aplica a todos os clientes da aplicação que hospedam serviços em seu domínio.

Usar o domínio de prefixo do Amazon Cognito para login gerenciado

A experiência padrão para login gerenciado é hospedada em um domínio que AWS possui. Essa abordagem é bem simples de usar: basta um nome de prefixo e a ativação estará feita. Porém, esse domínio não tem recursos que inspiram a confiança de um domínio personalizado. Não há diferença de custo entre a opção de domínio do Amazon Cognito e a opção de domínio personalizado. A única diferença é o domínio no endereço da web para o qual você direciona seus usuários. Para casos de redirecionamentos de IdP de terceiros e fluxos de credenciais de clientes, o domínio hospedado tem pouco efeito aparente. Um domínio personalizado é melhor nos casos em que os usuários fazem login com o login gerenciado e interagem com um domínio de autenticação que não corresponde ao domínio da aplicação.

O domínio hospedado do Amazon Cognito tem um prefixo de sua escolha, mas está hospedado no domínio raiz, `amazoncognito.com`. Este é um exemplo:

```
https://cognitoexample.auth.ap-south-1.amazoncognito.com
```

Todos os domínios de prefixo seguem este formato: *`prefix`*`.auth.`*`Região da AWS code`*`.amazoncognito.com`. Grupos de usuários de domínio personalizado podem hospedar as páginas de IU hospedada e login gerenciado em qualquer domínio que seja de sua propriedade.

> **Note**
>
> Para aumentar a segurança das aplicações do Amazon Cognito, os domínios principais dos endpoints do grupo de usuários são registrados na Public Suffix List (PSL). O PSL ajuda os navegadores da web dos usuários a estabelecer uma compreensão consistente dos endpoints do grupo de usuários e dos cookies que eles definem.
> Os domínios principais do grupo de usuários usam os formatos a seguir.

```
auth.Region.amazoncognito.com
auth-fips.Region.amazoncognito.com
```

Para adicionar um cliente de aplicativo e um domínio de grupo de usuários com o. Console de gerenciamento da AWS, consulteCriar um cliente de aplicação.

Tópicos

- Pré-requisitos
- Como configurar um prefixo de domínio do Amazon Cognito
- Verificar a página de login

Pré-requisitos

Antes de começar, você precisa de:

- Um grupo de usuários com um cliente de aplicativo. Para obter mais informações, consulte Conceitos básicos dos grupos de usuários.

Como configurar um prefixo de domínio do Amazon Cognito

Você pode usar a API Console de gerenciamento da AWS ou a AWS CLI ou para configurar um domínio de grupo de usuários.

Amazon Cognito console

Configurar um domínio

1. Navegue até o menu Domínio em Identidade visual.
2. Ao lado de Domínio, selecione Ações e clique em Criar domínio do Cognito. Se já tiver configurado um domínio de prefixo de grupo de usuários, selecione Excluir domínio do Cognito antes de criar seu novo domínio personalizado.
3. Insira um prefixo de domínio disponível para usar com um Domínio do Amazon Cognito. Para obter mais informações sobre como configurar um Domínio personalizado, consulte Usar o próprio domínio para fazer login gerenciado.

4. Escolha uma Versão de marca. Sua versão de marca se aplica a todas as páginas interativas nesse domínio. Seu grupo de usuários pode hospedar a identidade visual do login gerenciado ou da IU hospedada para todos os clientes da aplicação.

   > **Note**
   >
   > Você pode ter um domínio personalizado e um domínio de prefixo, mas o Amazon Cognito só fornece o endpoint `/.well-known/openid-configuration` para o domínio personalizado.

5. Escolha Criar.

CLI/API

Use os comandos a seguir para criar um prefixo de domínio personalizado e atribuí-lo ao grupo de usuários.

Para configurar um domínio de grupo de usuários

- AWS CLI: `aws cognito-idp create-user-pool-domain`

  Exemplo: `aws cognito-idp create-user-pool-domain --user-pool-id <user_pool_id> --domain <domain_name> --managed-login-version 2`
- Operação da API de grupos de usuários: CreateUserPoolDomain

Para obter informações sobre um domínio

- AWS CLI: `aws cognito-idp describe-user-pool-domain`

  Exemplo: `aws cognito-idp describe-user-pool-domain --domain <domain_name>`
- Operação da API de grupos de usuários: DescribeUserPoolDomain

Como excluir um domínio

- AWS CLI: `aws cognito-idp delete-user-pool-domain`

  Exemplo: `aws cognito-idp delete-user-pool-domain --domain <domain_name>`

- Operação da API de grupos de usuários: DeleteUserPoolDomain

Verificar a página de login

- Verifique se a página de login está disponível no seu domínio hospedado do Amazon Cognito.

```
https://<your_domain>/login?
response_type=code&client_id=<your_app_client_id>&redirect_uri=<your_callback_url>
```

O domínio é exibido na página Domain name (Nome do domínio) do console do Amazon Cognito. O ID de cliente do aplicativo e o URL de retorno de chamada são exibidos na página App client settings (Configurações do cliente do aplicativo).

Usar o próprio domínio para fazer login gerenciado

Após configurar um cliente da aplicação, você poderá configurar o grupo de usuários com um domínio personalizado para os serviços de domínio do login gerenciado. Com um domínio personalizado, os usuários podem fazer login na aplicação usando seu próprio endereço da web em vez do domínio de prefixo `amazoncognito.com`. Domínios personalizados melhoram a confiança do usuário em sua aplicação com um nome de domínio familiar, especialmente quando o domínio raiz corresponde ao domínio que hospeda a aplicação. Os domínios personalizados podem melhorar a conformidade com os requisitos de segurança da organização.

Um domínio personalizado tem alguns pré-requisitos, incluindo um grupo de usuários, um cliente da aplicação e um domínio da web de sua propriedade. Os domínios personalizados também exigem um certificado SSL para o domínio personalizado, gerenciado com AWS Certificate Manager (ACM) no Leste dos EUA (Norte da Virgínia). O Amazon Cognito cria uma CloudFront distribuição da Amazon, protegida em trânsito com seu certificado ACM. Como você é proprietário do domínio, você deve criar um registro DNS que direcione o tráfego para a CloudFront distribuição do seu domínio personalizado.

Com esses elementos prontos, você pode adicionar o domínio personalizado ao grupo de usuários por meio do console ou da API do Amazon Cognito. Isso envolve especificar o nome de domínio e o certificado SSL e, em seguida, atualizar sua configuração de DNS com o destino de alias fornecido. Depois de fazer essas alterações, você pode verificar se a página de login está acessível no seu domínio personalizado.

A maneira mais simples de criar um domínio personalizado é com uma zona hospedada pública no Amazon Route 53. O console do Amazon Cognito pode criar os registros DNS corretos em algumas etapas. Antes de começar, considere criar uma zona hospedada do Route 53 para um domínio ou subdomínio que você possua.

Tópicos

Como adicionar um domínio personalizado a um grupo de usuários

Para adicionar um domínio personalizado para seu grupo de usuários, especifique o nome de domínio no console do Amazon Cognito e forneça um certificado que você gerencia com o AWS Certificate Manager (ACM). Depois de adicionar seu domínio, o Amazon Cognito fornece um destino de alias, que você adiciona à sua configuração de DNS.

Pré-requisitos

Antes de começar, você precisa de:

- Um grupo de usuários com um cliente de aplicativo. Para obter mais informações, consulte Conceitos básicos dos grupos de usuários.
- Um domínio da Web do qual você é proprietário. O domínio superior deve ter um registro A DNS válido. Você pode atribuir qualquer valor a esse registro. O domínio superior pode ser a raiz do domínio ou um domínio inferior que fica um nível acima na hierarquia do domínio. Por exemplo, se o domínio personalizado for auth.xyz.exemplo.com, o Amazon Cognito precisará ser capaz de resolver xyz.exemplo.com como um endereço IP. Para evitar um impacto acidental na infraestrutura do cliente, o Amazon Cognito não suporta o uso de domínios de primeiro nível TLDs () para domínios personalizados. Para obter mais informações, consulte Nomes de domínio.
- A capacidade de criar um subdomínio para seu domínio personalizado. Recomendamos auth para o nome do subdomínio. Por exemplo: `auth.example.com`.

> **Note**
>
> Poderá ser necessário obter um novo certificado para o subdomínio do domínio personalizado se você não tiver um certificado curinga.

- Um SSL/TLS certificado público gerenciado pela ACM no Leste dos EUA (Norte da Virgínia). O certificado deve estar em us-east-1 porque será associado a uma distribuição CloudFront em, um serviço global.
- Clientes de navegador compatíveis com Server Name Indication (SNI). A CloudFront distribuição que o Amazon Cognito atribui aos domínios personalizados requer SNI. Você não pode alterar essa configuração. Para obter mais informações sobre o SNI nas CloudFront distribuições, consulte Usar o SNI para atender solicitações HTTPS no Amazon CloudFront Developer Guide.
- Uma aplicação que permite que o servidor de autorização do grupo de usuários adicione cookies às sessões do usuário. O Amazon Cognito define vários cookies obrigatórios para páginas de login gerenciado. Entre eles estão `cognito`, `cognito-fl` e `XSRF-TOKEN`. Embora cada cookie individual respeite os limites de tamanho do navegador, alterações na configuração do grupo de usuários podem fazer com que os cookies do login gerenciado aumentem de tamanho. Um serviço intermediário, como o Application Load Balancer (ALB), na frente do domínio personalizado pode impor um tamanho máximo de cabeçalho ou tamanho total do cookie. Se a aplicação também definir seus próprios cookies, as sessões dos usuários poderão exceder esses limites. Para evitar conflitos de limite de tamanho, recomendamos que a aplicação não defina cookies no subdomínio que hospeda os serviços de domínio do grupo de usuários.
- Permissão para atualizar as CloudFront distribuições da Amazon. Você pode fazer isso anexando a declaração de política do IAM a seguir a um usuário em sua Conta da AWS:

  JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
         {
            "Sid": "AllowCloudFrontUpdateDistribution",
            "Effect": "Allow",
            "Action": [
                "cloudfront:updateDistribution"
            ],
            "Resource": [
                "*"
```

```
            ]
        }
    ]
}
```

Para obter mais informações sobre como autorizar ações em CloudFront, consulte Usando políticas baseadas em identidade (políticas do IAM) para. CloudFront

O Amazon Cognito inicialmente usa suas permissões do IAM para configurar a CloudFront distribuição, mas a distribuição é gerenciada por. AWS Você não pode alterar a configuração da CloudFront distribuição que o Amazon Cognito associou ao seu grupo de usuários. Por exemplo, não é possível atualizar as versões de TLS compatíveis na política de segurança.

Etapa 1: insira o nome de domínio personalizado

É possível adicionar seu domínio ao grupo de usuários usando a API ou o console do Amazon Cognito.

Amazon Cognito console

Para adicionar o domínio ao grupo de usuários diretamente do console do Amazon Cognito:

1. Navegue até o menu Domínio em Identidade visual.
2. Ao lado de Domínio, escolha Ações e Criar domínio personalizado ou Criar domínio do Amazon Cognito. Se já tiver configurado um domínio personalizado de grupo de usuários, clique em Excluir domínio personalizado antes de criar seu novo domínio personalizado.
3. Ao lado de Domínio, selecione Ações e clique em Criar domínio personalizado. Se você já configurou um domínio personalizado, clique em Excluir domínio personalizado para excluir o domínio existente antes de criar seu novo domínio personalizado.
4. Para o Custom domain (Domínio personalizado), insira o URL do domínio que você deseja usar com o Amazon Cognito. Seu nome de domínio pode incluir somente letras minúsculas, números e hífens. Não use um hífen como primeiro ou último caractere. Use pontos para separar nomes de subdomínio.
5. Para o ACM certificate (Certificado do ACM), escolha o certificado SSL que você deseja usar para seu domínio. Somente certificados ACM no Leste dos EUA (Norte da Virgínia) estão qualificados para uso com um domínio personalizado do Amazon Cognito, independentemente Região da AWS do seu grupo de usuários.

Se você não tem um certificado disponível, poderá usar o ACM para implantar um no Leste dos EUA (Norte da Virgínia). Para obter mais informações, consulte Conceitos básicos no Guia do usuário do AWS Certificate Manager .

6. Escolha uma Versão de marca. Sua versão de marca se aplica a todas as páginas interativas nesse domínio. Seu grupo de usuários pode hospedar a identidade visual do login gerenciado ou da IU hospedada para todos os clientes da aplicação.

> **Note**
>
> Você pode ter um domínio personalizado e um domínio de prefixo, mas o Amazon Cognito só fornece o endpoint `/.well-known/openid-configuration` para o domínio personalizado.

7. Escolha Criar.
8. O Amazon Cognito retorna você ao menu Domínio. Uma mensagem intitulada Create an alias record in your domain's DNS (Criar um registro de alias no DNS do seu domínio) é exibida. Anote o Domain (Domínio) e Alias target (Destino do alias) exibidos no console. Eles serão usados na próxima etapa para direcionar o tráfego para o seu domínio personalizado.

API

O corpo da CreateUserPoolDomainsolicitação a seguir cria um domínio personalizado.

```
{
   "Domain": "auth.example.com",
   "UserPoolId": "us-east-1_EXAMPLE",
   "ManagedLoginVersion": 2,
   "CustomDomainConfig": {
    "CertificateArn": "arn:aws:acm:us-east-1:111122223333:certificate/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
   }
}
```

Etapa 2: adicionar um destino do alias e subdomínio

Nesta etapa, configure um alias por meio do seu provedor de serviços do servidor de nome de domínio (DNS) que aponta para o destino do alias da etapa anterior. Se você estiver usando o

Amazon Route 53 para resolução do endereço DNS, escolha a seção To add an alias target and subdomain using Route 53 (Para adicionar um destino do alias e subdomínio usando o Route 53).

Para adicionar um destino do alias e subdomínio à sua configuração atual do DNS

- Se não estiver usando o Route 53 para resolução do endereço DNS, é necessário utilizar as ferramentas de configuração do seu provedor de serviço DNS para adicionar o destino do alias da etapa anterior ao registro DNS do domínio. O provedor DNS também precisará configurar o subdomínio para o seu domínio personalizado.

Para adicionar um destino do alias e subdomínio usando o Route 53

1. Faça login no console do Route 53. Se solicitado, insira suas AWS credenciais.
2. Se não tiver uma zona hospedada pública no Route 53, crie uma com uma raiz que seja pai do seu domínio personalizado. Para obter mais informações, consulte Criar uma zona hospedada pública no Guia do desenvolvedor do Amazon Route 53.
   a. Escolha Criar zona hospedada.
   b. Insira o domínio principal, por exemplo*auth.example.com*, do seu domínio personalizado, por exemplo*myapp.auth.example.com*, na lista de nomes de domínio.
   c. Insira uma Descrição para a sua zona hospedada.
   d. Selecione um Type (Tipo) de zona hospedada de Public hosted zone (Zona hospedada pública) para permitir que clientes públicos resolvam seu domínio personalizado. Não há compatibilidade com a seleção de Private hosted zone (Zona hospedada privada).
   e. Aplique Etiquetas como desejar.
   f. Escolha Criar zona hospedada.

> **Note**
>
> Também é possível criar uma nova zona hospedada para o domínio personalizado com um conjunto de delegação na zona hospedada principal que direciona consultas para a zona hospedada do subdomínio. Caso contrário, crie um registro A. Esse método oferece mais flexibilidade e segurança com suas zonas hospedadas. Para mais informações, consulte Creating a subdomain for a domain hosted through Amazon Route 53 (Criar um subdomínio para um domínio hospedado por meio do Amazon Route 53).

3. Na página Hosted Zones (Zonas hospedadas), escolha o nome da sua zona hospedada.
4. Adicione um registro DNS ao domínio pai do seu domínio personalizado, caso ainda não tenha um. Crie um registro DNS para o domínio pai com as propriedades a seguir:

   - Nome do registro: deixe em branco.
   - Tipo de registro: `A`.
   - Alias: não ative.
   - Valor: insira o valor desejado. Esse registro deve ser resolvido como algo, mas o valor do registro não importa para o Amazon Cognito.
   - TTL: defina como seu TTL preferido ou deixe o padrão.
   - Política de roteamento: escolha roteamento simples.
5. Escolha Criar registros. Veja a seguir um exemplo de registro para o domínio*example.com*:

   ```
   example.com. 60 IN A 198.51.100.1
   ```

   > **Note**
   >
   > O Amazon Cognito verifica que há um registro DNS para o domínio pai do seu domínio personalizado para proteger contra o sequestro acidental de domínios de produção. Se você não tiver um registro DNS para o domínio pai, o Amazon Cognito retornará um erro quando você tentar definir o domínio personalizado. Um registro de Início de autoridade (SOA) não é um registro DNS suficiente para fins de verificação do domínio pai.
6. Adicione outro registro DNS para seu domínio personalizado com as seguintes propriedades:

   - Nome do registro: prefixo do domínio personalizado, por exemplo, `auth` para criar um registro para `auth.example.com`.
   - Tipo de registro: `A`.
   - Alias: ative.
   - Rotear tráfego para: escolha Alias para distribuição do CloudFront. Insira o Destino do alias que você registrou anteriormente, por exemplo, `123example.cloudfront.net`.
   - Política de roteamento: escolha roteamento simples.
7. Escolha Criar registros.

> **Note**
>
> A propagação de seus novos registros para todos os servidores de DNS do Route 53 pode levar cerca de 60 segundos. Você pode usar o método da GetChangeAPI Route 53 para verificar se suas alterações foram propagadas.

Etapa 3: verificar a página de acesso

- Verifique se a página de login está disponível no seu domínio personalizado.

  Faça login com o domínio e o subdomínio personalizados inserindo esse endereço no navegador. Este é um exemplo de URL de um domínio personalizado *example.com* com o subdomínio: *auth*

```
https://myapp.auth.example.com/login?
response_type=code&client_id=<your_app_client_id>&redirect_uri=<your_callback_url>
```

Como alterar o certificado SSL do seu domínio personalizado

Quando necessário, você poderá usar o Amazon Cognito para alterar o certificado aplicado ao seu domínio personalizado.

Geralmente, isso é desnecessário ao ser seguida a rotina de renovação do certificado com o ACM. Quando você renovar seu certificado existente no ACM, o ARN do certificado permanecerá o mesmo, e seu domínio personalizado usará o novo certificado automaticamente.

No entanto, se você substituir o certificado existente por um novo, o ACM dará ao novo certificado um novo ARN. Para aplicar o novo certificado ao seu domínio personalizado, você deve fornecer esse ARN ao Amazon Cognito.

Depois de fornecer o novo certificado, o Amazon Cognito precisará de até uma hora para distribuí-lo ao seu domínio personalizado.

> **Antes de começar**
>
> Antes de alterar seu certificado no Amazon Cognito, você deve adicionar o certificado ao ACM. Para obter mais informações, consulte Conceitos básicos no Guia do usuário do AWS Certificate Manager .
> Ao adicionar seu certificado ao ACM, você deve escolher Leste dos EUA (Norte da Virgínia) como região da AWS .

É possível alterar seu certificado usando a API ou o console do Amazon Cognito.

Console de gerenciamento da AWS

Para renovar um certificado no console do Amazon Cognito:

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Cognito em. https://console.aws.amazon.com/cognito/home
2. Escolha User Pools (Grupos de usuários).
3. Escolha o grupo de usuários para o qual deseja atualizar o certificado.
4. Clique no menu Domínio.
5. Escolha Actions (Ações), Edit ACM certificate (Editar certificado do ACM).
6. Selecione o novo certificado que deseja associar ao seu domínio personalizado.
7. Escolha Salvar alterações.

API

Para renovar um certificado (API do Amazon Cognito)

- Use a ação UpdateUserPoolDomain.

Aplicar a identidade visual às páginas de login gerenciado

É recomendável proporcionar uma experiência de usuário consistente entre o serviço de autenticação e a aplicação. Você pode atingir essa meta com formulários personalizados e operações de API de back-end em um AWS SDK ou com login gerenciado. O login gerenciado e a IU hospedada clássica são frontends da web para o componente da sua aplicação que fornece autenticação com grupos de usuários. Para sincronizar os serviços de autenticação gerenciada

com a UX da aplicação, existem duas opções de personalização: o editor de identidade visual e a identidade visual de IU hospedada. Escolha a experiência mais indicada para você no console do Amazon Cognito e com as operações de API do grupo de usuários.

O editor de identidade visual

O editor de identidade visual é a mais nova opção de personalização para a mais nova experiência de IU de grupos de usuários, o login gerenciado. O editor de identidade visual é um editor visual no-code para estilos e ativos de login gerenciado e um conjunto de operações de API para configuração programática de inúmeras opções de configuração. Os grupos de usuários configurados com um domínio e um login gerenciado exibem automaticamente a versão de designer de marcas de suas páginas de login.

Identidade visual de IU hospedada (clássica)

A experiência de marca da IU (clássica) tem duas opções: modificar um arquivo Cascading Style Sheets (CSS) com um conjunto fixo de opções de estilo e adicionar uma imagem de logotipo personalizada. Você pode definir essas opções no console do Amazon Cognito ou com a operação Set UICustomization API. Quando o serviço foi lançado, o Amazon Cognito tinha somente essa opção. Os grupos de usuários configurados com um domínio e a versão de marca da IU hospedada exibem automaticamente a versão clássica de suas páginas de login. Seu plano de recursos também pode ser compatível somente com a interface hospedada.

> **Note**
>
> O editor de identidade visual e a experiência de identidade visual clássica modificam as propriedades visuais do seu serviço de autenticação hospedado. Atualmente, não é possível modificar o texto exibido em suas páginas de login gerenciado, exceto para aplicar a localização em um dos vários idiomas. Para obter mais informações sobre localização, consulte Managed login localization.

Escolher uma experiência de identidade visual e atribuir estilos

No console do Amazon Cognito, novos grupos de usuários usam como padrão a experiência de identidade visual de login gerenciado. Os grupos de usuários configurados antes da disponibilização do login gerenciado terão a identidade visual de IU hospedada (clássica). Você pode alternar entre a identidade visual de login gerenciado e de IU hospedada. Quando você altera sua versão de marca, o Amazon Cognito aplica imediatamente a alteração às páginas interativas do domínio do seu grupo

de usuários. Com o login gerenciado e a IU hospedada, seu grupo de usuários pode ter um estilo para cada cliente de aplicação.

Cada cliente de aplicação pode ter um estilo de identidade visual distinto, mas um domínio do grupo de usuários serve tanto para o login gerenciado quanto para a IU hospedada. Um estilo é o conjunto de configurações de personalização aplicadas a um cliente de aplicação. Você pode configurar um domínio personalizado e um domínio de prefixo por grupo de usuários. Você pode atribuir diferentes versões de marca aos seus domínios personalizados e de prefixo. No entanto, um domínio de prefixo não é totalmente funcional quando você também tem um domínio personalizado; os endpoints de descoberta `.well-known` do OIDC apresentam somente caminhos de domínio personalizados. Você só pode usar o domínio de prefixo para operações que não exijam descoberta de endpoint (`openid-configuration`) em um grupo de usuários com essa configuração. Devido a essas propriedades dos grupos de usuários, é possível escolher efetivamente uma versão de marca por grupo de usuários.

Você pode atribuir estilos aos clientes da aplicação em um grupo de usuários em que um domínio é definido para a versão da marca de login gerenciado. Estilos são um conjunto de configurações visuais composto por arquivos de imagem, opções de exibição e valores CSS. Quando você atribui um estilo a um cliente de aplicação, o Amazon Cognito envia imediatamente suas atualizações para suas páginas de login interativas. O Amazon Cognito exibe suas páginas interativas com a versão de identidade visual escolhida e a personalização que você aplicou a ela.

Atualizar e excluir estilos

Ao criar um estilo, você o vincula a um cliente de aplicação. Para alterar uma atribuição de estilo para um cliente de aplicação, primeiro exclua o estilo original. No momento, não é possível copiar configurações entre estilos. Isso deve ser feito de maneira programática. Para replicar as configurações entre estilos e clientes de aplicativos, obtenha as configurações de um estilo com a operação da DescribeManagedLoginBrandingAPI e aplique-as com CreateManagedLoginBrandingou UpdateManagedLoginBranding. Você não pode alterar os estilos atribuídos de um cliente de aplicação. Você pode somente excluir o original e definir um novo. Para obter mais informações sobre como gerenciar estilos com operações de API e SDK, consulte Operações de API e SDK para identidade visual de login gerenciado.

> **Note**
>
> As solicitações programáticas que criam ou atualizam o estilo de identidade visual devem ter um tamanho de solicitação não superior a 2 MB. Se a solicitação for maior que esse limite, divida-a em várias solicitações `UpdateManagedLoginBranding` para grupos de

parâmetros que não excedam o tamanho máximo da solicitação. Essas solicitações não resultam na definição de parâmetros não especificados como padrão, portanto, você pode enviar solicitações parciais sem afetar as configurações existentes.

Para excluir um estilo no console do Amazon Cognito, acesse o menu Login gerenciado. Em Estilos, escolha o estilo que deseja excluir e clique em Excluir estilo.

Em linhas gerais, o processo de atribuição de identidade visual a um domínio consiste nas etapas a seguir.

1. Crie um domínio e defina a versão da marca.
2. Crie um estilo de identidade visual e atribua-o a um cliente de aplicação.

Como atribuir um estilo a um cliente de aplicação

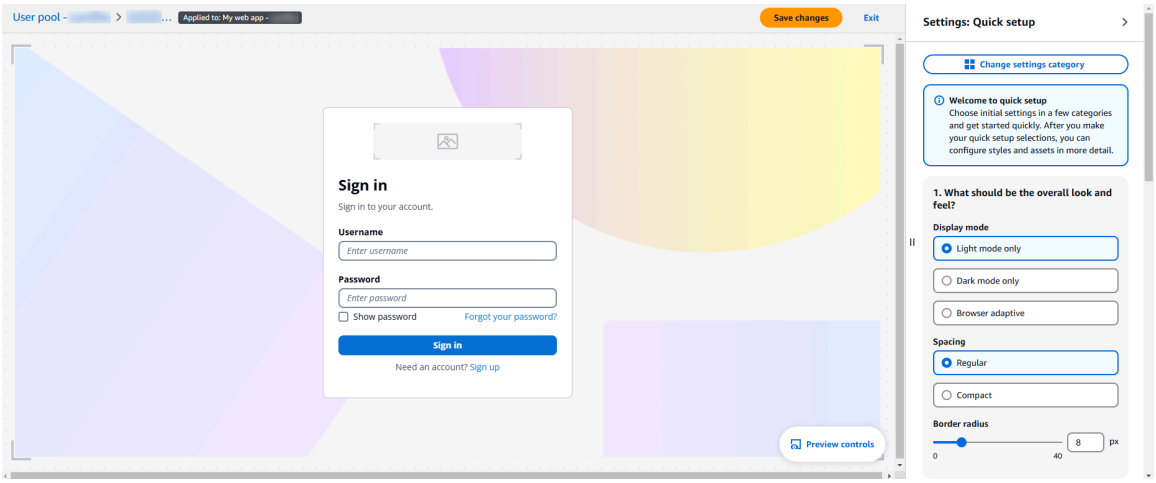
1. No menu Domínio do grupo de usuários, crie um domínio e defina a Versão de marca como Login gerenciado.
2. Acesse o menu Login gerenciado. Em Estilos, selecione Criar um estilo.
3. Escolha o cliente de aplicação ao qual deseja atribuir seu estilo ou crie um novo cliente de aplicação.
4. Para começar a definir suas configurações de marca, selecione Iniciar editor de identidade visual.

Tópicos

- O editor de identidade visual e a identidade visual de login gerenciado
- Personalizar identidade visual de IU hospedada (clássica)

O editor de identidade visual e a identidade visual de login gerenciado

O editor de identidade visual é uma ferramenta visual de design e edição para suas páginas da web de login gerenciado. Ele está integrado ao console do Amazon Cognito. No editor de identidade visual, você começa com uma prévia de suas páginas de login e pode prosseguir para uma opção de configuração rápida ou uma visualização detalhada com opções avançadas. É possível modificar e visualizar parâmetros de estilo ou adicionar uma imagem de fundo e um logotipo personalizados. Também é possível configurar o modo claro e o modo escuro.



Para começar, crie um estilo que você possa aplicar ao seu grupo de usuários ou a um cliente de aplicação.

Como começar a usar o editor de identidade visual

1. Crie um domínio na guia Domínio ou atualize seu domínio existente. Em Versão de marca, defina seu domínio para usar o Login gerenciado.
2. Exclua o estilo de cliente de aplicação existente, se houver.
   a. No menu Clientes da aplicação, selecione seu cliente de aplicação.
   b. Em Estilo de login gerenciado, selecione o estilo atribuído ao seu cliente de aplicação.
   c. Selecione Excluir estilo. Confirme a seleção.
3. Acesse o menu Login gerenciado no grupo de usuários. Siga as instruções para selecionar um plano de recursos que inclua login gerenciado, caso ainda não o tenha feito. Você também pode selecionar Visualizar este recurso se quiser conferir o editor de identidade visual sem fazer alterações.
4. Em Estilos, selecione Criar um estilo.
5. Escolha o cliente de aplicação ao qual deseja atribuir seu estilo e selecione Criar. Você também pode criar um novo cliente de aplicação.
6. O console do Amazon Cognito inicia o editor de identidade visual.
7. Escolha uma guia na qual deseja começar a editar ou selecione Iniciar editor e acesse a configuração rápida. As seguintes guias estão disponíveis:

   Demonstração

   Veja como suas seleções atuais aparecem em suas páginas de login gerenciado.

Fundamentos

Defina um tema geral, configure links para provedores de identidades externos e estilize os campos do formulário.

Componentes

Configure estilos para cabeçalhos, rodapés e elementos individuais da IU.

8. Para definir as configurações iniciais, acesse a configuração rápida. Selecione Alterar categoria de configurações e clique em Configuração rápida. Ao selecionar Prosseguir, o editor de identidade visual será iniciado com um conjunto de opções básicas para você configurar.

Texto e localização

Não é possível modificar nem localizar texto no editor de identidade visual. Em vez disso, adicione um parâmetro de consulta `lang` ao URL que você distribui aos usuários. Esse parâmetro fará com que suas páginas de login gerenciado sejam localizadas em um dos vários idiomas disponíveis. Para obter mais informações, consulte Managed login localization.

Configuração rápida

O botão Iniciar editor de identidade visual carrega um editor visual para a configuração do login gerenciado, onde você pode selecionar entre diversas opções básicas de personalização. Conforme você faz as seleções, o Amazon Cognito exibe as alterações do login gerenciado em uma janela de pré-visualização. Para retornar ao menu de configurações detalhadas, clique no botão Alterar categoria de configurações.

Qual deve ser a aparência geral?

Defina as configurações básicas do tema para o login gerenciado.

Modo de exibição

Escolha um modo claro, escuro ou uma experiência adaptável para seu login gerenciado. As configurações adaptáveis se referem à preferência do navegador do usuário quando o Amazon Cognito exibe o login gerenciado. Ao escolher um modo adaptável ao navegador, você pode escolher cores e imagens de logotipo diferentes para os modos claro e escuro.

Espaçamento

Defina o espaçamento padrão entre os elementos na página.

Raio da borda

Defina a profundidade de arredondamento da borda externa dos elementos.

Qual deve ser a aparência do plano de fundo da página?

Tipo de plano de fundo

A caixa de seleção Mostrar imagem indica se você deseja uma imagem de fundo ou definir uma cor de fundo sólida.

1. Para usar uma imagem, selecione Mostrar imagem e escolha uma imagem de fundo para os modos claro e escuro. Você também pode definir uma Cor do plano de fundo da página no modo escuro e no modo claro para áreas do plano de fundo que não estão cobertas pela imagem.
2. Para usar somente uma cor para o plano de fundo, desmarque Mostrar imagem e escolha a Cor do plano de fundo da página no modo claro e no modo escuro.

Qual deve ser a aparência dos formulários?

Defina as configurações dos elementos do formulário do login gerenciado. Exemplos de elementos do formulário incluem solicitações de login e código.

Alinhamento horizontal

Defina o alinhamento horizontal dos campos do formulário.

Logotipo do formulário

Defina o posicionamento da imagem do logotipo.

Imagem do logotipo

Escolha um arquivo de imagem de logotipo para incluir no elemento do formulário nos modos claro e escuro. Para carregar uma imagem, selecione o menu suspenso Imagem do logotipo, clique em Adicionar novo ativo e adicione um arquivo de logotipo.

Cor primária da identidade visual

Defina uma cor de tema para os modos claro e escuro. Essa cor será aplicada como cor de fundo a todos os elementos classificados como primários.

Qual deve ser a aparência dos cabeçalhos?

Escolha se você deseja incluir um cabeçalho em suas páginas de login gerenciado. O cabeçalho pode conter uma imagem de logotipo.

Logotipo do cabeçalho

Defina a posição da imagem do logotipo no cabeçalho.

Imagem do logotipo

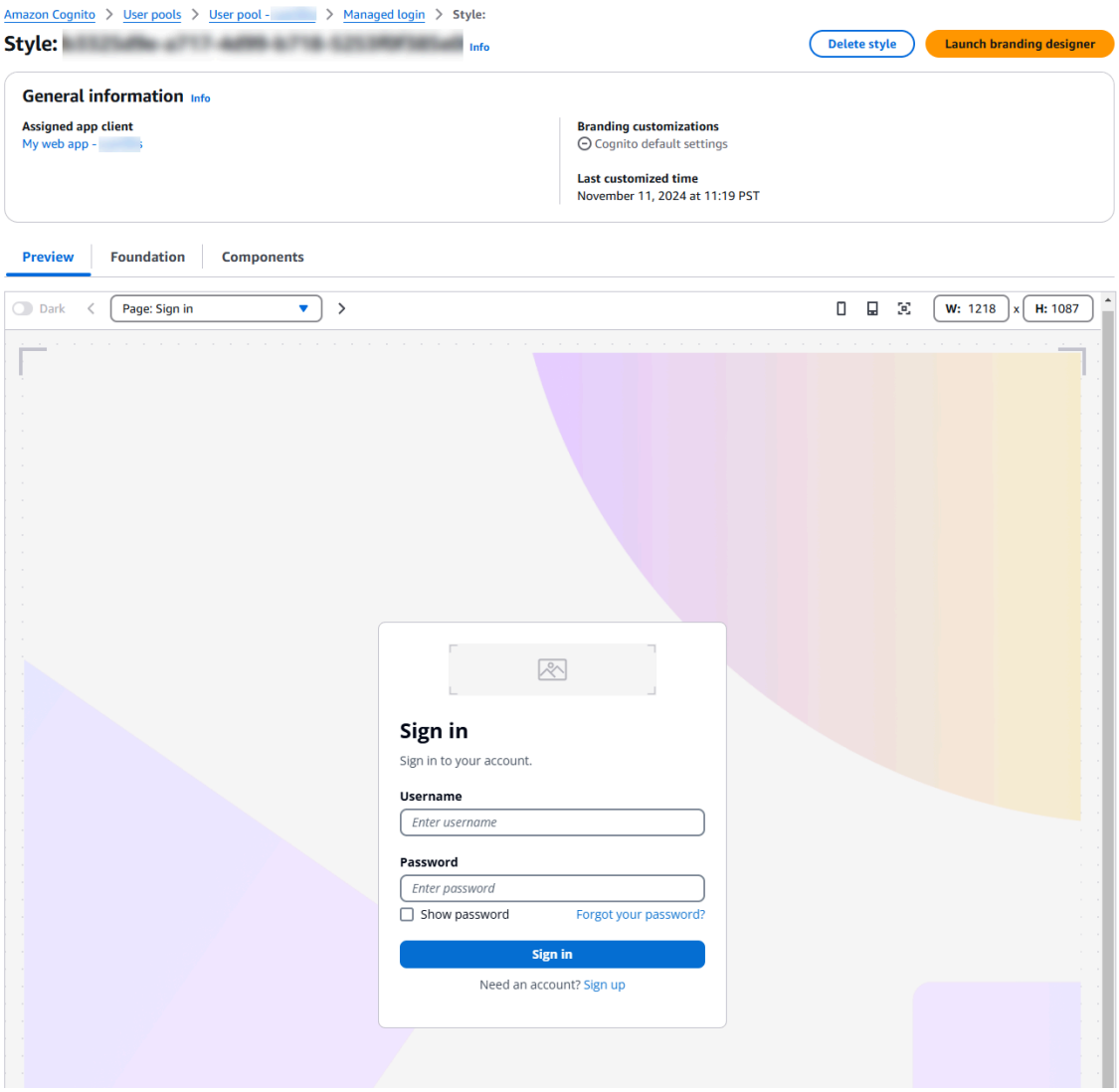
Escolha a posição do logotipo e um arquivo de imagem do logotipo para incluir no cabeçalho. Para carregar uma imagem, selecione o menu suspenso Imagem do logotipo, clique em Adicionar novo ativo e adicione um arquivo de logotipo.

Cor do plano de fundo do cabeçalho

Defina as cores dos modos claro e escuro para o plano de fundo do cabeçalho.

Configurações detalhadas

Na visualização de configurações detalhadas, você pode modificar componentes individuais na Fundação e nos Componentes. A guia Pré-visualização exibe uma prévia do login gerenciado no contexto atual com suas personalizações.



Para acessar o editor visual de um componente, selecione o ícone de edição no bloco do componente. No editor do estúdio de temas, é possível alternar entre os componentes com o botão Alterar categoria de configuração.

Fundamentos

Estilo da aplicação

Configure os aspectos básicos da configuração de login gerenciado. Esta categoria tem configurações para o tema geral, espaçamento de texto e cabeçalho e rodapé da página.

Modo de exibição

Escolha um modo claro, escuro ou uma experiência adaptável para suas páginas de login gerenciado. Ao escolher um modo adaptável ao navegador, você pode escolher cores e imagens de logotipo diferentes para os modos claro e escuro.

Espaçamento

Defina o espaçamento padrão entre os elementos na página.

Comportamento de autenticação

Configure estilos para os botões que conectam seus usuários a provedores de identidade externos (IdPs). Esta seção inclui a opção Entrada de pesquisa de domínio para que o login gerenciado solicite aos usuários um endereço de e-mail e os associe ao identificador do provedor de identidades SAML correspondente.

Comportamento do formulário

Configure estilos para formulários de entrada: posicionamento de entradas, cores e alinhamento de elementos.

Componentes

Botões

Estilos para botões que o Amazon Cognito exibe em páginas de login gerenciado.

Divisor

Estilos para linhas divisórias e limites entre elementos de login gerenciado, como o formulário de entrada e o seletor de login de provedor externo.

Suspenso

Estilos para menus suspensos.

Favicon

Estilos para a imagem que o Amazon Cognito fornece para o ícone de guia e marcador.

Anéis de foco

Estilos para os destaques que indicam uma entrada atualmente selecionada.

Contêiner de formulário

Estilos para os elementos que delimitam um formulário.

Rodapé global

Estilos para o rodapé que o Amazon Cognito exibe na parte inferior das páginas de login gerenciado.

Cabeçalho global

Estilos para o cabeçalho que o Amazon Cognito exibe na parte superior das páginas de login gerenciado.

Indicações

Estilos para mensagens de erro e sucesso.

Controles de opções

Estilos para caixas de seleção, seleções múltiplas e outros prompts de entrada.

Plano de fundo da página

Estilos para o plano de fundo geral do login gerenciado.

Entradas

Estilos para prompts de entrada de campo de formulário.

Link

Estilos para hiperlinks em páginas de login gerenciado.

Texto para página

Estilos para texto na página.

Texto para campo

Estilos para o texto ao redor dos campos de formulário.

Operações de API e SDK para identidade visual de login gerenciado

Você também pode aplicar a marca a um estilo de login gerenciado com as operações de API CreateManagedLoginBrandinge. UpdateManagedLoginBranding Essas operações são ideais para

criar versões idênticas ou ligeiramente modificadas de um estilo de identidade visual para outra aplicação, cliente ou grupo de usuários. Consulte a marca de login gerenciado de um estilo existente com a operação da API DescribeManagedLoginBranding, modifique a saída conforme necessário e aplique-a a outro recurso.

A operação `UpdateManagedLoginBranding` não altera o cliente de aplicação ao qual seu estilo é aplicado. Ela somente atualiza o estilo existente atribuído a um cliente de aplicação. Para substituir completamente o estilo de um cliente de aplicativo, exclua o estilo existente com DeleteManagedLoginBrandinge atribua um novo estilo com`CreateManagedLoginBranding`. No console do Amazon Cognito, o mesmo se aplica: você deve excluir o estilo existente e criar um novo.

Configurar a identidade visual de login gerenciado em uma solicitação de API ou SDK exige que suas configurações sejam incorporadas em um arquivo JSON convertido em um tipo de dados `Document`. Veja a seguir uma orientação sobre imagens que você pode adicionar e como gerar solicitações programáticas para configurar um estilo de identidade visual.

Ativos de imagem

CreateManagedLoginBrandinge UpdateManagedLoginBrandinginclua um `Assets` parâmetro. Esse parâmetro é uma matriz de arquivos de imagem em formato binário codificado em base64.

> **Note**
>
> As solicitações programáticas que criam ou atualizam o estilo de identidade visual devem ter um tamanho de solicitação não superior a 2 MB. Os ativos em sua solicitação podem fazer com que ela exceda esse limite. Se for o caso, divida sua solicitação em várias solicitações `UpdateManagedLoginBranding` para grupos de parâmetros que não excedam o tamanho máximo da solicitação. Essas solicitações não resultam na definição de parâmetros não especificados como padrão, portanto, você pode enviar solicitações parciais sem afetar as configurações existentes.

Alguns ativos têm limitações quanto aos tipos de arquivo que você pode enviar.

Ativo	Extensões de arquivo aceitas
FAVICON_ICO	ico
FAVICON_SVG	svg

Ativo	Extensões de arquivo aceitas
EMAIL_GRAPHIC	png, svg, jpeg
SMS_GRAPHIC	png, svg, jpeg
AUTH_APP_GRAPHIC	png, svg, jpeg
PASSWORD_GRAPHIC	png, svg, jpeg
PASSKEY_GRAPHIC	png, svg, jpeg
PAGE_HEADER_LOGO	png, svg, jpeg
PAGE_HEADER_BACKGROUND	png, svg, jpeg
PAGE_FOOTER_LOGO	png, svg, jpeg
PAGE_FOOTER_BACKGROUND	png, svg, jpeg
PAGE_BACKGROUND	png, svg, jpeg
FORM_BACKGROUND	png, svg, jpeg
FORM_LOGO	png, svg, jpeg
IDP_BUTTON_ICON	ico, svg

Os arquivos do tipo SVG são compatíveis com os atributos e elementos a seguir.

Attributes

```
accent-height, accumulate, additivive, alignment-baseline, ascent, attributename,
 attributetype, azimuth, basefrequency, baseline-shift, begin, bias, by, class,
 clip, clip-path, clip-rule, color, color-interpolation, color-interpolation-
filters, color-profile, color-rendering, cx, cy, d, dx, dy, diffuseconstant,
 direction, display, divisor, dur, edgemode, elevation, end, fill, fill-opacity,
 fill-rule, filter, filterunits, flood-color, flood-opacity, font-family, font-
size, font-size-adjust, font-stretch, font-style, font-variant, font-weight, fx,
 fy, g1, g2, glyph-name, glyphref, gradientunits, gradienttransform, height, href,
 id, image-rendering, in, in2, k, k1, k2, k3, k4, kerning, keypoints, keysplines,
 keytimes, lang, lengthadjust, letter-spacing, kernelmatrix, kernelunitlength,
```

```
 lighting-color, local, marker-end, marker-mid, marker-start, markerheight,
 markerunits, markerwidth, maskcontentunits, maskunits, max, mask, media,
 method, mode, min, name, numoctaves, offset, operator, opacity, order, orient,
 orientation, origin, overflow, paint-order, path, pathlength, patterncontentunits,
 patterntransform, patternunits, points, preservealpha, preserveaspectratio, r,
 rx, ry, radius, refx, refy, repeatcount, repeatdur, restart, result, rotate,
 scale, seed, shape-rendering, specularconstant, specularexponent, spreadmethod,
 stddeviation, stitchtiles, stop-color, stop-opacity, stroke-dasharray, stroke-
dashoffset, stroke-linecap, stroke-linejoin, stroke-miterlimit, stroke-opacity,
 stroke, stroke-width, style, surfacescale, tabindex, targetx, targety, transform,
 text-anchor, text-decoration, text-rendering, textlength, type, u1, u2, unicode,
 values, viewbox, visibility, vert-adv-y, vert-origin-x, vert-origin-y, width, word-
spacing, wrap, writing-mode, xchannelselector, ychannelselector, x, x1, x2, xmlns,
 y, y1, y2, z, zoomandpan
```

Elements

```
svg, a, altglyph, altglyphdef, altglyphitem, animatecolor, animatemotion,
 animatetransform, audio, canvas, circle, clippath, defs, desc, ellipse, filter,
 font, g, glyph, glyphref, hkern, image, line, lineargradient, marker, mask,
 metadata, mpath, path, pattern, polygon, polyline, radialgradient, rect, stop,
 style, switch, symbol, text, textpath, title, tref, tspan, video, view, vkern,
 feBlend, feColorMatrix, feComponentTransfer, feComposite, feConvolveMatrix,
 feDiffuseLighting, feDisplacementMap, feDistantLight, feFlood, feFuncA, feFuncB,
 feFuncG, feFuncR, feGaussianBlur, feMerge, feMergeNode, feMorphology, feOffset,
 fePointLight, feSpecularLighting, feSpotLight, feTile, feTurbulence
```

Ferramentas para operações de identidade visual de login gerenciado

O Amazon Cognito gerencia um arquivo no formato de esquema JSON para o objeto de configurações de identidade visual de login gerenciado. Veja a seguir como atualizar programaticamente seu estilo de identidade visual.

Como atualizar a identidade visual na API de grupos de usuários

1. No console do Amazon Cognito, crie um estilo padrão de identidade visual de login gerenciado no menu Login gerenciado do grupo de usuários. Atribua-o a um cliente de aplicação.
2. Registre o ID do cliente de aplicação para o qual você criou o estilo, por exemplo, `1example23456789`.

3. Recupere as configurações do estilo de marca com uma solicitação de DescribeManagedLoginBrandingByClientAPI `ReturnMergedResources` definida como. `true` Veja a seguir um exemplo de corpo da solicitação .

```
{
   "ClientId": "1example23456789",
   "ReturnMergedResources": true,
   "UserPoolId": "us-east-1_EXAMPLE"
}
```

4. Modifique a saída `DescribeManagedLoginBrandingByClient` com suas personalizações.

   a. O corpo da resposta está envolvido em um elemento `ManagedLoginBranding` que não faz parte da sintaxe das operações de criação e atualização. Remova esse nível superior do objeto JSON.

   b. Para substituir imagens, substitua o valor `Bytes` pelos dados binários codificados em Base64 de cada arquivo de imagem.

   c. Para atualizar as configurações, modifique a saída do objeto `Settings` e inclua-a em sua próxima solicitação. O Amazon Cognito ignora os valores no objeto `Settings` que não estejam no esquema recebido na resposta da API.

5. Use o corpo de resposta atualizado em uma UpdateManagedLoginBrandingsolicitação CreateManagedLoginBrandingor. Se essa solicitação exceder 2 MB, separe-a em várias solicitações. Essas operações funcionam em um modelo `PATCH` no qual as configurações originais permanecem inalteradas, a menos que você especifique o contrário.

Personalizar identidade visual de IU hospedada (clássica)

Você pode usar a Console de gerenciamento da AWS, ou a API AWS CLI ou, para especificar as configurações clássicas de personalização para a interface do usuário hospedada. É possível fazer upload de uma imagem de logo personalizada para exibição no aplicativo. Também é possível aplicar algumas opções de Cascading Style Sheets (CSS) à aparência da IU.

Você pode personalizar os padrões da IU e substituir clientes da aplicação individuais por configurações específicas. O Amazon Cognito aplica a configuração padrão a cada cliente de aplicação que não tem configurações no nível do cliente.

No console do Amazon Cognito e nas solicitações de API, a solicitação que define a personalização da IU não deve exceder 135 KB. Em casos raros, a soma dos cabeçalhos da solicitação, do

arquivo CSS e do logotipo pode exceder 135 KB. O Amazon Cognito codifica o arquivo de imagem em Base64. Isso aumenta o tamanho de uma imagem de 100 KB para 130 KB, mantendo 5 KB para cabeçalhos de solicitação e o CSS. Se a solicitação for muito grande, a solicitação de `SetUICustomization` API Console de gerenciamento da AWS ou sua solicitação retornará um `request parameters too large` erro. Ajuste a imagem do logotipo para não ultrapassar 100 KB e o arquivo CSS para não passar de 3 KB. Você não pode definir o CSS e a personalização do logotipo separadamente.

> **Note**
>
> Para personalizar a interface de usuário, é necessário configurar um domínio para o grupo de usuários.

Especificar um logotipo personalizado em uma identidade visual clássica

O Amazon Cognito centraliza o logotipo personalizado acima dos campos de entrada no Endpoint de login.

Escolha um arquivo PNG, JPG ou JPEG que possa ser dimensionado para 350 por 178 pixels para o logotipo personalizado de interface de usuário hospedado. O arquivo de logotipo não pode ter mais de 100 KB de tamanho, ou 130 KB após a codificação do Amazon Cognito em Base64. Para definir um `ImageFile` SetUICustomizationin na API, converta seu arquivo em uma string de texto codificada em Base64 ou, no AWS CLI, forneça um caminho de arquivo e deixe o Amazon Cognito codificá-lo para você.

Especificar personalizações de CSS em uma identidade visual clássica

Você pode personalizar o CSS para as páginas hospedadas do aplicativo, considerando as seguintes restrições:

- Você pode usar qualquer um dos nomes de classe CSS a seguir:
  - `background-customizable`
  - `banner-customizable`
  - `errorMessage-customizable`
  - `idpButton-customizable`
  - `idpButton-customizable:hover`
  - `idpDescription-customizable`

- `inputField-customizable`
- `inputField-customizable:focus`
- `label-customizable`
- `legalText-customizable`
- `logo-customizable`
- `passwordCheck-valid-customizable`
- `passwordCheck-notValid-customizable`
- `redirect-customizable`
- `socialButton-customizable`
- `submitButton-customizable`
- `submitButton-customizable:hover`
- `textDescription-customizable`

- Os valores de propriedade podem conter HTML, exceto pelos seguintes valores: instruções `@import`, `@supports`, `@page` ou `@media` ou Javascript.

Você pode personalizar as seguintes propriedades do CSS.

Rótulos

- font-weight é um múltiplo de 100, entre 100 e 900.
- color é a cor do texto. Deve ser um valor de cor CSS válido.

Campos de entrada

- width é a largura do bloco de contenção em percentual.
- height é a altura do campo de entrada em pixels (px).
- color é a cor do texto. Ele pode ser qualquer valor de cor padrão do CSS.
- background-color é a cor do plano de fundo do campo de entrada. Ele pode ser qualquer valor de cor padrão do CSS.
- border é um valor padrão de borda do CSS que especifica a largura, a transparência e a cor da borda da janela do seu aplicativo. A largura pode apresentar qualquer valor entre 1 e 100 px. A transparência pode ser sólida ou nenhuma. A cor pode assumir qualquer valor de cor padrão.

Descrições do texto

- padding-top é a quantidade de preenchimento acima da descrição do texto.

- padding-bottom é a quantidade de preenchimento abaixo da descrição do texto.
- display pode ser `block` ou `inline`.
- font-size é o tamanho da fonte para as descrições do texto.
- color é a cor do texto. Deve ser um valor de cor CSS válido.

Botão de envio

- font-size é o tamanho da fonte para o texto do botão.
- font-weight é a densidade da fonte para o texto do botão: `bold`, `italic` ou `normal`.
- margin é uma string de quatro valores que indica o tamanho das margens superior, inferior, direita e esquerda para o botão.
- font-size é o tamanho da fonte para as descrições do texto.
- width é a largura do texto do botão em porcentagem do bloco.
- height é a altura do botão em pixels (px).
- color é a cor do texto do botão. Ele pode ser qualquer valor de cor padrão do CSS.
- background-color é a cor do plano de fundo do botão. Ele pode ser qualquer valor de cor padrão.

Banner

- padding é uma string de quatro valores que indica o tamanho dos preenchimentos superior, inferior, direito e esquerdo para o banner.
- background-color é a cor do plano de fundo do banner. Ele pode ser qualquer valor de cor padrão do CSS.

Sobreposição do botão de envio

- color é a cor de primeiro plano do botão ao passar por cima dele. Ele pode ser qualquer valor de cor padrão do CSS.
- background-color é a cor do plano de fundo do botão ao passar por cima dele. Ele pode ser qualquer valor de cor padrão do CSS.

Sobreposição do botão do provedor de identidade

- color é a cor de primeiro plano do botão ao passar por cima dele. Ele pode ser qualquer valor de cor padrão do CSS.
- background-color é a cor do plano de fundo do botão ao passar por cima dele. Ele pode ser qualquer valor de cor padrão do CSS.

Verificação de senha não válida

- color é a cor do texto da mensagem `"Password check not valid"`. Ele pode ser qualquer valor de cor padrão do CSS.

Contexto

- background-color é a cor do plano de fundo da janela do aplicativo. Ele pode ser qualquer valor de cor padrão do CSS.

Mensagens de erro

- margin é uma string de quatro valores que indica o tamanho das margens superior, inferior, direita e esquerda.
- padding é o tamanho do preenchimento.
- font-size é o tamanho da fonte.
- width é a largura da mensagem de erro como uma porcentagem do bloco.
- background é a cor do plano de fundo da mensagem de erro. Ele pode ser qualquer valor de cor padrão do CSS.
- border é uma string de três valores que especifica a largura, a transparência e a cor da borda.
- color é a cor do texto da mensagem de erro. Ele pode ser qualquer valor de cor padrão do CSS.
- box-sizing é usado para indicar ao navegador o que as propriedades de dimensionamento (largura e altura) devem incluir.

Botões do provedor de identidade

- height é a altura do botão em pixels (px).
- width é a largura do texto do botão como porcentagem do bloco.
- text-align é a configuração de alinhamento do texto. Ela pode ser: `left`, `right` ou `center`.
- margin-bottom é a configuração da margem inferior.
- color é a cor do texto do botão. Ele pode ser qualquer valor de cor padrão do CSS.
- background-color é a cor do plano de fundo do botão. Ele pode ser qualquer valor de cor padrão do CSS.
- border-color é a cor da borda do botão. Ele pode ser qualquer valor de cor padrão do CSS.

Descrições do provedor de identidade

- padding-top é a quantidade de preenchimento acima da descrição.
- padding-bottom é a quantidade de preenchimento abaixo da descrição.

- display pode ser `block` ou `inline`.
- font-size é o tamanho da fonte para as descrições.
- color é a cor do texto para os cabeçalhos das seções do IdP, por exemplo, Fazer login com seu ID corporativo. Deve ser um valor de cor CSS válido.

Texto legal

- color é a cor do texto. Ele pode ser qualquer valor de cor padrão do CSS.
- font-size é o tamanho da fonte.

> **Note**
>
> Quando você personaliza Legal text (Texto legal), você está personalizando a mensagem We won't post to any of your accounts without asking first (Não publicaremos em nenhuma de suas contas sem pedir permissão antes) que é exibida na página de acesso em provedores de identidade sociais.

Logo

- max-width é a largura máxima como porcentagem do bloco.
- max-height é a altura máxima como porcentagem do bloco.
- background-color é a cor do plano de fundo para logs com seções transparentes. Deve ser um valor de cor CSS válido.

Foco do campo de entrada

- border-color é a cor do campo de entrada. Ele pode ser qualquer valor de cor padrão do CSS.
- outline é a largura da borda do campo de entrada, em pixels.

Botão social

- height é a altura do botão em pixels (px).
- text-align é a configuração de alinhamento do texto. Ela pode ser: `left`, `right` ou `center`.
- width é a largura do texto do botão como porcentagem do bloco.
- margin-bottom é a configuração da margem inferior.

Verificação de senha válida

- color é a cor do texto da mensagem `"Password check valid"`. Ele pode ser qualquer valor de cor padrão do CSS.

Personalizando a interface de usuário hospedada com a marca clássica no Console de gerenciamento da AWS

Você pode usar o Console de gerenciamento da AWS para especificar as configurações de personalização da interface do usuário para seu aplicativo.

> **Note**
>
> Você pode visualizar a interface do usuário hospedada com as personalizações construindo o URL a seguir, com as especificações para o seu grupo de usuários, e digitando-o em um navegador: `https://<your_domain>/login?response_type=code&client_id=<your_app_client_id>&redirect_uri=<your_callbac`
> É provável que seja necessário esperar em torno de um minuto para atualizar o navegador antes que as alterações feitas no console apareçam.
> Seu domínio é exibido na guia App integration (Integração da aplicação) em Domain (Domínio). O ID de cliente da aplicação e o URL de retorno de chamada são exibidos em App client (Cliente da aplicação).

Para especificar as configurações de personalização de interface do usuário do aplicativo

1. Faça login no console do Amazon Cognito.
2. No painel de navegação, escolha User Pools (Grupos de usuários) e escolha o grupo de usuários que deseja editar.
3. Crie um domínio na guia Domínio ou atualize seu domínio existente. Na Versão de marca, defina seu domínio para usar a IU hospedada (clássica).
4. Clique no menu Login gerenciado.
5. Para personalizar as configurações da IU para todos os clientes da aplicação, localize Estilo em Configurações da interface de usuário hospedada e selecione Editar.
6. Para personalizar as configurações da IU para um único cliente de aplicação, acesse o menu Clientes da aplicação e selecione o cliente de aplicação que deseja modificar. Depois, localize Estilo de interface de usuário hospedada (clássico) e selecione Substituir. Selecione Editar.
7. Para carregar seu próprio arquivo de imagem de logo, escolha Choose file (Escolher arquivo) ou Replace current file (Substituir arquivo atual).
8. Para personalizar o CSS da interface do usuário hospedada, baixe CSS template.css e modifique o modelo com os valores que deseja personalizar. Somente as chaves incluídas no

modelo podem ser usadas com a interface do usuário hospedada. As chaves CSS adicionadas não serão refletidas na interface do usuário. Após personalizar o arquivo CSS, escolha Choose file (Escolher arquivo) ou Replace current file (Substituir arquivo atual) para carregar seu arquivo CSS personalizado.

Personalizando a interface de usuário hospedada com a marca clássica na API de grupos de usuários e com o AWS CLI

Use os comandos a seguir para especificar as configurações de personalização da interface do usuário para o seu grupo de usuários.

Para obter as configurações de personalização da interface do usuário para uma interface do usuário de aplicação integrada do grupo de usuários, use as operações de API a seguir.

- AWS CLI: `aws cognito-idp get-ui-customization`
- AWS API: GetUICustomization

Para definir as configurações de personalização da interface do usuário para uma interface do usuário de aplicação integrada do grupo de usuários, use as operações de API a seguir.

- AWS CLI do arquivo de imagem: `aws cognito-idp set-ui-customization --user-pool-id <your-user-pool-id> --client-id <your-app-client-id> --image-file fileb://"<path-to-logo-image-file>" --css ".label-customizable{ color: <color>;}"`
- AWS CLI com imagem codificada como texto binário Base64: `aws cognito-idp set-ui-customization --user-pool-id <your-user-pool-id> --client-id <your-app-client-id> --image-file <base64-encoded-image-file> --css ".label-customizable{ color: <color>;}"`
- AWS API: SetUICustomization

Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda

O Amazon Cognito trabalha com AWS Lambda funções para modificar o comportamento de autenticação do seu grupo de usuários. É possível configurar o grupo de usuários para invocar automaticamente as funções do Lambda antes da primeira inscrição, após a conclusão da

autenticação e em vários estágios intermediários. Suas funções podem modificar o comportamento padrão do seu fluxo de autenticação, fazer solicitações de API para modificar seu grupo de usuários ou outros AWS recursos e se comunicar com sistemas externos. O código nas funções do Lambda é seu. O Amazon Cognito envia dados de eventos para a função, espera que a função processe os dados e, na maioria dos casos, antecipa um evento de resposta que reflete as alterações que você deseja fazer na sessão.

No sistema de eventos de solicitação e resposta, você pode apresentar seus próprios desafios de autenticação, migrar usuários entre seu grupo de usuários e outro repositório de identidades, personalizar mensagens e modificar tokens web JSON ()JWTs.

Os gatilhos do Lambda podem personalizar a resposta que o Amazon Cognito fornece ao usuário depois que ele inicia uma ação em seu grupo de usuários. Por exemplo, é possível impedir o login de um usuário que, de outra forma, seria bem-sucedido. Eles também podem realizar operações de tempo de execução em seu AWS ambiente externo APIs, bancos de dados ou repositórios de identidade. O gatilho de migração de usuário, por exemplo, pode combinar uma ação externa com uma alteração no Amazon Cognito: você pode pesquisar informações do usuário em um diretório externo e definir atributos em um novo usuário com base nessas informações externas.

Quando você tem um gatilho do Lambda atribuído ao seu grupo de usuários, o Amazon Cognito interrompe seu fluxo padrão para solicitar informações de sua função. O Amazon Cognito gera um evento JSON e o transmite para sua função. O evento contém informações sobre a solicitação do usuário para criar uma conta de usuário, fazer login, redefinir uma senha ou atualizar um atributo. Sua função então tem a oportunidade de agir ou enviar o evento de volta sem modificações. Um evento retornado sem modificação notifica seu grupo de usuários para continuar com a ação padrão do evento. Por exemplo, seu acionador de pré-inscrição pode confirmar automaticamente os usuários para a origem do acionador `PreSignUp_SignUp`, mas retornar o evento inalterado para usuários externos e criados pelo administrador.

A tabela a seguir resume algumas das maneiras pelas quais os acionadores do Lambda são usados para personalizar operações do grupo de usuários:

Fluxo de grupo de usuários	Operation	Description
Fluxo de autenticação personalizado	Definir o desafio de autentica ção	Determina o próximo desafio em um fluxo de autenticação personalizado

Fluxo de grupo de usuários	Operation	Description
	Criar desafio de autenticação	Cria um desafio em um fluxo de autenticação personalizado
	Verificar a resposta do desafio de autenticação	Determina se uma resposta está correta em um fluxo de autenticação personalizado
Eventos de autenticação	the section called “Pré-autenticação”	Validação personalizada para aceitar ou negar a solicitação de login
	the section called “Pós-autenticação”	Registra eventos para análise personalizada
	the section called “Pré-geração de tokens”	Aumenta ou suprime solicitações de token
Federação	the section called “Federação receptiva”	Transforma os atributos do usuário federado antes da criação ou atualização do usuário nos grupos de usuários do Amazon Cognito
Cadastrar-se	the section called “Pré-cadastro”	Executa uma validação personalizada que aceita ou nega a solicitação de cadastro
	the section called “Publicar confirmação”	Adiciona mensagens de boas-vindas personalizadas ou registro de eventos para análise personalizada
	the section called “Migrar usuário”	Migra um usuário de um diretório de usuário existente para grupos de usuários

Fluxo de grupo de usuários	Operation	Description
Mensagens	the section called “Mensagem personalizada”	Realiza personalização avançada e localização de mensagens
Criação de token	the section called “Pré-geração de tokens”	Adiciona ou remove atributos em tokens de ID e acesso
Provedores de terceiros de e-mail e SMS	the section called “Remetentes personalizados”	Usa um provedor de terceiros para enviar mensagens de e-mail e SMS

Tópicos

- O que é importante saber sobre acionadores do Lambda
- Adicionar um acionador do Lambda do grupo de usuários
- Evento de acionador do Lambda do grupo de usuários
- Parâmetros comuns do acionador do Lambda do grupo de usuários
- Metadados do cliente
- Conectar operações de API a gatilhos do Lambda
- Conectar gatilhos do Lambda às operações funcionais do grupo de usuários
- Acionador do Lambda de pré-cadastro
- Acionador do Lambda de pós-confirmação
- Acionador do Lambda de pré-autenticação
- Acionador do Lambda de pós-autenticação
- Acionador Lambda de federação de entrada
- Acionadores do Lambda de desafio personalizado de autenticação
- Acionador do Lambda antes da geração do token
- Migrar o acionador do Lambda do usuário
- Acionador do Lambda de mensagem personalizada
- Acionadores do Lambda remetente personalizado

O que é importante saber sobre acionadores do Lambda

Ao preparar os grupos de usuários para as funções do Lambda, considere o seguinte:

- Os eventos que o Amazon Cognito envia aos gatilhos do Lambda podem mudar com novos atributos. As posições dos elementos de resposta e solicitação na hierarquia JSON podem mudar ou os nomes dos elementos podem ser adicionados. Na função do Lambda, é possível esperar receber os pares de chave-valor do elemento de entrada descritos neste guia, mas uma validação de entrada mais rigorosa pode fazer com que as funções falhem.
- É possível selecionar uma das várias versões dos eventos que o Amazon Cognito envia a alguns gatilhos. Algumas versões podem exigir que você aceite uma alteração nos preços do Amazon Cognito. Para obter mais informações sobre a definição de preços, consulte Preço do Amazon Cognito. Para personalizar os tokens de acesso em um Acionador do Lambda antes da geração do token, é necessário configurar o grupo de usuários com um plano de recursos que não seja o Lite e atualizar a configuração do acionador do Lambda para usar a versão 2 do evento.
- Exceto Acionadores do Lambda remetente personalizado, o Amazon Cognito invoca funções do Lambda de forma síncrona. Quando o Amazon Cognito chama sua função do Lambda, ela deve responder em até cinco segundos. Se isso não acontecer e se a chamada puder ser repetida, o Amazon Cognito poderá repetir a chamada. Se todas as tentativas de repetição falharem, a função expirará. Não é possível alterar esse valor de tempo limite de cinco segundos. Para obter mais informações, consulte o modelo de programação Lambda no Guia do AWS Lambda desenvolvedor.

  O Amazon Cognito não repete chamadas de função que retornam um erro de invocação com um código de status HTTP de 500 a 599. Esses códigos indicam um problema de configuração que faz com que o Lambda não consiga iniciar a função. Para obter mais informações, consulte Tratamento de erros e novas tentativas automáticas em AWS Lambda.
- Você não pode declarar uma versão da função na configuração do acionador do Lambda. Os grupos de usuários do Amazon Cognito invocam a versão mais recente da função por padrão. No entanto, você pode associar uma versão da função a um alias e definir seu gatilho `LambdaArn` para o alias ARN em uma solicitação de API ou de uma CreateUserPoolsolicitação de API. UpdateUserPool Essa opção não está disponível no Console de gerenciamento da AWS. Para obter mais informações sobre aliases, consulte Aliases de função do Lambda no Guia do desenvolvedor do AWS Lambda .
- Se você excluir um acionador do Lambda, deverá atualizar o acionador correspondente no grupo de usuários. Por exemplo, se excluir o acionador pós-autenticação, você deverá definir o

acionador Post authentication (Pós-autenticação) no grupo de usuários correspondente como none (nenhum).

- Se a função do Lambda não retornar os parâmetros de solicitação e resposta para o Amazon Cognito ou retornar um erro, o evento de autenticação não será bem-sucedido. Você pode retornar um erro na função para impedir a inscrição, a autenticação, a geração de tokens ou qualquer outro estágio do fluxo de autenticação de um usuário que invoque o acionador do Lambda.

  O login gerenciado retorna erros que os acionadores do Lambda geram como texto de erro acima da solicitação de login. A API de grupos de usuários do Amazon Cognito retorna erros do acionador no formato `[trigger] failed with error [error text from response]`. Como prática recomendada, gerem apenas erros nas funções do Lambda que você deseja que seus usuários vejam. Use métodos de saída, como `print()` registrar qualquer informação confidencial ou de depuração no Logs. CloudWatch Para ver um exemplo, consulte Exemplo de pré-cadastro: negar cadastro se o nome de usuário tiver menos de cinco caracteres.

- Você pode adicionar uma função Lambda em outra Conta da AWS como acionador para seu grupo de usuários. Você deve adicionar gatilhos entre contas com as operações da UpdateUserPoolAPI CreateUserPoole, ou seus equivalentes em e a. CloudFormation AWS CLI Você não pode adicionar funções de várias contas no Console de gerenciamento da AWS.

- Quando você inclui um acionador do Lambda no console do Amazon Cognito, o Amazon Cognito adiciona uma política baseada em recursos à sua função que permita que o grupo de usuários a invoque. Quando você cria um acionador do Lambda fora do console do Amazon Cognito, é necessário adicionar permissões à função do Lambda. Suas permissões adicionadas devem permitir que o Amazon Cognito invoque a função em nome do grupo de usuários. Você pode adicionar permissões do Lambda Console ou usar a operação da API AddPermissionLambda.

  Exemplo de política baseada em recursos do Lambda

  O seguinte exemplo de política baseada em recursos do Lambda concede ao Amazon Cognito uma capacidade limitada de invocar uma função do Lambda. O Amazon Cognito só pode invocar a função quando o fizer em nome do grupo de usuários na condição `aws:SourceArn` e da conta na condição `aws:SourceAccount`.

  JSON

```
{
    "Version":"2012-10-17",
    "Id": "default",
    "Statement": [
```

```
        {
            "Sid": "LambdaCognitoIdpTrust",
            "Effect": "Allow",
            "Principal": {
                "Service": "cognito-idp.amazonaws.com"
            },
            "Action": "lambda:InvokeFunction",
            "Resource": "arn:aws:lambda:us-
east-1:111122223333:function:MyFunction",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "AWS:SourceArn": "arn:aws:cognito-idp:us-
east-1:111122223333:userpool/us-east-1_EXAMPLE"
                }
            }
        }
    ]
}
```

Adicionar um acionador do Lambda do grupo de usuários

Para adicionar um acionador do Lambda do grupo de usuários com o console

1. Use o console do Lambda para criar uma função do Lambda. Para obter mais informações, sobre funções Lambda, consulte o Guia do desenvolvedor do AWS Lambda.
2. Acesse o console do Amazon Cognito e escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Selecione o menu Extensões e localize os Acionadores do Lambda.
5. Selecione Add a Lambda trigger (Adicionar um acionador do Lambda).
6. Selecione uma Category (Categoria) de acionador do Lambda com base no estágio de autenticação que deseja personalizar.
7. Selecione Atribuir função Lambda e selecione uma função no mesmo grupo Região da AWS de usuários.

> **Note**
>
> Se suas credenciais AWS Identity and Access Management (IAM) tiverem permissão para atualizar a função Lambda, o Amazon Cognito adicionará uma política baseada em recursos do Lambda. Com essa política, o Amazon Cognito pode invocar a função selecionada. Se as credenciais conectadas não tiverem permissões suficientes do IAM, você deverá atualizar a política baseada em recursos separadamente. Para obter mais informações, consulte the section called “Coisas a saber”.

8. Escolha Salvar alterações.
9. Você pode usar CloudWatch no console Lambda para registrar sua função do Lambda. Para obter mais informações, consulte Acessando CloudWatch registros para Lambda.

Evento de acionador do Lambda do grupo de usuários

O Amazon Cognito transmite informações de evento para a função do Lambda. A função do Lambda retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. Se a sua função retornar o evento de entrada sem modificação, o Amazon Cognito continuará com o comportamento padrão. Veja a seguir os parâmetros comuns a todos os eventos de entrada do acionador do Lambda. Para obter a sintaxe de eventos específica do acionador, revise o esquema de eventos na seção deste guia para cada acionador.

JSON

```
{
    "version": "string",
    "triggerSource": "string",
    "region": AWSRegion,
    "userPoolId": "string",
    "userName": "string",
    "callerContext":
        {
            "awsSdkVersion": "string",
            "clientId": "string"
        },
    "request":
        {
            "userAttributes": {
```

```
                "string": "string",
                ....
            }
        },
    "response": {}
}
```

Parâmetros comuns do acionador do Lambda do grupo de usuários

versionamento

O número da versão da função do Lambda.

triggerSource

O nome do evento que acionou a função do Lambda. Para uma descrição de cada triggerSource, consulte Conectar gatilhos do Lambda às operações funcionais do grupo de usuários.

region

O Região da AWS como uma `AWSRegion` instância.

userPoolId

O ID do grupo de usuários.

userName

O nome do usuário atual.

callerContext

Metadados sobre a solicitação e o ambiente de código. Ele contém os campos awsSdkVersione o ClientID.

awsSdkVersion

A versão do AWS SDK que gerou a solicitação.

clientId

O ID do cliente da aplicação do grupo de usuários.

request

Detalhes da solicitação de API do usuário. Ele inclui os seguintes campos e quaisquer parâmetros de solicitação específicos do gatilho. Por exemplo, um evento que o Amazon Cognito

envia a um acionador de pré-autenticação também conterá um parâmetro `userNotFound`. Você pode processar o valor desse parâmetro para realizar uma ação personalizada quando o usuário tentar fazer login com um nome de usuário não registrado.

userAttributes

Um ou mais pares de chave-valor de nomes e valores de atributos, por exemplo, `"email": "john@example.com"`.

resposta

Esse parâmetro não contém nenhuma informação na solicitação original. Sua função do Lambda deve retornar todo o evento ao Amazon Cognito e adicionar quaisquer parâmetros de retorno à `response`. Para ver quais parâmetros de retorno sua função pode incluir, consulte a documentação do gatilho que você deseja usar.

Metadados do cliente

Você pode enviar parâmetros personalizados para suas funções de acionador do Lambda em operações de API e solicitações de Endpoint de token. Com os metadados do cliente, sua aplicação pode coletar informações adicionais sobre o ambiente em que as solicitações se originam. Quando você transmite metadados do cliente para suas funções do Lambda, elas podem processar os dados adicionais e usá-los nos logs ou na personalização dos fluxos de autenticação. Os metadados do cliente são pares de strings de sua escolha e design em um formato de valor-chave JSON.

Exemplos de casos de uso de metadados do cliente

- Transmita dados de geolocalização no momento da inscrição para o acionador de pré-inscrição e evite o login de locais indesejados.
- Transmita dados de identificação do locatário para acionadores de desafios personalizados e emita desafios diferentes para clientes de diferentes unidades de negócios.
- Transmita o token de um usuário para o acionador de pré-geração de tokens e gere um log da entidade principal em nome da qual foi feita uma solicitação M2M. Para ver um exemplo de solicitação, consulte Credenciais do cliente com autorização básica.

Este é um exemplo de transmissão de metadados do cliente para o acionador de pré-inscrição.

SignUp request

Veja a seguir um exemplo de SignUpsolicitação com metadados do cliente que o Amazon Cognito passa para um gatilho de pré-inscrição.

```
POST HTTP/1.1
Host: cognito-idp.us-east-1.amazonaws.com
X-Amz-Date: 20230613T200059Z
Accept-Encoding: gzip, deflate, br
X-Amz-Target: AWSCognitoIdentityProviderService.SignUp
User-Agent: <UserAgentString>
Authorization: AWS4-HMAC-SHA256 Credential=<Credential>, SignedHeaders=<Headers>,
 Signature=<Signature>
Content-Length: <PayloadSizeBytes>

{
    "ClientId": "1example23456789",
    "Username": "mary_major",
    "Password": "<Password>",
    "SecretHash": "<Secret hash>",
    "ClientMetadata": {
        "IpAddress" : "192.0.2.252",
        "GeoLocation" : "Netherlands (Kingdom of the) [NL]"
    }
    "UserAttributes": [
        {
            "Name": "name",
            "Value": "Mary"
        },
        {
            "Name": "email",
            "Value": "mary_major@example.com"
        },
        {
            "Name": "phone_number",
            "Value": "+12065551212"
        }
    ],
}
```

Lambda trigger input event

A solicitação resulta no corpo de solicitação a seguir para sua função de pré-cadastro.

```
{
    "callerContext": {
        "awsSdkVersion": "aws-sdk-unknown-unknown",
        "clientId": "1example23456789"
    },
    "region": "us-west-2",
    "request": {
        "clientMetadata": {
            "GeoLocation": "Netherlands (Kingdom of the) [NL]",
            "IpAddress": "192.0.2.252"
        },
        "userAttributes": {
            "email": "mary_major@example.com",
            "name": "Mary",
            "phone_number": "+12065551212"
        },
        "validationData": null
    },
    "response": {
        "autoConfirmUser": false,
        "autoVerifyEmail": false,
        "autoVerifyPhone": false
    },
    "triggerSource": "PreSignUp_SignUp",
    "userName": "mary_major2",
    "userPoolId": "us-west-2_EXAMPLE",
    "version": "1"
}
```

Metadados do cliente para credenciais do cliente machine-to-machine (M2M)

Você pode transmitir metadados do cliente em solicitações de M2M. Os metadados do cliente são informações adicionais de um usuário ou ambiente de aplicação que podem contribuir para os resultados de um Acionador do Lambda antes da geração do token. Nas operações de autenticação com um usuário principal, você pode passar os metadados do cliente para o gatilho de pré-geração do token no corpo das solicitações AdminRespondToAuthChallengede RespondToAuthChallengeAPI. Como as aplicações conduzem o fluxo de geração de tokens de acesso para M2M com solicitações diretas ao Endpoint de token, elas têm um modelo diferente. No corpo POST das solicitações de token para credenciais do cliente, transmita um parâmetro `aws_client_metadata` com o objeto de metadados do cliente codificado em URL (`x-www-form-`

`urlencoded`) para string. Para ver um exemplo de solicitação, consulte Credenciais do cliente com autorização básica. Veja a seguir um exemplo de parâmetro que transmite os pares de chave-valor `{"environment": "dev", "language": "en-US"}`.

```
aws_client_metadata=%7B%22environment%22%3A%20%22dev%22,%20%22language%22%3A%20%22en-US%22%7D
```

Atributos temporários do usuário: **`validationData`**

Algumas operações de autenticação também têm um parâmetro `validationData`. Assim como os metadados do cliente, essa é uma oportunidade de transmitir informações externas que o Amazon Cognito não coleta automaticamente para os acionadores do Lambda. O campo de dados de validação tem como objetivo fornecer à sua função Lambda um contexto de usuário adicional nas operações de inscrição e login. SignUpe AdminCreateUserpasse `validationData` para o gatilho de pré-inscrição. InitiateAuthe AdminInitiateAuthpasse `ClientMetadata` o corpo da solicitação da API como `validationData` no evento de entrada para os gatilhos do usuário de pré-autenticação e migração.

Para mapear as operações da API para as funções para as quais elas podem transmitir metadados do cliente, consulte as seções de origem do acionador a seguir.

Conectar operações de API a gatilhos do Lambda

As seções a seguir descrevem os gatilhos do Lambda que o Amazon Cognito invoca a partir da atividade em seu grupo de usuários.

Quando a aplicação conecta usuários pela API de grupos de usuários do Amazon Cognito, do login gerenciado ou de endpoints de grupo de usuários, o Amazon Cognito invoca suas funções do Lambda com base no contexto da sessão. Para ter mais informações sobre a API de grupos de usuários do Amazon Cognito e endpoints de grupo de usuários, consulte Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado. As tabelas nas seções a seguir descrevem eventos que fazem com que o Amazon Cognito invoque uma função e a string `triggerSource` que o Amazon Cognito inclui na solicitação.

Tópicos

- Gatilhos do Lambda na API do Amazon Cognito
- Acionadores do Lambda para usuários locais do Amazon Cognito no login gerenciado
- Acionadores do Lambda para usuários federados

Gatilhos do Lambda na API do Amazon Cognito

A tabela a seguir descreve as strings de origem dos acionadores do Lambda que o Amazon Cognito pode invocar quando a aplicação cria, faz login ou atualiza um usuário local.

Fontes locais de gatilho de usuário na API do Amazon Cognito

Operação de API	Gatilho do Lambda	Fonte de gatilhos
AdminCreateUser	Pré-cadastro	`PreSignUp_AdminCreateUser`
	Pré-geração de tokens	`TokenGeneration_NewPasswordChallenge`
	Mensagem personalizada	`CustomMessage_AdminCreateUser`
	Remetente de e-mail personalizado	`CustomEmailSender_AdminCreateUser`
	Remetente de SMS personalizado	`CustomSMSSender_AdminCreateUser`
SignUp	Pré-cadastro	`PreSignUp_SignUp`
	Mensagem personalizada	`CustomMessage_SignUp`
	Remetente de e-mail personalizado	`CustomEmailSender_SignUp`
	Remetente de SMS personalizado	`CustomSMSSender_SignUp`
ConfirmSignUp AdminConfirmSignUp	Confirmação do post	`PostConfirmation_ConfirmSignUp`
InitiateAuth AdminInitiateAuth	Pré-autenticação	`PreAuthentication_Authentication`

Operação de API	Gatilho do Lambda	Fonte de gatilhos
	Pós-autenticação	`PostAuthentication_Authentication`
	Definir o desafio de autenticação	`DefineAuthChallenge_Authentication`
	Criar desafio de autenticação	`CreateAuthChallenge_Authentication`
	Verificar desafio de autenticação	`VerifyAuthChallenge_Authentication`

	Pré-geração de tokens	`TokenGeneration_Authentication`
		`TokenGeneration_AuthenticateDevice`
		`TokenGeneration_RefreshTokens`

	Migrar usuário	`UserMigration_Authentication`
	Mensagem personalizada	`CustomMessage_Authentication`
	Remetente de e-mail personalizado	`CustomEmailSender_AccountTakeOverNotification` `CustomEmailSender_Authentication`
	Remetente de SMS personalizado	`CustomSMSSender_Authentication`

Operação de API	Gatilho do Lambda	Fonte de gatilhos
RespondToAuthChallenge AdminRespondToAuthChallenge	Pós-autenticação	`PostAuthentication_Authentication`
	Definir o desafio de autenticação	`DefineAuthChallenge_Authentication`
	Criar desafio de autenticação	`CreateAuthChallenge_Authentication`
	Verificar desafio de autenticação	`VerifyAuthChallenge_Authentication`

	Pré-geração de tokens	`TokenGeneration_Authentication`
		
		`TokenGeneration_AuthenticateDevice`
		
		`TokenGeneration_RefreshTokens`

	Mensagem personalizada	`CustomMessage_Authentication`

	Remetente de e-mail personalizado	`CustomEmailSender_AccountTakeOverNotification`
		
		`CustomEmailSender_Authentication`

	Remetente de SMS personalizado	`CustomSMSSender_Authentication`
ForgotPassword	Migrar usuário	`UserMigration_ForgotPassword`

Operação de API	Gatilho do Lambda	Fonte de gatilhos
	Mensagem personalizada	`CustomMessage_ForgotPassword`
	Remetente de e-mail personalizado	`CustomEmailSender_ForgotPassword`
	Remetente de SMS personalizado	`CustomSMSSender_ForgotPassword`
ConfirmForgotPassword	Confirmação do post	`PostConfirmation_ConfirmForgotPassword`
UpdateUserAttributes AdminUpdateUserAttributes	Mensagem personalizada	`CustomMessage_UpdateUserAttribute`
	Remetente de e-mail personalizado	`CustomEmailSender_UpdateUserAttribute`
	Remetente de SMS personalizado	`CustomSMSSender_UpdateUserAttribute`
VerifyUserAttributes	Mensagem personalizada	`CustomMessage_VerifyUserAttribute`
	Remetente de e-mail personalizado	`CustomEmailSender_VerifyUserAttribute`
	Remetente de SMS personalizado	`CustomSMSSender_VerifyUserAttribute`
GetTokensFromRefreshToken	Pré-geração de tokens	`TokenGeneration_Authentication`

Acionadores do Lambda para usuários locais do Amazon Cognito no login gerenciado

A tabela a seguir descreve as strings de origem dos acionadores do Lambda que o Amazon Cognito pode invocar quando um usuário local faz login em seu grupo de usuários com o login gerenciado.

Fontes locais de acionador de usuário no login gerenciado

URI de login gerenciado	Gatilho do Lambda	Fonte de gatilhos
`/signup`	Pré-cadastro	`PreSignUp_SignUp`
	Mensagem personalizada	`CustomMessage_SignUp`
	Remetente de e-mail personalizado	`CustomEmailSender_SignUp`
	Remetente de SMS personalizado	`CustomSMSSender_SignUp`
`/confirmuser`	Confirmação do post	`PostConfirmation_ConfirmSignUp`
`/login`	Pré-autenticação	`PreAuthentication_Authentication`

	Pré-geração de tokens	`TokenGeneration_Authentication`
		`TokenGeneration_AuthenticateDevice`
		`TokenGeneration_RefreshTokens`

	Migrar usuário	`UserMigration_Authentication`
	Mensagem personalizada	`CustomMessage_Authentication`

URI de login gerenciado	Gatilho do Lambda	Fonte de gatilhos
/forgotpassword	Remetente de e-mail personalizado	CustomEmailSender_AccountTakeOverNotification CustomEmailSender_Authentication
	Remetente de SMS personalizado	CustomSMSSender_Authentication
	Migrar usuário	UserMigration_ForgotPassword
	Mensagem personalizada	CustomMessage_ForgotPassword
	Remetente de e-mail personalizado	CustomEmailSender_ForgotPassword
	Remetente de SMS personalizado	CustomSMSSender_ForgotPassword
/confirmforgotpassword	Confirmação do post	PostConfirmation_ConfirmForgotPassword

Acionadores do Lambda para usuários federados

Você pode usar os seguintes acionadores do Lambda para personalizar seus fluxos de trabalho do grupo de usuários para usuários que fazem login com um provedor federado.

> **Note**
>
> Usuários federados podem usar o login gerenciado para fazer login, ou você pode gerar uma solicitação para o Autorizar endpoint que os redireciona silenciosamente para a página de

login do provedor de identidades. Não é possível fazer login de usuários federados com a API de grupos de usuários do Amazon Cognito.

Fontes de acionador de usuário federado

Evento de login	Gatilho do Lambda	Fonte de gatilhos
Primeiro login	Pré-cadastro	`PreSignUp_ExternalProvider`
	Confirmação do post	`PostConfirmation_ConfirmSignUp`
	Pré-geração de tokens	`TokenGeneration_HostedAuth`
Logins subsequentes	Pré-autenticação	`PreAuthentication_Authentication`
	Pós-autenticação	`PostAuthentication_Authentication`
	Pré-geração de tokens	`TokenGeneration_HostedAuth`

O login federado não invoca nenhum Acionadores do Lambda de desafio personalizado de autenticação, Migrar o acionador do Lambda do usuário, Acionador do Lambda de mensagem personalizada ou Acionadores do Lambda remetente personalizado no grupo de usuários.

Conectar gatilhos do Lambda às operações funcionais do grupo de usuários

Cada gatilho do Lambda tem uma função em seu grupo de usuários. Por exemplo, um gatilho pode modificar seu fluxo de inscrição ou adicionar um desafio de autenticação personalizado. O evento que o Amazon Cognito envia para uma função do Lambda pode refletir uma das várias ações que compõem essa função. Por exemplo, o Amazon Cognito invoca um gatilho de pré-inscrição quando o usuário se inscreve e quando você cria um usuário. Cada um desses casos diferentes para a mesma

função tem seu próprio valor `triggerSource`. Sua função do Lambda pode processar eventos recebidos de forma diferente com base na operação que a invocou.

O Amazon Cognito também invoca todas as funções atribuídas quando um evento corresponde a uma fonte de gatilhos. Por exemplo, quando um usuário faz login em um grupo de usuários ao qual você atribuiu gatilhos de migração de usuário e pré-autenticação, ele ativa ambos.

Acionadores de inscrição, confirmação e login (autenticação)

Trigger	Valor de triggerSource	Event
Pré-cadastro	`PreSignUp_SignUp`	Pré-cadastro.
Pré-cadastro	`PreSignUp_AdminCreateUser`	Pré-cadastro quando um administrador cria um novo usuário.
Pré-cadastro	`PreSignUp_ExternalProvider`	Pré-cadastro para provedores de identidade externos.
Confirmação do post	`PostConfirmation_ConfirmSignUp`	Confirmação pós-cadastro.
Confirmação do post	`PostConfirmation_ConfirmForgotPassword`	Confirmação após esquecimento de senha.
Pré-autenticação	`PreAuthentication_Authentication`	Pré-autenticação.
Pós-autenticação	`PostAuthentication_Authentication`	Pós-autenticação.

Acionadores de desafio de autenticação personalizado

Trigger	Valor de triggerSource	Event
Definir o desafio de autenticação	`DefineAuthChallenge_Authentication`	Definir o desafio de autenticação.

Trigger	Valor de triggerSource	Event
Criar desafio de autenticação	`CreateAuthChallenge_Authentication`	Criar desafio de autenticação.
Verificar desafio de autenticação	`VerifyAuthChallengeResponse_Authentication`	Verificar a resposta do desafio de autenticação.

Acionadores da federação

Trigger	Valor de triggerSource	Event
Federação receptiva	`InboundFederation_ExternalProvider`	Federação receptiva.

Acionadores de geração de pré-token

Trigger	Valor de triggerSource	Event
Pré-geração de tokens	`TokenGeneration_HostedAuth`	O Amazon Cognito autentica o usuário na página de login do login gerenciado.
Pré-geração de tokens	`TokenGeneration_Authentication`	Autenticação do usuário ou atualização do token concluída.
Pré-geração de tokens	`TokenGeneration_NewPasswordChallenge`	O administrador cria o usuário. O Amazon Cognito invoca isso quando o usuário precisa alterar uma senha temporária.
Pré-geração de tokens	`TokenGeneration_AuthenticateDevice`	Final da autenticação do dispositivo de um usuário.
Pré-geração de tokens	`TokenGeneration_RefreshTokens`	O usuário tenta atualizar a identidade e acessar tokens.

Acionadores de migração do usuário

Trigger	Valor de triggerSource	Event
Migração do usuário	`UserMigration_Authentication`	Migração de usuários no momento de fazer login.
Migração do usuário	`UserMigration_ForgotPassword`	Migração de usuários durante o fluxo de esquecimento de senha.

Acionadores de mensagem personalizada

Trigger	Valor de triggerSource	Event
Mensagem personalizada	`CustomMessage_SignUp`	Mensagem personalizada quando um usuário se cadastra no grupo de usuários.
Mensagem personalizada	`CustomMessage_AdminCreateUser`	Mensagem personalizada quando você cria um usuário como administrador e o Amazon Cognito envia uma senha temporária.
Mensagem personalizada	`CustomMessage_ResendCode`	Mensagem personalizada quando o usuário existente solicita um novo código de confirmação.
Mensagem personalizada	`CustomMessage_ForgotPassword`	Mensagem personalizada quando o usuário solicita uma redefinição de senha.
Mensagem personalizada	`CustomMessage_UpdateUserAttribute`	Mensagem personalizada quando um usuário altera o endereço de e-mail ou número de telefone e o Amazon

Trigger	Valor de triggerSource	Event
		Cognito envia]um código de verificação.
Mensagem personalizada	`CustomMessage_VerifyUserAttribute`	Mensagem personalizada quando um usuário adiciona um endereço de e-mail ou um número de telefone e o Amazon Cognito envia um código de verificação.
Mensagem personalizada	`CustomMessage_Authentication`	Mensagem personalizada quando um usuário que configurou a MFA SMS faz login.

Acionadores de remetente personalizados

Trigger	Valor de triggerSource	Event

Remetente personalizado	`CustomEmailSender_SignUp`
	
	`CustomSmsSender_SignUp`
		Quando um usuário se cadastra no grupo de usuários.

Remetente personalizado	`CustomEmailSender_AdminCreateUser`
	
	`CustomSmsSender_AdminCreateUser`
		Quando você cria um usuário como administrador e o Amazon Cognito envia uma senha temporária.

Remetente personalizado	`CustomEmailSender_ForgotPassword`
	
	`CustomSmsSender_ForgotPassword`
		Quando o usuário solicita uma redefinição de senha.

Trigger	Valor de triggerSource	Event

Remetente personalizado	`CustomEmailSender_UpdateUserAttribute`
	
	`CustomSmsSender_UpdateUserAttribute`
		Quando um usuário altera o endereço de e-mail ou número de telefone e o Amazon Cognito envia um código de verificação.

Remetente personalizado	`CustomEmailSender_VerifyUserAttribute`
	
	`CustomSmsSender_VerifyUserAttribute`
		Quando um usuário adiciona um endereço de e-mail ou um número de telefone e o Amazon Cognito envia um código de verificação.

Remetente personalizado	`CustomEmailSender_Authentication`
	
	`CustomSmsSender_Authentication`
		Quando um usuário que configurou MFA do SMS ou e-mail, ou OTP faz login.

Remetente personalizado	`CustomEmailSender_AccountTakeOverNotification`	Quando suas configurações de proteção contra ameaças realizam uma ação automática contra a tentativa de login de um usuário e a ação para o nível de risco inclui uma notificação.

Acionador do Lambda de pré-cadastro

Você pode personalizar o processo de cadastro em grupos de usuários que têm opções por autoatendimento. Alguns usos comuns do acionador pre sign-up são: realizar análises e registros personalizados de novos usuários, aplicar padrões de segurança e governança ou vincular usuários de um IdP de terceiros a um perfil de usuário consolidado. Você também pode ter usuários confiáveis que não precisam passar por verificação e confirmação.

Imediatamente antes de o Amazon Cognito concluir a criação de um novo usuário local ou federado, ele ativa a função do Lambda de pré-cadastro. O objeto `userAttributes` de solicitação enviado

para essa função contém atributos fornecidos pelo cadastro do usuário local ou que foram mapeados com sucesso com base nos atributos do provedor para um usuário federado. Seu grupo de usuários invoca esse gatilho na inscrição por autoatendimento SignUpou no primeiro login com um provedor de identidade confiável e na criação de usuários com. AdminCreateUser Como parte do processo de cadastro, você pode usar essa função para analisar o evento de login com a lógica personalizada e modificar ou negar o novo usuário.

Tópicos

- Parâmetros do acionador do Lambda de pré-cadastro
- Exemplo de pré-cadastro: confirmação automática de usuários em um domínio registrado
- Exemplo de pré-cadastro: confirmação e verificação automáticas de todos os usuários
- Exemplo de pré-cadastro: negar cadastro se o nome de usuário tiver menos de cinco caracteres

Parâmetros do acionador do Lambda de pré-cadastro

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
            . . .
        },
        "validationData": {
            "string": "string",
            . . .
         },
        "clientMetadata": {
            "string": "string",
            . . .
         }
    },

    "response": {
        "autoConfirmUser": "boolean",
```

```
        "autoVerifyPhone": "boolean",
        "autoVerifyEmail": "boolean"
    }
}
```

Parâmetros de solicitação de pré-cadastro

userAttributes

Um ou mais pares de nome-valor que representam atributos de usuário. Os nomes de atributo são as chaves.

validationData

Um ou mais pares de chave-valor com dados de atributos do usuário que a aplicação passou para o Amazon Cognito na solicitação para criar um usuário. Envie essas informações para sua função Lambda no ValidationData parâmetro da sua solicitação AdminCreateUserou da SignUpAPI.

O Amazon Cognito não define seus ValidationData dados como atributos do usuário que você cria. ValidationData são informações temporárias do usuário que você fornece para fins de seu gatilho Lambda de pré-inscrição.

clientMetadata

Um ou mais pares de chave-valor que você pode fornecer como entrada personalizada para a função Lambda especificada para o acionamento de pré-cadastro. Você pode passar esses dados para sua função Lambda usando o ClientMetadata parâmetro nas seguintes ações de API: AdminCreateUser, AdminRespondToAuthChallengeForgotPassword, e. SignUp

Parâmetros de resposta de pré-cadastro

Na resposta, você pode definir `autoConfirmUser` para `true` se quiser confirmar o usuário automaticamente. Você pode definir `autoVerifyEmail` como `true` para verificar o e-mail do usuário automaticamente. Você pode definir `autoVerifyPhone` como `true` para verificar automaticamente o número de telefone do usuário.

> **Note**
>
> Os parâmetros de resposta `autoVerifyPhone`, `autoVerifyEmail` e `autoConfirmUser` são ignorados pelo Amazon Cognito quando a função do Lambda de pré-inscrição é acionada pela API `AdminCreateUser`.

autoConfirmUser

Definido como `true` para confirmar o usuário automaticamente; do contrário, defina-o como `false`.

autoVerifyEmail

Defina como `true` para especificar como verificado o e-mail de um usuário que está se cadastrando ou, do contrário, defina como `false`. Se `autoVerifyEmail` for definido como `true`, o atributo `email` deverá ter um valor válido, não nulo. Caso contrário, ocorrerá um erro e o usuário não poderá concluir o cadastro.

Se o atributo `email` for selecionado como um alias, será criado um alias para o e-mail do usuário quando `autoVerifyEmail` for definido. Se já houver um alias com esse endereço de e-mail, ele será movido para o novo usuário e o endereço de e-mail do usuário anterior será marcado como não verificado. Para obter mais informações, consulte Personalização dos atributos de login.

autoVerifyPhone

Defina como `true` para definir como verificado o número de telefone de um usuário que está se cadastrando; do contrário, defina-o como `false`. Se `autoVerifyPhone` for definido como `true`, o atributo `phone_number` deverá ter um valor válido, não nulo. Caso contrário, ocorrerá um erro e o usuário não poderá concluir o cadastro.

Se o atributo `phone_number` for selecionado como um alias, este será criado para o número de telefone do usuário quando `autoVerifyPhone` for definido. Se um alias com esse número de telefone já existir, o alias será movido para o novo usuário e o número de telefone do usuário anterior será marcado como não verificado. Para obter mais informações, consulte Personalização dos atributos de login.

Exemplo de pré-cadastro: confirmação automática de usuários em um domínio registrado

Este é um exemplo de código de acionador do Lambda. O acionador de pré-inscrição é invocado imediatamente antes que o Amazon Cognito processe a solicitação de inscrição. Ele usa um atributo personalizado custom:domain para confirmar automaticamente novos usuários de um determinado domínio de e-mail. Os novos usuários que não estiverem no domínio personalizado serão adicionados ao seu grupo de usuários, mas não automaticamente confirmados.

Node.js

```
export const handler = async (event, context, callback) => {
  // Set the user pool autoConfirmUser flag after validating the email domain
  event.response.autoConfirmUser = false;

  // Split the email address so we can compare domains
  var address = event.request.userAttributes.email.split("@");

  // This example uses a custom attribute "custom:domain"
  if (event.request.userAttributes.hasOwnProperty("custom:domain")) {
    if (event.request.userAttributes["custom:domain"] === address[1]) {
      event.response.autoConfirmUser = true;
    }
  }

  // Return to Amazon Cognito
  callback(null, event);
};
```

Python

```
def lambda_handler(event, context):
    # It sets the user pool autoConfirmUser flag after validating the email domain
    event['response']['autoConfirmUser'] = False

    # Split the email address so we can compare domains
    address = event['request']['userAttributes']['email'].split('@')

    # This example uses a custom attribute 'custom:domain'
    if 'custom:domain' in event['request']['userAttributes']:
        if event['request']['userAttributes']['custom:domain'] == address[1]:
            event['response']['autoConfirmUser'] = True
```

```
    # Return to Amazon Cognito
    return event
```

O Amazon Cognito transmite informações de evento para a função Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
    "request": {
        "userAttributes": {
            "email": "testuser@example.com",
            "custom:domain": "example.com"
        }
    },
    "response": {}
}
```

Exemplo de pré-cadastro: confirmação e verificação automáticas de todos os usuários

Este exemplo confirma todos os usuários e define os atributos `email` e `phone_number` do usuário como verificados, se o atributo estiver presente. Além disso, se o alias estiver habilitado, eles serão criados para `phone_number` e `email` quando a verificação automática for definida.

> **Note**
>
> Se um alias com o mesmo número de telefone já existir, o alias será movido para o novo usuário e o `phone_number` do usuário anterior será marcado como não verificado. O mesmo se aplica para endereços de e-mail. Para evitar que isso aconteça, você pode usar a ListUsers API de grupos de usuários para ver se há um usuário existente que já está usando o número de telefone ou endereço de e-mail do novo usuário como alias.

Node.js

```
exports.handler = (event, context, callback) => {
  // Confirm the user
  event.response.autoConfirmUser = true;

  // Set the email as verified if it is in the request
  if (event.request.userAttributes.hasOwnProperty("email")) {
    event.response.autoVerifyEmail = true;
  }

  // Set the phone number as verified if it is in the request
  if (event.request.userAttributes.hasOwnProperty("phone_number")) {
    event.response.autoVerifyPhone = true;
  }

  // Return to Amazon Cognito
  callback(null, event);
};
```

Python

```
def lambda_handler(event, context):
    # Confirm the user
    event['response']['autoConfirmUser'] = True

    # Set the email as verified if it is in the request
    if 'email' in event['request']['userAttributes']:
        event['response']['autoVerifyEmail'] = True

    # Set the phone number as verified if it is in the request
    if 'phone_number' in event['request']['userAttributes']:
        event['response']['autoVerifyPhone'] = True

    # Return to Amazon Cognito
    return event
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
  "request": {
    "userAttributes": {
      "email": "user@example.com",
      "phone_number": "+12065550100"
    }
  },
  "response": {}
}
```

Exemplo de pré-cadastro: negar cadastro se o nome de usuário tiver menos de cinco caracteres

Esse exemplo verifica a extensão do nome de usuário em uma solicitação de cadastro. O exemplo retornará um erro se o usuário tiver inserido um nome com menos de cinco caracteres.

Node.js

```
export const handler = (event, context, callback) => {
    // Impose a condition that the minimum length of the username is 5 is imposed on
 all user pools.
    if (event.userName.length < 5) {
        var error = new Error("Cannot register users with username less than the
 minimum length of 5");
        // Return error to Amazon Cognito
        callback(error, event);
    }
    // Return to Amazon Cognito
    callback(null, event);
};
```

Python

```
def lambda_handler(event, context):
    if len(event['userName']) < 5:
        raise Exception("Cannot register users with username less than the minimum
 length of 5")
    # Return to Amazon Cognito
    return event
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
  "userName": "rroe",
  "response": {}
}
```

Acionador do Lambda de pós-confirmação

O Amazon Cognito invoca esse acionador depois que um usuário cadastrado confirma a conta de usuário. Na função do Lambda de pós-confirmação, você pode enviar mensagens personalizadas ou adicionar solicitações de API personalizadas. Por exemplo, é possível consultar um sistema externo e preencher atributos adicionais para o usuário. O Amazon Cognito invoca esse acionador somente para os usuários que se cadastram no grupo de usuários, não para contas de usuário criadas com as credenciais de administrador.

A solicitação contém os atributos atuais do usuário confirmado. Seu grupo de usuários invoca sua função de confirmação de postagem em ConfirmSignUpAdminConfirmSignUp, e. ConfirmForgotPassword Esse acionador também é executado quando os usuários confirmam a inscrição ou a redefinição de senha no login gerenciado.

Tópicos

- Parâmetros do acionador do Lambda de pós-confirmação
- Exemplo de pós-confirmação

Parâmetros do acionador do Lambda de pós-confirmação

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "request": {
            "userAttributes": {
                "string": "string",
                . . .
            },
            "clientMetadata": {
             "string": "string",
             . . .
            }
        },
    "response": {}
}
```

Parâmetros de solicitação de pós-confirmação

userAttributes

Um ou mais pares de chave-valor que representam atributos de usuário.

clientMetadata

Um ou mais pares de chave-valor que você pode fornecer como entrada personalizada para a função Lambda especificada para o acionador de pós-confirmação. Você pode passar esses dados para sua função Lambda usando o ClientMetadata parâmetro nas seguintes ações de API: AdminConfirmSignUp, ConfirmForgotPasswordConfirmSignUp, e. SignUp

Parâmetros de resposta de pós-confirmação

Nenhuma informação de retorno adicional é esperada na resposta.

Exemplo de pós-confirmação

Esse exemplo de função Lambda envia um e-mail de confirmação para o usuário usando o Amazon SES. Para obter mais informações, consulte o Guia do desenvolvedor do Amazon Simple Storage Service.

Node.js

```
// Import required AWS SDK clients and commands for Node.js. Note that this requires
```

```
// the `@aws-sdk/client-ses` module to be either bundled with this code or included
// as a Lambda layer.
import { SES, SendEmailCommand } from "@aws-sdk/client-ses";
const ses = new SES();

const handler = async (event) => {
  if (event.request.userAttributes.email) {
    await sendTheEmail(
      event.request.userAttributes.email,
      `Congratulations ${event.userName}, you have been confirmed.`,
    );
  }
  return event;
};

const sendTheEmail = async (to, body) => {
  const eParams = {
    Destination: {
      ToAddresses: [to],
    },
    Message: {
      Body: {
        Text: {
          Data: body,
        },
      },
      Subject: {
        Data: "Cognito Identity Provider registration completed",
      },
    },
    // Replace source_email with your SES validated email address
    Source: "<source_email>",
  };
  try {
    await ses.send(new SendEmailCommand(eParams));
  } catch (err) {
    console.log(err);
  }
};

export { handler };
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
    "request": {
        "userAttributes": {
            "email": "user@example.com",
            "email_verified": true
        }
    },
    "response": {}
}
```

Acionador do Lambda de pré-autenticação

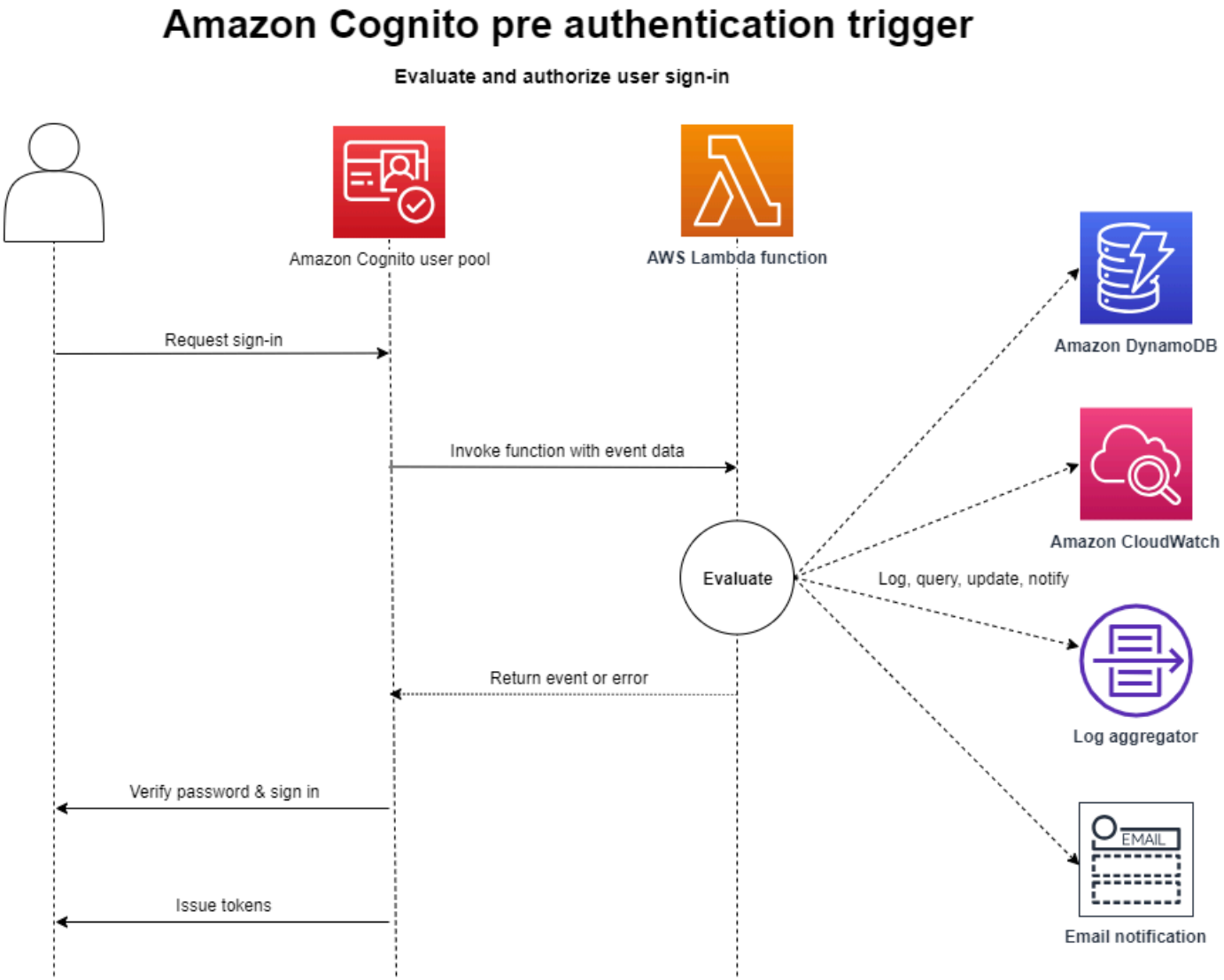
O Amazon Cognito invoca esse acionador quando um usuário tenta fazer login, de forma que você possa criar uma validação personalizada que realize ações preparatórias. Por exemplo, você pode negar a solicitação de autenticação ou registrar os dados da sessão em um sistema externo.

> **Note**
>
> Esse acionador do Lambda não é ativado quando um usuário não existe, a menos que a configuração `PreventUserExistenceErrors` de um cliente de aplicação do grupo de usuários esteja definida como `ENABLED`. A renovação de uma sessão de autenticação existente também não ativa esse acionador.

Tópicos

- Visão geral do fluxo
- Parâmetros do acionador do Lambda de pré-autenticação
- Exemplo de pré-autenticação

Visão geral do fluxo



A solicitação inclui dados de validação do cliente dos valores `ClientMetadata` que a aplicação transmite para as operações de API `InitiateAuth` e `AdminInitiateAuth` do grupo de usuários.

Para obter mais informações, consulte Um exemplo de sessão de autenticação.

Parâmetros do acionador do Lambda de pré-autenticação

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
            . . .
        },
        "validationData": {
            "string": "string",
            . . .
        },
        "userNotFound": boolean
    },
    "response": {}
}
```

Parâmetros de solicitação de pré-autenticação

userAttributes

Um ou mais pares de nome-valor que representam atributos de usuário.

userNotFound

Quando você define `PreventUserExistenceErrors` como `ENABLED` para o cliente do grupo de usuários, o Amazon Cognito preenche esse booleano.

validationData

Um ou mais pares de chave-valor que contêm os dados de validação na solicitação de login do usuário. Para passar esses dados para sua função Lambda, use o ClientMetadata parâmetro nas ações InitiateAuthe da AdminInitiateAuthAPI.

Parâmetros de resposta de pré-autenticação

O Amazon Cognito não processa nenhuma informação adicionada que sua função retorna na resposta. Sua função pode retornar um erro para rejeitar a tentativa de login ou usar operações de API para consultar e modificar seus recursos.

Exemplo de pré-autenticação

Essa função de exemplo impede que usuários façam login em seu grupo de usuários com um cliente de aplicação específico. Como a função do Lambda de pré-autenticação não invoca quando o usuário já tem uma sessão, essa função só impede novas sessões com o ID do cliente da aplicação que você deseja bloquear.

Node.js

```
const handler = async (event) => {
  if (
    event.callerContext.clientId === "user-pool-app-client-id-to-be-blocked"
  ) {
    throw new Error("Cannot authenticate users from this user pool app client");
  }

  return event;
};

export { handler };
```

Python

```
def lambda_handler(event, context):
    if event['callerContext']['clientId'] == "<user pool app client id to be
 blocked>":
        raise Exception("Cannot authenticate users from this user pool app client")

    # Return to Amazon Cognito
    return event
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
    "callerContext": {
```

```
        "clientId": "<user pool app client id to be blocked>"
    },
    "response": {}
}
```

Acionador do Lambda de pós-autenticação

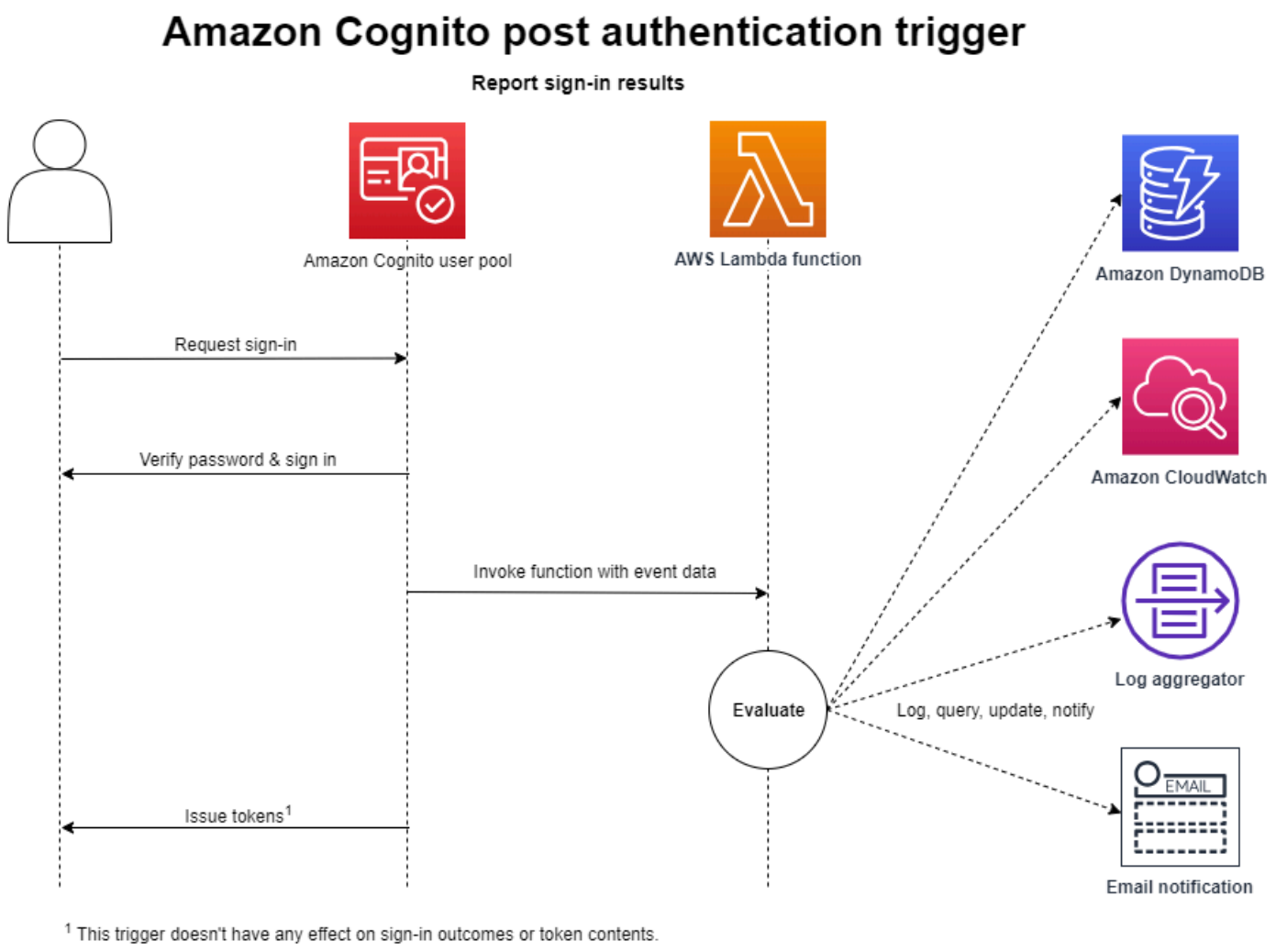
O acionador post authentication não altera o fluxo de autenticação de um usuário. O Amazon Cognito invoca esse Lambda após a conclusão da autenticação, antes que o usuário receba os tokens. Adicione um acionador post authentication quando quiser adicionar um pós-processamento personalizado de eventos de autenticação, por exemplo, registros ou ajustes de perfil de usuário que serão refletidos no próximo login.

Um Lambda post authentication que não retorna o corpo da solicitação ao Amazon Cognito ainda pode gerar falha na autenticação. Para obter mais informações, consulte O que é importante saber sobre acionadores do Lambda.

Tópicos

- Visão geral do fluxo de autenticação
- Parâmetros do acionador do Lambda de pós-autenticação
- Exemplo de pós-autenticação

Visão geral do fluxo de autenticação



Para obter mais informações, consulte Um exemplo de sessão de autenticação.

Parâmetros do acionador do Lambda de pós-autenticação

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "request": {
```

```
        "userAttributes": {
             "string": "string",
             . . .
         },
         "newDeviceUsed": boolean,
         "clientMetadata": {
             "string": "string",
             . . .
            }
        },
    "response": {}
}
```

Parâmetros de solicitação de pós-autenticação

newDeviceUsed

Esse sinalizador indica se o usuário fez login em um novo dispositivo. O Amazon Cognito só definirá esse sinalizador se o valor dos dispositivos memorizados do grupo de usuários for `Always` ou `User Opt-In`.

userAttributes

Um ou mais pares de nome-valor que representam atributos de usuário.

clientMetadata

Um ou mais pares de chave-valor que você pode fornecer como entrada personalizada para a função Lambda especificada para o acionador de pós-autenticação. Para passar esses dados para sua função Lambda, você pode usar o ClientMetadata parâmetro nas ações AdminRespondToAuthChallengee da RespondToAuthChallengeAPI. O Amazon Cognito não inclui dados do ClientMetadata parâmetro AdminInitiateAuthe operações de InitiateAuthAPI na solicitação que ele passa para a função de pós-autenticação.

Parâmetros de resposta de pós-autenticação

O Amazon Cognito não espera nenhuma outra informação de retorno na resposta. Sua função pode usar operações de API para consultar e modificar seus recursos ou registrar metadados de eventos em um sistema externo.

Exemplo de pós-autenticação

Este exemplo de função Lambda de pós-autenticação envia dados de um login bem-sucedido para o Logs. CloudWatch

Node.js

```
const handler = async (event) => {
  // Send post authentication data to Amazon CloudWatch logs
  console.log("Authentication successful");
  console.log("Trigger function =", event.triggerSource);
  console.log("User pool = ", event.userPoolId);
  console.log("App client ID = ", event.callerContext.clientId);
  console.log("User ID = ", event.userName);

  return event;
};

export { handler };
```

Python

```
import os
def lambda_handler(event, context):

    # Send post authentication data to Cloudwatch logs
    print ("Authentication successful")
    print ("Trigger function =", event['triggerSource'])
    print ("User pool = ", event['userPoolId'])
    print ("App client ID = ", event['callerContext']['clientId'])
    print ("User ID = ", event['userName'])

    # Return to Amazon Cognito
    return event
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
  "triggerSource": "testTrigger",
  "userPoolId": "testPool",
  "userName": "testName",
  "callerContext": {
      "clientId": "12345"
  },
  "response": {}
}
```

Acionador Lambda de federação de entrada

O gatilho de federação de entrada transforma os atributos do usuário federado durante o processo de autenticação com provedores de identidade externos. Quando os usuários se autenticam por meio de provedores de identidade configurados, esse gatilho permite que você modifique as respostas de provedores externos de SAML e OIDC interceptando e transformando dados no processo de autenticação, fornecendo controle programático sobre como os grupos de usuários do Amazon Cognito lidam com usuários federados e seus atributos.

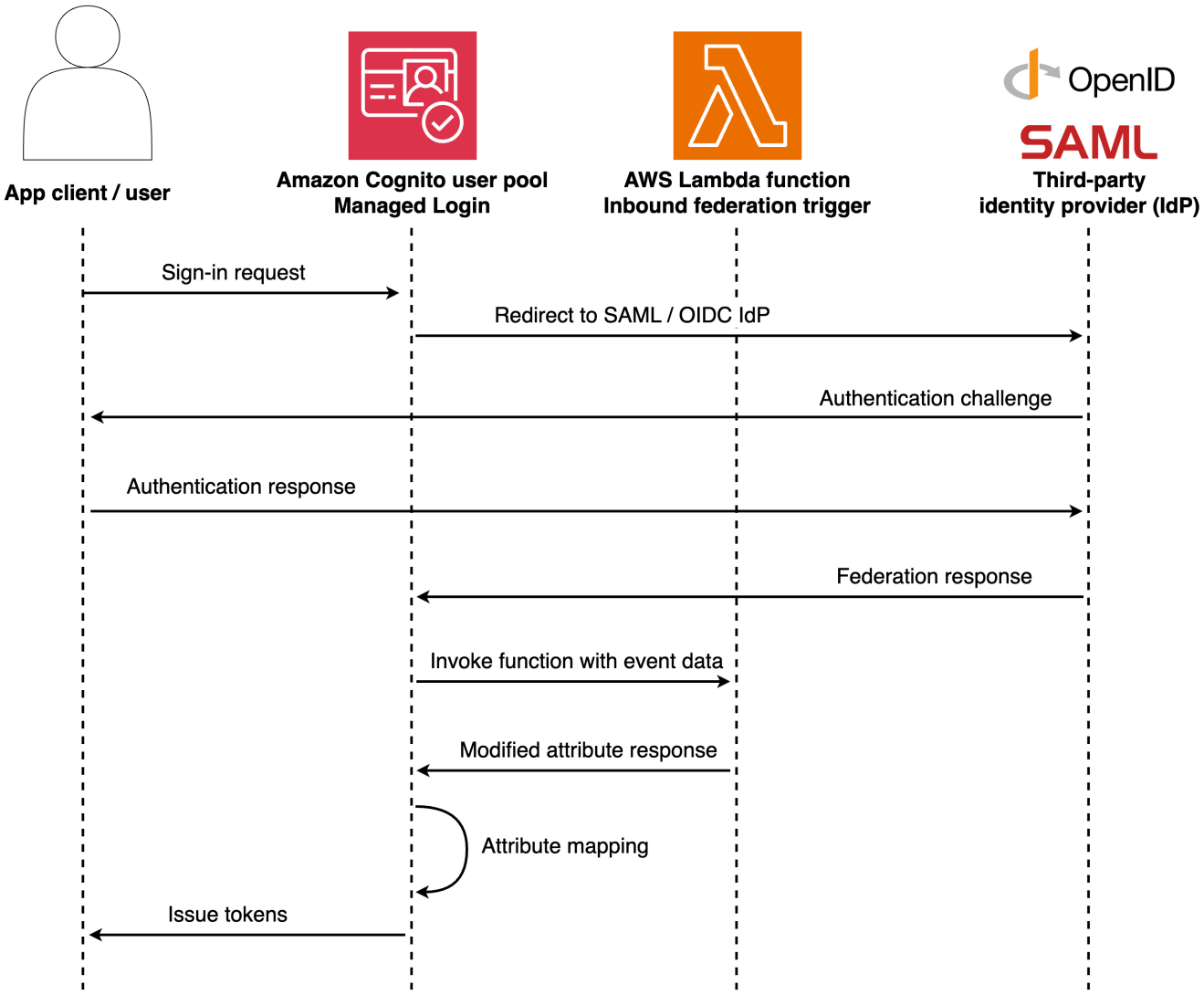
Use esse gatilho para adicionar, substituir ou suprimir atributos antes de criar novos usuários ou atualizar perfis de usuários federados existentes. Esse gatilho recebe atributos brutos do provedor de identidade como entrada e retorna atributos modificados que o Amazon Cognito aplica ao perfil do usuário.

Tópicos

Visão geral do fluxo

Quando um usuário se autentica com um provedor de identidade externo, o Amazon Cognito invoca o gatilho de federação de entrada antes de criar ou atualizar o perfil do usuário. O gatilho recebe os

atributos brutos do provedor de identidade e pode transformá-los antes que o Amazon Cognito os armazene. Esse fluxo ocorre tanto para novos usuários federados quanto para usuários existentes que se conectam novamente por meio da federação.



Parâmetros de gatilho do Lambda de federação de entrada

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
```

```
    "version": "string",
    "triggerSource": "InboundFederation_ExternalProvider",
    "region": AWSRegion,
    "userPoolId": "string",
    "userName": "string",
    "callerContext": {
        "awsSdkVersion": "string",
        "clientId": "string"
    },
    "request": {
        "providerName": "string",
        "providerType": "string",
        "attributes": {
            "tokenResponse": {
                "access_token": "string",
                "token_type": "string",
                "expires_in": "string"
            },
            "idToken": {
                "sub": "string",
                "email": "string",
                "email_verified": "string"
            },
            "userInfo": {
                "email": "string",
                "given_name": "string",
                "family_name": "string"
            },
            "samlResponse": {
                "string": "string"
            }
        }
    },
    "response": {
        "userAttributesToMap": {
            "string": "string"
        }
    }
}
```

Parâmetros de solicitação de federação de entrada

Nome do provedor

O nome do provedor de identidade externo.

Tipo de provedor

O tipo do provedor de identidade externo. Valores válidos:`OIDC`,`SAML`,`Facebook`,`Google`,`SignInWithApple`,`LoginWithAmazon`.

attributes

Os atributos brutos recebidos do provedor de identidade antes do processamento. A estrutura varia de acordo com o tipo de provedor.

Atributos. TokenResponse

OAuth dados de resposta do token do `/token` endpoint. Disponível somente para OIDC e provedores sociais. Contém `access_token`id_token, `refresh_token`token_type,`expires_in`,, `scope` e.

Attributes.idToken

O token de ID decodificado e validado que a JWT reivindica. Disponível somente para OIDC e provedores sociais. Contém informações verificadas de identidade do usuário, incluindo `sub` (identificador de usuário exclusivo) `email`name,,, `iss` (emissor), `aud` (público), `exp` (expiração) e `iat` (horário de emissão).

Atributos. Informações do usuário

Informações estendidas do perfil do usuário a partir do UserInfo endpoint. Disponível somente para OIDC e provedores sociais. Contém atributos de perfil detalhados`given_name`, como`family_name`,`picture`,`address`, e outros campos específicos do provedor. Pode estar vazio se o IdP não suportar o UserInfo endpoint ou se a chamada do endpoint falhar.

Atributos. Resposta SAML

Atributos de asserção SAML. Disponível somente para provedores de SAML. Contém atributos da resposta SAML.

Parâmetros de resposta da federação de entrada

userAttributesToMapa

Os atributos do usuário a serem aplicados ao perfil do usuário.

Important

Você deve incluir TODOS os atributos do usuário que deseja reter na resposta, incluindo os atributos que você não está modificando. Todos os atributos não incluídos na `userAttributesToMap` resposta serão descartados e não serão armazenados no perfil do usuário. Isso se aplica tanto aos atributos modificados quanto aos não modificados.

Comportamento de resposta vazia

Se você retornar um objeto vazio `{}` para`userAttributesToMap`, todos os atributos originais do provedor de identidade serão mantidos inalterados. Isso funciona como um sistema autônomo, como se a função Lambda nunca tivesse sido executada. Isso é diferente de omitir atributos, o que os elimina.

Atributos específicos do provedor

A estrutura do `request.attributes` varia com base no`providerType`. O OIDC e os provedores sociais incluem `tokenResponse`, `idToken`, e `userInfo` objetos. Os provedores de SAML incluem somente o `samlResponse` objeto.

Exemplo de federação de entrada: gerenciamento de membros de grupos

Este exemplo mostra como mapear grupos de provedores de identidade federados para grupos de grupos de usuários do Amazon Cognito. Essa função extrai a associação ao grupo da resposta federada e adiciona automaticamente usuários aos grupos correspondentes do Amazon Cognito, eliminando a necessidade de acionadores de pós-autenticação.

Node.js

```
exports.handler = async (event) => {
    const { providerType, attributes } = event.request;

    // Extract user attributes based on provider type
    let userAttributesFromIdp = {};
    if (providerType === 'SAML') {
        userAttributesFromIdp = attributes.samlResponse || {};
    } else {
        // For OIDC and Social providers, merge userInfo and idToken
        userAttributesFromIdp = {
            ...(attributes.userInfo || {}),
            ...(attributes.idToken || {})
        };
    }

    // Extract groups from federated response
    const federatedGroups = userAttributesFromIdp.groups?.split(',') || [];

    // Map federated groups to Cognito groups
    const groupMapping = {
        'Domain Admins': 'Administrators',
        'Engineering': 'Developers',
        'Sales': 'SalesTeam'
    };

    // Filter to only in-scope groups
    const mappedGroups = federatedGroups
        .map(group => groupMapping[group.trim()])
        .filter(group => group); // Remove undefined values

    // Pass through attributes with mapped groups as custom attribute
    const attributesToMap = {
        ...userAttributesFromIdp,
        'custom:user_groups': mappedGroups.join(',')
    };

    // Remove original groups attribute
    delete attributesToMap.groups;

    event.response.userAttributesToMap = attributesToMap;
    return event;
```

```
};
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
    "userPoolId": "us-east-1_XXXXXXXXX",
    "request": {
        "providerName": "CorporateAD",
        "providerType": "SAML",
        "attributes": {
            "samlResponse": {
                "email": "jane.smith@company.com",
                "given_name": "Jane",
                "family_name": "Smith",
                "groups": "Engineering,Domain Admins",
                "department": "Engineering"
            }
        }
    },
    "response": {
        "userAttributesToMap": {}
    }
}
```

Exemplo de federação de entrada: truncar atributos grandes

Este exemplo mostra como truncar valores de atributos que excedem os limites de armazenamento do Amazon Cognito. Essa função verifica cada atributo do provedor de identidade. Se um valor de atributo exceder 2048 caracteres, ele trunca o valor e adiciona reticências para indicar truncamento. Todos os outros atributos passam inalterados.

Node.js

```
exports.handler = async (event) => {
    const MAX_ATTRIBUTE_LENGTH = 2048;
```

```
    // Get the identity provider attributes based on provider type
    const { providerType, attributes } = event.request;
    let idpAttributes = {};

    if (providerType === 'SAML') {
        idpAttributes = attributes.samlResponse || {};
    } else {
        // For OIDC and Social providers, merge userInfo and idToken
        idpAttributes = {
            ...(attributes.userInfo || {}),
            ...(attributes.idToken || {})
        };
    }

    const userAttributes = {};

    // Process each attribute
    for (const [key, value] of Object.entries(idpAttributes)) {
        if (typeof value === 'string' && value.length > MAX_ATTRIBUTE_LENGTH) {
            // Truncate the value and add ellipsis
            userAttributes[key] = value.substring(0, MAX_ATTRIBUTE_LENGTH - 3) +
 '...';
            console.log(`Truncated attribute ${key} from ${value.length} to
 ${userAttributes[key].length} characters`);
        } else {
            // Keep the original value
            userAttributes[key] = value;
        }
    }

    // Return the modified attributes
    event.response.userAttributesToMap = userAttributes;
    return event;
};
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
    "version": "string",
    "triggerSource": "InboundFederation_ExternalProvider",
    "region": "us-east-1",
    "userPoolId": "us-east-1_XXXXXXXXX",
    "userName": "ExampleProvider_12345",
    "callerContext": {
        "awsSdkVersion": "string",
        "clientId": "string"
    },
    "request": {
        "providerName": "ExampleProvider",
        "providerType": "OIDC",
        "attributes": {
            "tokenResponse": {
                "access_token": "abcDE...",
                "token_type": "Bearer",
                "expires_in": "3600"
            },
            "idToken": {
                "sub": "12345",
                "email": "user@example.com"
            },
            "userInfo": {
                "email": "user@example.com",
                "given_name": "Example",
                "family_name": "User",
                "bio": "This is a very long biography that contains more than 2048
 characters..."
            }
        }
    },
    "response": {
        "userAttributesToMap": {}
    }
}
```

Exemplo de federação de entrada: registro de eventos de federação

Este exemplo mostra como registrar eventos de autenticação federada para monitoramento e depuração. Esse exemplo de função captura informações detalhadas sobre usuários federados e seus atributos, fornecendo visibilidade do processo de autenticação.

Node.js

```
exports.handler = async (event) => {
    const { providerName, providerType, attributes } = event.request;

    // Extract user attributes based on provider type
    let userAttributesFromIdp = {};
    if (providerType === 'SAML') {
        userAttributesFromIdp = attributes.samlResponse || {};
    } else {
        // For OIDC and Social providers, merge userInfo and idToken
        userAttributesFromIdp = {
            ...(attributes.userInfo || {}),
            ...(attributes.idToken || {})
        };
    }

    // Log federated authentication details
    console.log(JSON.stringify({
        timestamp: new Date().toISOString(),
        providerName,
        providerType,
        userEmail: userAttributesFromIdp.email,
        attributeCount: Object.keys(userAttributesFromIdp).length,
        attributes: userAttributesFromIdp
    }));

    // Pass through all attributes unchanged
    event.response.userAttributesToMap = userAttributesFromIdp;
    return event;
};
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console

do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
    "version": "string",
    "triggerSource": "InboundFederation_ExternalProvider",
    "region": "us-east-1",
    "userPoolId": "us-east-1_XXXXXXXXX",
    "userName": "CorporateAD_john.doe",
    "callerContext": {
        "awsSdkVersion": "string",
        "clientId": "string"
    },
    "request": {
        "providerName": "CorporateAD",
        "providerType": "SAML",
        "attributes": {
            "samlResponse": {
                "email": "john.doe@company.com",
                "given_name": "John",
                "family_name": "Doe",
                "department": "Engineering",
                "employee_id": "EMP12345"
            }
        }
    },
    "response": {
        "userAttributesToMap": {}
    }
}
```

Saída CloudWatch de registros esperada:

JSON

```
{
    "timestamp": "2025-01-14T21:17:40.153Z",
    "providerName": "CorporateAD",
    "providerType": "SAML",
    "userEmail": "john.doe@company.com",
```

```
    "attributeCount": 5,
    "attributes": {
        "email": "john.doe@company.com",
        "given_name": "John",
        "family_name": "Doe",
        "department": "Engineering",
        "employee_id": "EMP12345"
    }
}
```

Acionadores do Lambda de desafio personalizado de autenticação

Ao criar seus fluxos de autenticação para seu grupo de usuários do Amazon Cognito, você pode querer estender seu modelo de autenticação além dos fluxos integrados. Um caso de uso comum dos acionadores de desafio personalizados é implementar verificações de segurança adicionais além do nome de usuário, senha e autenticação multifator (MFA). Desafio personalizado é qualquer pergunta e resposta que você possa gerar em uma linguagem de programação compatível com Lambda. Por exemplo, você pode exigir que os usuários resolvam um CAPTCHA ou respondam a uma pergunta de segurança antes de poderem se autenticar. Outra necessidade potencial é a integração com fatores ou dispositivos de autenticação especializados. Ou talvez você já tenha desenvolvido um software que autentica usuários com uma chave de segurança de hardware ou um dispositivo biométrico. A definição de sucesso na autenticação para um desafio personalizado é qualquer resposta que sua função do Lambda aceite como correta: uma string fixa, por exemplo, ou uma resposta satisfatória de uma API externa.

Você pode iniciar a autenticação com seu desafio personalizado e controlar totalmente o processo de autenticação, ou pode realizar a autenticação por nome de usuário e senha antes que sua aplicação receba seu desafio personalizado.

O acionador do Lambda do desafio de autenticação personalizada:

Define

Inicia uma sequência de desafios. Determina se você deseja iniciar um novo desafio, marcar a autenticação como concluída ou interromper a tentativa de autenticação.

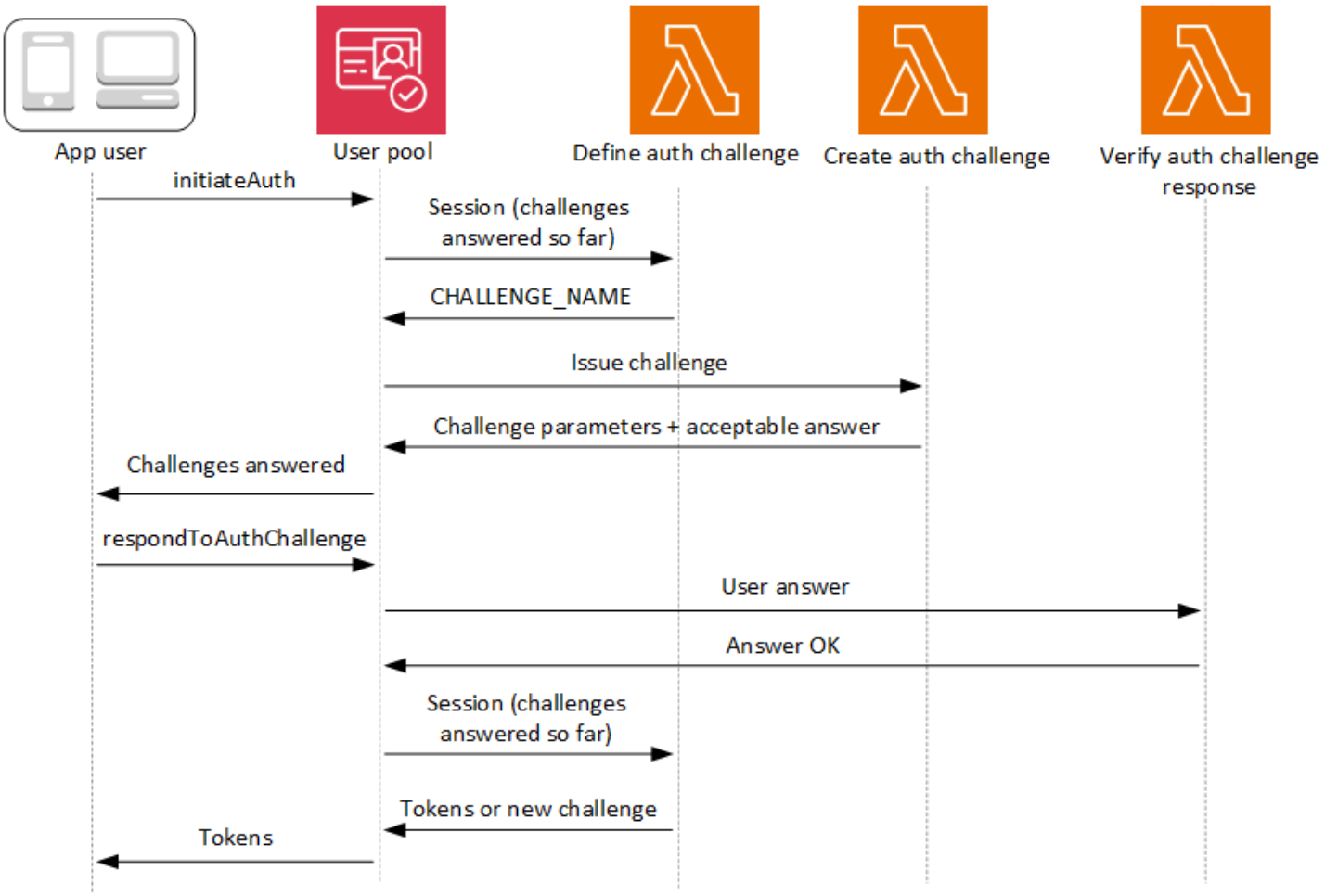
Cria

Emite a pergunta para a aplicação para o usuário responder. Essa função pode apresentar uma pergunta de segurança ou um link para um CAPTCHA que sua aplicação deve exibir para o usuário.

Verifica

Conhece a resposta esperada e a compara com a resposta que sua aplicação fornece na resposta ao desafio. A função pode chamar a API do seu serviço CAPTCHA para recuperar os resultados esperados da tentativa de solução do usuário.

Essas três funções do Lambda se agrupam para apresentar um mecanismo de autenticação que está completamente sob seu controle e sob seu próprio padrão. Como a autenticação personalizada requer lógica de aplicação no seu cliente e nas funções do Lambda, você não pode processar a autenticação personalizada no login gerenciado. Esse sistema de autenticação exige um esforço adicional do desenvolvedor. Sua aplicação deve realizar o fluxo de autenticação com a API de grupos de usuários e lidar com o desafio resultante com uma interface de login personalizada que coloque a pergunta no centro do desafio de autenticação personalizada.



Para mais informações sobre como implementar uma autenticação personalizada, consulte Fluxo de autenticação personalizado e desafios.

Autenticação entre as operações da API InitiateAuthou AdminInitiateAuth, e RespondToAuthChallengeou AdminRespondToAuthChallenge. Nesse fluxo, um usuário faz a autenticação respondendo desafios sucessivos até que ela falhe ou ele receba os tokens. Uma resposta ao desafio pode ser um novo desafio. Nesse caso, sua aplicação responde quantas vezes forem necessárias aos novos desafios. A autenticação bem-sucedida acontece quando a função define auth analisa os resultados até o momento, determina que todos os desafios foram respondidos e retorna `IssueTokens`.

Tópicos

- Autenticação SRP em fluxos de desafio personalizados
- Acionador do Lambda para definir desafio de autenticação
- Acionador do Lambda de criar desafio de autenticação

- Acionador do Lambda de verificar resposta do desafio de autenticação

Autenticação SRP em fluxos de desafio personalizados

Você pode fazer com que o Amazon Cognito verifique senhas de usuário antes que ele emita seus desafios personalizados. Todos os gatilhos do Lambda associados à categoria Autenticação das cotas de taxa de solicitação serão executados quando você realizar a autenticação SRP em um fluxo de desafio personalizado. Veja uma visão geral do processo:

1. Sua aplicação inicia o login chamando `InitiateAuth` ou `AdminInitiateAuth` com o mapa `AuthParameters`. Os parâmetros devem incluir `CHALLENGE_NAME: SRP_A,` e os valores de `SRP_A` e `USERNAME`.
2. O Amazon Cognito invoca o acionador do Lambda de desafio de autenticação com uma sessão inicial que contém `challengeName: SRP_A` e `challengeResult: true`.
3. Depois de receber essas entradas, a função do Lambda responde com `challengeName: PASSWORD_VERIFIER`, `issueTokens: false`, `failAuthentication: false`.
4. Se a verificação de senha for bem-sucedida, o Amazon Cognito invocará sua função do Lambda novamente com uma nova sessão contendo `challengeName: PASSWORD_VERIFIER` e `challengeResult: true`.
5. Para iniciar seus desafios personalizados, sua função do Lambda responde com `challengeName: CUSTOM_CHALLENGE`, `issueTokens: false` e `failAuthentication: false`. Se você não quiser iniciar seu fluxo de autenticação personalizado com a verificação de senha, poderá iniciar o login com o mapa `AuthParameters` incluindo `CHALLENGE_NAME: CUSTOM_CHALLENGE`.
6. O loop de desafios se repetirá até que todos os desafios sejam respondidos.

O exemplo a seguir mostra uma solicitação `InitiateAuth` inicial que precede a autenticação personalizada com um fluxo SRP.

```
{
    "AuthFlow": "CUSTOM_AUTH",
    "ClientId": "1example23456789",
    "AuthParameters": {
        "CHALLENGE_NAME": "SRP_A",
        "USERNAME": "testuser",
        "SRP_A": "[SRP_A]",
```

```
        "SECRET_HASH": "[secret hash]"
    }
}
```

Redefinição de senha no fluxo SRP de autenticação personalizada

Quando os usuários têm status `FORCE_CHANGE_PASSWORD`, seu fluxo de autenticação personalizado deve integrar a etapa de alteração de senha e, ao mesmo tempo, manter a integridade de seus desafios de autenticação. O Amazon Cognito invoca seu acionador do Lambda desafio de definição de autenticação durante o desafio `NEW_PASSWORD_REQUIRED`. Nesse cenário, um usuário que faz login com um fluxo de desafio personalizado e autenticação SRP pode definir uma nova senha se estiver em um estado de redefinição de senha.

Quando os usuários têm o status `FORCE_CHANGE_PASSWORD` ou `RESET_REQUIRED`, eles devem responder a um desafio `NEW_PASSWORD_REQUIRED` com um `NEW_PASSWORD`. Na autenticação personalizada com SRP, o Amazon Cognito retorna um desafio `NEW_PASSWORD_REQUIRED` depois que os usuários concluem o desafio `PASSWORD_VERIFIER` SRP. Seu acionador de desafio de definição de autenticação recebe os dois resultados do desafio na matriz `session` e pode continuar com desafios personalizados adicionais depois que o usuário alterar a senha com sucesso.

Seu acionador do Lambda do desafio de autenticação definido deve gerenciar a sequência de desafios pela autenticação SRP, redefinição de senha e desafios personalizados subsequentes. O acionador recebe uma matriz de desafios concluídos no parâmetro `session`, incluindo os resultados `PASSWORD_VERIFIER` e `NEW_PASSWORD_REQUIRED`. Para obter um exemplo de implementação, consulte Exemplo de definição do desafio de autenticação.

Etapas do fluxo de autenticação

Para usuários que precisam verificar a senha antes de desafios personalizados, o processo segue estas etapas:

1. Sua aplicação inicia o login chamando `InitiateAuth` ou `AdminInitiateAuth` com o mapa `AuthParameters`. Os parâmetros devem incluir `CHALLENGE_NAME: SRP_A` e os valores de `SRP_A` e `USERNAME`.
2. O Amazon Cognito invoca o acionador do Lambda de desafio de autenticação com uma sessão inicial que contém `challengeName: SRP_A` e `challengeResult: true`.
3. Depois de receber essas entradas, a função do Lambda responde com `challengeName: PASSWORD_VERIFIER`, `issueTokens: false`, `failAuthentication: false`.

4. Se a verificação da senha for bem-sucedida, uma das duas coisas acontecerá:

   Para usuários com status normal:

   O Amazon Cognito invoca sua função do Lambda novamente com uma nova sessão contendo `challengeName: PASSWORD_VERIFIER` e `challengeResult: true`.

   Para iniciar seus desafios personalizados, sua função do Lambda responde com `challengeName: CUSTOM_CHALLENGE`, `issueTokens: false` e `failAuthentication: false`.

   Para usuários com status **`RESET_REQUIRED`** ou **`FORCE_CHANGE_PASSWORD`**:

   O Amazon Cognito invoca sua função do Lambda com uma sessão contendo `challengeName: PASSWORD_VERIFIER` e `challengeResult: true`.

   Sua função do Lambda deve responder com `challengeName: NEW_PASSWORD_REQUIRED`, `issueTokens: false` e `failAuthentication: false`.

   Após a alteração bem-sucedida da senha, o Amazon Cognito invoca sua função do Lambda com uma sessão contendo os resultados `PASSWORD_VERIFIER` e `NEW_PASSWORD_REQUIRED`.

   Para iniciar seus desafios personalizados, sua função do Lambda responde com `challengeName: CUSTOM_CHALLENGE`, `issueTokens: false` e `failAuthentication: false`.
5. O loop de desafios se repetirá até que todos os desafios sejam respondidos.

Se você não quiser iniciar seu fluxo de autenticação personalizado com a verificação de senha, poderá iniciar o login com o mapa `AuthParameters` incluindo `CHALLENGE_NAME: CUSTOM_CHALLENGE`.

Gerenciamento de sessões

O fluxo de autenticação mantém a continuidade da sessão por meio de uma série de resultados de sessões IDs e desafios. Cada resposta ao desafio gera um novo ID de sessão para evitar erros de reutilização da sessão, o que é importante principalmente para fluxos de autenticação multifator.

Os resultados do desafio são armazenados cronologicamente na matriz de sessões que seus acionadores do Lambda recebem. Para usuários com status `FORCE_CHANGE_PASSWORD`, a matriz da sessão contém:

1. `session[0]` – Desafio `SRP_A` inicial
2. `session[1]` – Resultado `PASSWORD_VERIFIER`
3. `session[2]` – Resultado `NEW_PASSWORD_REQUIRED`
4. Elementos subsequentes – Resultados de desafios personalizados adicionais

Exemplo de fluxo de autorização

O exemplo a seguir demonstra um fluxo completo de autenticação personalizada para um usuário com status `FORCE_CHANGE_PASSWORD` que precisa concluir a alteração da senha e um desafio CAPTCHA personalizado.

1. InitiateAuth request

```
{
    "AuthFlow": "CUSTOM_AUTH",
    "ClientId": "1example23456789",
    "AuthParameters": {
        "CHALLENGE_NAME": "SRP_A",
        "USERNAME": "testuser",
        "SRP_A": "[SRP_A]"
    }
}
```

2. InitiateAuth resposta

```
{
    "ChallengeName": "PASSWORD_VERIFIER",
    "ChallengeParameters": {
        "USER_ID_FOR_SRP": "testuser"
    },
    "Session": "[session_id_1]"
}
```

3. RespondToAuthChallenge solicitação com **`PASSWORD_VERIFIER`**

```
{
    "ChallengeName": "PASSWORD_VERIFIER",
    "ClientId": "1example23456789",
    "ChallengeResponses": {
        "PASSWORD_CLAIM_SIGNATURE": "[claim_signature]",
        "PASSWORD_CLAIM_SECRET_BLOCK": "[secret_block]",
```

```
        "TIMESTAMP": "[timestamp]",
        "USERNAME": "testuser"
    },
    "Session": "[session_id_1]"
}
```

4. RespondToAuthChallenge resposta com **NEW_PASSWORD_REQUIRED** desafio

```
{
    "ChallengeName": "NEW_PASSWORD_REQUIRED",
    "ChallengeParameters": {},
    "Session": "[session_id_2]"
}
```

5. RespondToAuthChallenge solicitação com **NEW_PASSWORD_REQUIRED**

```
{
    "ChallengeName": "NEW_PASSWORD_REQUIRED",
    "ClientId": "1example23456789",
    "ChallengeResponses": {
        "NEW_PASSWORD": "[password]",
        "USERNAME": "testuser"
    },
    "Session": "[session_id_2]"
}
```

6. RespondToAuthChallenge resposta com desafio personalizado de CAPTCHA

```
{
    "ChallengeName": "CUSTOM_CHALLENGE",
    "ChallengeParameters": {
        "captchaUrl": "url/123.jpg"
    },
    "Session": "[session_id_3]"
}
```

7. RespondToAuthChallenge solicitação com resposta ao desafio personalizado CAPTCHA

```
{
    "ChallengeName": "CUSTOM_CHALLENGE",
    "ClientId": "1example23456789",
    "ChallengeResponses": {
        "ANSWER": "123",
```

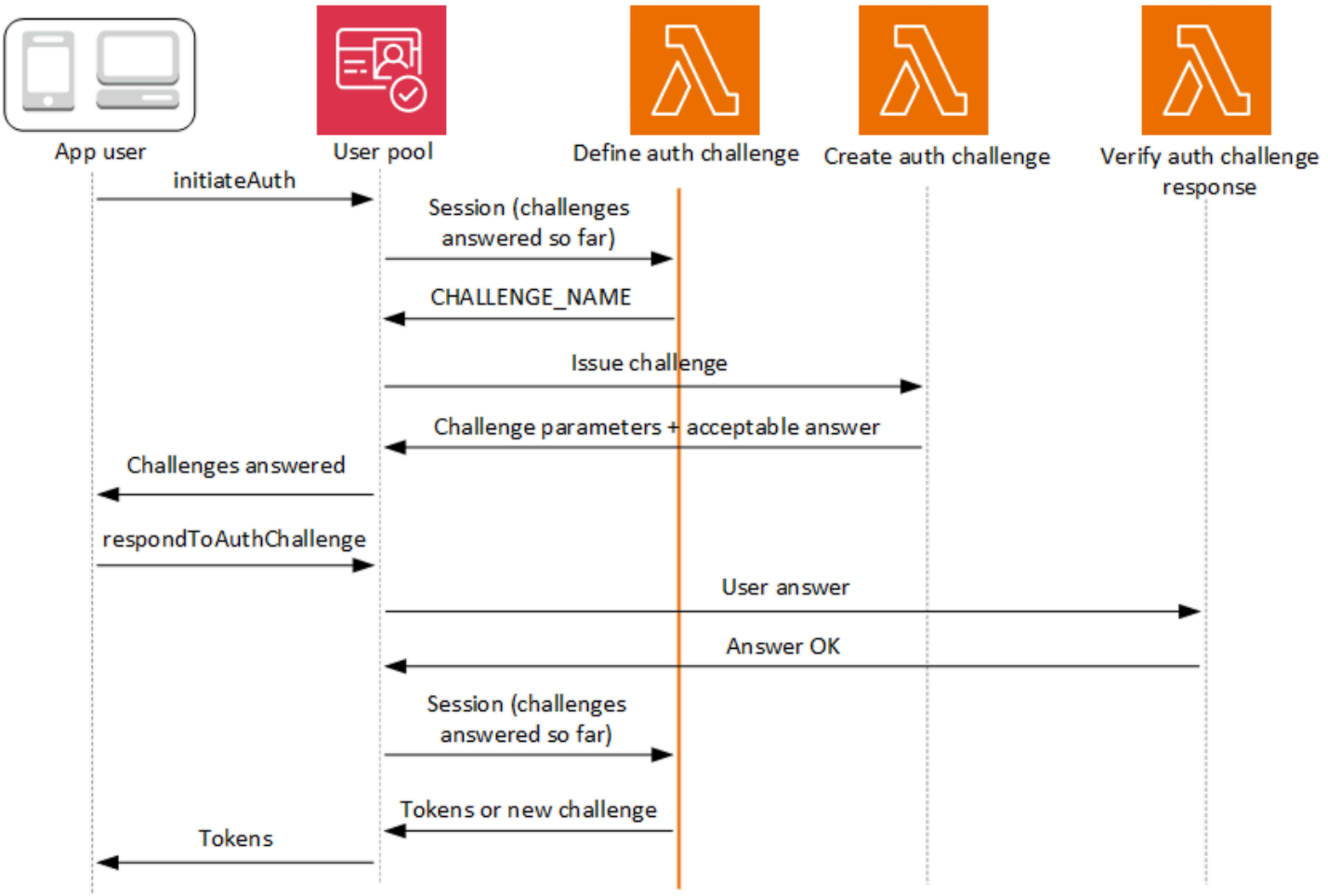
```
        "USERNAME": "testuser"
    },
    "Session": "[session_id_3]"
}
```

6. Resposta final de sucesso

```
{
    "AuthenticationResult": {
        "AccessToken": "eyJra456defEXAMPLE",
        "ExpiresIn": 3600,
        "IdToken": "eyJra789ghiEXAMPLE",
        "RefreshToken": "eyJjd123abcEXAMPLE",
        "TokenType": "Bearer"
    },
    "ChallengeParameters": {}
}
```

Acionador do Lambda para definir desafio de autenticação

O gatilho de desafio define auth é uma função do Lambda que mantém a sequência de desafios em um fluxo de autenticação personalizado. Ele declara o sucesso ou o fracasso da sequência de desafios e define o próximo desafio se a sequência ainda não estiver completa.



Definir o desafio de autenticação

O Amazon Cognito invoca esse acionador para iniciar o fluxo de autenticação personalizado.

A solicitação desse acionador do Lambda contém `session`. O parâmetro `session` é uma matriz que contém todos os desafios apresentados ao usuário no processo de autenticação atual. A solicitação também inclui o resultado correspondente. A matriz `session` armazena detalhes do desafio (`ChallengeResult`) em ordem cronológica. O desafio `session[0]` representa o primeiro que o usuário recebe.

Tópicos

- Parâmetros do acionador do Lambda para definir o desafio de autenticação
- Exemplo de definição do desafio de autenticação

Parâmetros do acionador do Lambda para definir o desafio de autenticação

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
                . . .
        },
        "session": [
            ChallengeResult,
            . . .
        ],
        "clientMetadata": {
            "string": "string",
            . . .
        },
        "userNotFound": boolean
    },
    "response": {
        "challengeName": "string",
        "issueTokens": boolean,
        "failAuthentication": boolean
    }
}
```

Parâmetros de solicitação para definir o desafio de autenticação

Quando o Amazon Cognito invoca sua função do Lambda, ele fornece os seguintes parâmetros:

userAttributes

Um ou mais pares de nome-valor que representam atributos de usuário.

userNotFound

Um booleano que é preenchido pelo Amazon Cognito quando `PreventUserExistenceErrors` é definido como `ENABLED` para o cliente de grupo de usuários. Um valor de `true` significa que

o ID do usuário (nome de usuário, endereço de e-mail e outros detalhes) não correspondeu a nenhum usuário existente. Quando `PreventUserExistenceErrors` é definido como `ENABLED`, o serviço não informa a aplicação dos usuários inexistentes. Recomendamos que suas funções do Lambda mantenham a mesma experiência do usuário e contabilizem a latência. Dessa forma, o autor da chamada não consegue detectar comportamentos diferentes quando o usuário existe ou não existe.

sessão

Uma matriz de elementos `ChallengeResult`. Cada regra contém os seguintes elementos:

challengeName

Um dos seguintes tipos de desafio: `CUSTOM_CHALLENGE`, `SRP_A`, `PASSWORD_VERIFIER`, `SMS_MFA`, `EMAIL_OTP`, `SOFTWARE_TOKEN_MFA`, `DEVICE_SRP_AUTH`, `DEVICE_PASSWORD_VERIFIER` ou `ADMIN_NO_SRP_AUTH`.

Quando sua função define auth challenge emite um desafio `PASSWORD_VERIFIER` para um usuário que configurou a autenticação multifator, o Amazon Cognito prossegue com um desafio `SMS_MFA`, `EMAIL_OTP` ou `SOFTWARE_TOKEN_MFA`. Essas são as instruções para um código de autenticação multifator. Em sua função, inclua o tratamento de eventos de entrada de desafios `SMS_MFA`, `EMAIL_OTP` e `SOFTWARE_TOKEN_MFA`. Você não precisa invocar os desafios de MFA usando sua função de desafio define auth.

> **Important**
>
> Quando sua função estiver determinando se um usuário fez a autenticação com êxito e você precisar emitir tokens para ele, sempre confira `challengeName` em sua função “define auth challenge” e garantir que corresponda ao valor esperado.

challengeResult

Defina como `true` se o usuário tiver concluído o desafio com êxito; do contrário, defina-o como `false`.

challengeMetadata

Seu nome para o desafio personalizado. Usado somente se `challengeName` for `CUSTOM_CHALLENGE`.

clientMetadata

Um ou mais pares de chave/valor que você pode fornecer como entrada personalizada para a função do Lambda especificada para o acionador definir desafio de autenticação. Para passar esses dados para sua função Lambda, você pode usar o `ClientMetadata` parâmetro nas operações AdminRespondToAuthChallengee da RespondToAuthChallengeAPI. A solicitação que invoca a função define auth challenge não inclui dados transmitidos no ClientMetadata parâmetro AdminInitiateAuthe InitiateAuthoperações de API.

Parâmetros de resposta para definir o desafio de autenticação

Na resposta, você pode retornar o próximo estágio do processo de autenticação.

challengeName

Uma string que contém o nome do próximo desafio. Se você deseja apresentar um novo desafio ao seu usuário, especifique o nome do desafio aqui.

issueTokens

Se você determinar que o usuário concluiu os desafios de autenticação de forma adequada; defina-o como `true`. Se o usuário não cumprir os desafios devidamente, defina como `false`.

failAuthentication

Se quiser encerrar o processo de autenticação atual, defina-o como `true`. Para continuar o processo de autenticação atual, defina-o como `false`.

Exemplo de definição do desafio de autenticação

Este exemplo definirá uma série de desafios de autenticação e emitirá tokens somente se o usuário concluir todos os desafios com êxito. Quando os usuários concluem a autenticação SRP com os desafios `SRP_A` e `PASSWORD_VERIFIER`, essa função transmite a eles um `CUSTOM_CHALLENGE` que invoca o acionador do desafio de criação de autorização. Em combinação com nosso exemplo de desafio de criação de autenticação, essa sequência oferece um desafio CAPTCHA para o desafio três e uma pergunta de segurança para o desafio quatro.

Depois que o usuário resolve o CAPTCHA e responde à pergunta de segurança, essa função confirma que seu grupo de usuários pode emitir tokens. A autenticação SRP não é necessária; você também pode definir o CAPTCHA e a pergunta de segurança como desafios um e dois. Caso sua

função de definição de desafio de autenticação não declare desafios de SRP, o sucesso de seus usuários será determinado inteiramente pelas respostas deles aos seus prompts personalizados.

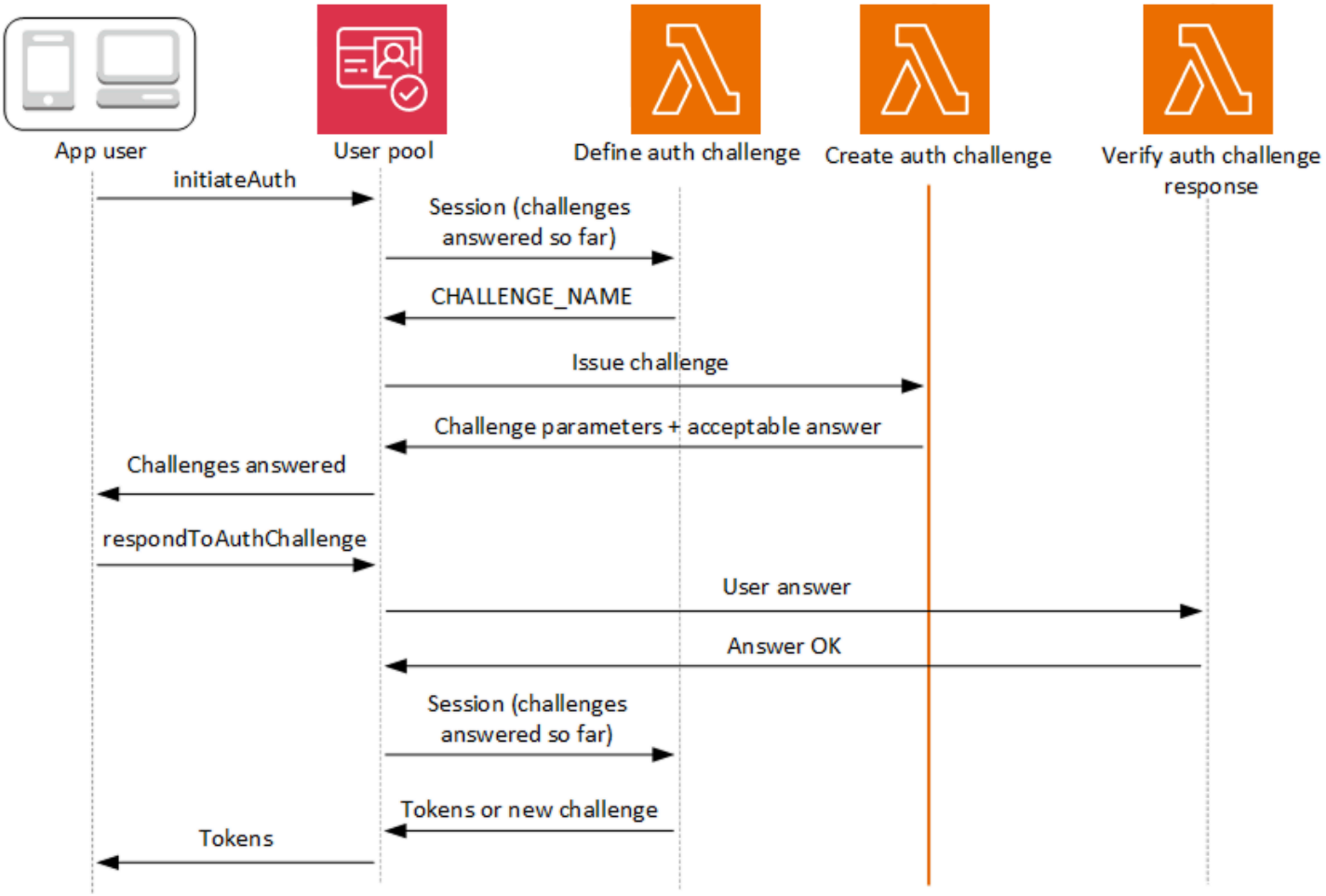
Node.js

```
const handler = async (event) => {
  if (
    event.request.session.length === 1 &&
    event.request.session[0].challengeName === "SRP_A"
  ) {
    event.response.issueTokens = false;
    event.response.failAuthentication = false;
    event.response.challengeName = "PASSWORD_VERIFIER";
  } else if (
    event.request.session.length === 2 &&
    event.request.session[1].challengeName === "PASSWORD_VERIFIER" &&
    event.request.session[1].challengeResult === true
  ) {
    event.response.issueTokens = false;
    event.response.failAuthentication = false;
    event.response.challengeName = "CUSTOM_CHALLENGE";
  } else if (
    event.request.session.length === 3 &&
    event.request.session[2].challengeName === "CUSTOM_CHALLENGE" &&
    event.request.session[2].challengeResult === true
  ) {
    event.response.issueTokens = false;
    event.response.failAuthentication = false;
    event.response.challengeName = "CUSTOM_CHALLENGE";
  } else if (
    event.request.session.length === 4 &&
    event.request.session[3].challengeName === "CUSTOM_CHALLENGE" &&
    event.request.session[3].challengeResult === true
  ) {
    event.response.issueTokens = true;
    event.response.failAuthentication = false;
  } else {
    event.response.issueTokens = false;
    event.response.failAuthentication = true;
  }

  return event;
};
```

```
export { handler };
```

Acionador do Lambda de criar desafio de autenticação

O acionador de desafio create auth é uma função do Lambda que tem os detalhes de cada desafio declarado pelo acionador de desafio define auth. Ele processa o nome do desafio declarado pelo acionador de desafio define auth e retorna um `publicChallengeParameters` que sua aplicação deve apresentar ao usuário. Essa função então fornece ao seu grupo de usuários a resposta para o desafio `privateChallengeParameters`, que seu grupo de usuários passa para o acionador do desafio verify auth. Onde seu acionador de desafio define auth gerencia a sequência de desafios, seu acionador de desafio create auth gerencia o conteúdo do desafio.



Criar desafio de autenticação

O Amazon Cognito invocará esse acionador depois de Definir desafio de autenticação se um desafio personalizado tiver sido especificado como parte do acionador Definir desafio de autenticação. Ele cria um fluxo de autenticação personalizado.

Esse acionador do Lambda é invocado para criar um desafio a ser apresentado ao usuário. A solicitação deste acionador do Lambda inclui `challengeName` e `session`. O `challengeName` é uma string que representa o nome do próximo desafio a ser apresentado ao usuário. O valor desse atributo é definido no acionador do Lambda Definir desafio de autenticação.

O loop de desafio será repetido até todos os desafios serem respondidos.

Tópicos

- Parâmetros do acionador do Lambda de criar desafio de autenticação
- Exemplo de criar desafio de autenticação

Parâmetros do acionador do Lambda de criar desafio de autenticação

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
            . . .
        },
        "challengeName": "string",
        "session": [
            ChallengeResult,
            . . .
        ],
        "clientMetadata": {
            "string": "string",
            . . .
        },
```

```
        "userNotFound": boolean
    },
    "response": {
        "publicChallengeParameters": {
            "string": "string",
            . . .
        },
        "privateChallengeParameters": {
            "string": "string",
            . . .
        },
        "challengeMetadata": "string"
    }
}
```

Parâmetros de solicitação de criar desafio de autenticação

userAttributes

Um ou mais pares de nome-valor que representam atributos de usuário.

userNotFound

Este booleano é preenchido quando `PreventUserExistenceErrors` é configurado como `ENABLED` para o cliente de grupo de usuários.

challengeName

O nome do novo desafio.

sessão

O elemento session é uma matriz de elementos `ChallengeResult`, cada um deles contendo os seguintes elementos:

challengeName

O tipo de desafio. Um destes: `"CUSTOM_CHALLENGE"`, `"PASSWORD_VERIFIER"`, `"SMS_MFA"`, `"DEVICE_SRP_AUTH"`, `"DEVICE_PASSWORD_VERIFIER"`, `"NEW_PASSWORD_REQUIRED"` ou `"ADMIN_NO_SRP_AUTH"`.

challengeResult

Defina como `true` se o usuário tiver concluído o desafio com êxito; do contrário, defina-o como `false`.

challengeMetadata

Seu nome para o desafio personalizado. Usado somente se `challengeName` for `"CUSTOM_CHALLENGE"`.

clientMetadata

Um ou mais pares de chave-valor que você pode fornecer como entrada personalizada para a função Lambda especificada para o acionador de criação do desafio de autenticação. Você pode usar o ClientMetadata parâmetro nas ações AdminRespondToAuthChallenge e da RespondToAuthChallengeAPI para passar esses dados para sua função Lambda. A solicitação que invoca a função create auth challenge não inclui dados transmitidos no ClientMetadata parâmetro AdminInitiateAuth e InitiateAuthnas operações da API.

Parâmetros de resposta de criar desafio de autenticação

publicChallengeParameters

Um ou mais pares de chave-valor do aplicativo cliente que serão usados no desafio a ser apresentado ao usuário. Este parâmetro deve conter todas as informações necessárias para apresentar com precisão o desafio ao usuário.

privateChallengeParameters

Esse parâmetro é usado somente pelo acionador do Lambda Verificar resposta do desafio de autenticação. Este parâmetro deve conter todas as informações necessárias para validar a resposta do usuário para o desafio. Em outras palavras, o parâmetro `publicChallengeParameters` contém a pergunta apresentada ao usuário, enquanto `privateChallengeParameters` contém as respostas válidas da pergunta.

challengeMetadata

Seu nome para o desafio personalizado, caso esse seja um desafio personalizado.

Exemplo de criar desafio de autenticação

Essa função tem dois desafios personalizados que correspondem à sequência de desafios em nosso exemplo de definição de desafio de autenticação. Os dois primeiros desafios são a autenticação SRP. Para o terceiro desafio, essa função retorna um URL CAPTCHA para sua aplicação na resposta do desafio. Sua aplicação renderiza o CAPTCHA no URL fornecido e retorna a entrada

do usuário. O URL da imagem CAPTCHA é adicionado aos parâmetros de desafio público como "`captchaUrl`", e a resposta esperado é adicionada aos parâmetros de desafio privado.

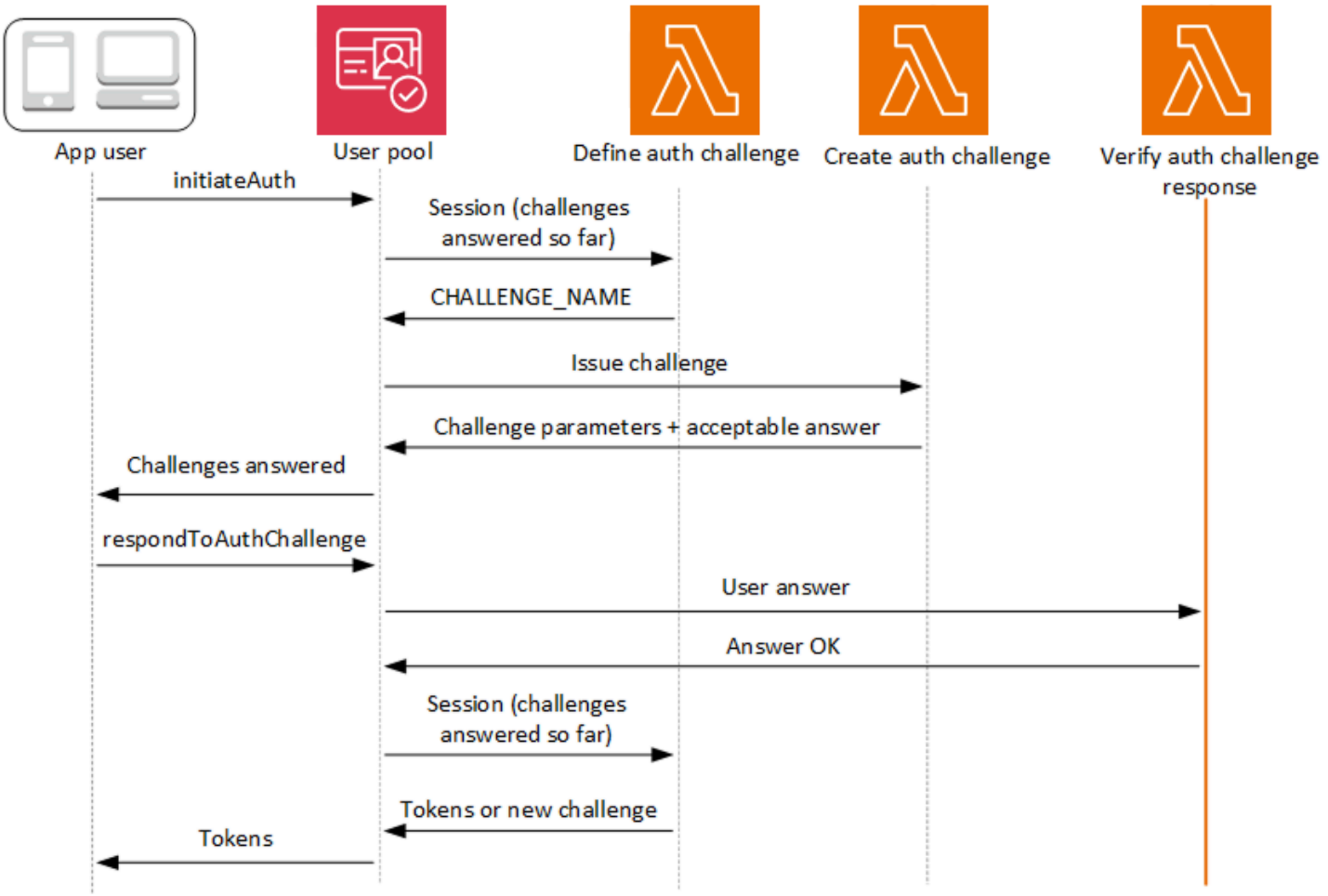
Para o quarto desafio, essa função retorna uma pergunta de segurança. Sua aplicação renderiza a pergunta e solicita que o usuário responda. Depois que os usuários resolverem os dois desafios personalizados, o acionador do de definição de desafio autenticação confirma que seu grupo de usuários pode emitir tokens.

Node.js

```
const handler = async (event) => {
  if (event.request.challengeName !== "CUSTOM_CHALLENGE") {
    return event;
  }

  if (event.request.session.length === 2) {
    event.response.publicChallengeParameters = {};
    event.response.privateChallengeParameters = {};
    event.response.publicChallengeParameters.captchaUrl = "url/123.jpg";
    event.response.privateChallengeParameters.answer = "5";
  }

  if (event.request.session.length === 3) {
    event.response.publicChallengeParameters = {};
    event.response.privateChallengeParameters = {};
    event.response.publicChallengeParameters.securityQuestion =
      "Who is your favorite team mascot?";
    event.response.privateChallengeParameters.answer = "Peccy";
  }

  return event;
};

export { handler };
```

Acionador do Lambda de verificar resposta do desafio de autenticação

O acionador do desafio verify auth é uma função do Lambda que compara a resposta fornecida pelo usuário com uma resposta conhecida. Essa função informa ao seu grupo de usuários se o usuário respondeu ao desafio corretamente. Quando o gatilho do desafio verify auth responde com um `answerCorrect` of`true`, a sequência de autenticação pode continuar.



Verificar a resposta do desafio de autenticação

> O Amazon Cognito invoca esse acionador para verificar se a resposta do usuário a um desafio de autenticação personalizado é válida ou não. Ele faz parte de um fluxo de autenticação personalizado do grupo de usuários.

A solicitação deste trigger contém os parâmetros `privateChallengeParameters` e `challengeAnswer`. O acionador do Lambda de criação de desafio de autenticação retorna valores `privateChallengeParameters` e contém a resposta esperada do usuário. O parâmetro `challengeAnswer` contém a resposta do usuário para o desafio.

A resposta contém o atributo `answerCorrect`. Se o usuário concluir o desafio com êxito, o Amazon Cognito definirá o valor do atributo como `true`. Se o usuário não concluir o desafio com êxito, o Amazon Cognito definirá o valor como `false`.

O loop de desafios se repetirá até que o usuário responda a todos os desafios.

Tópicos

Parâmetros do acionador do Lambda de verificar desafio de autenticação

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
            . . .
        },
        "privateChallengeParameters": {
            "string": "string",
            . . .
        },
        "challengeAnswer": "string",
        "clientMetadata": {
            "string": "string",
            . . .
        },
        "userNotFound": boolean
    },
    "response": {
        "answerCorrect": boolean
    }
}
```

Parâmetros de solicitação de verificar desafio de autenticação

userAttributes

Esse parâmetro contém um ou mais pares de nome-valor que representam atributos de usuário.

userNotFound

Quando o Amazon Cognito define `PreventUserExistenceErrors` como `ENABLED` para o cliente de grupo de usuários, ele preenche esse booleano.

privateChallengeParameters

Esse parâmetro vem do acionador de criação de desafio de autenticação. Para determinar se o usuário passou em um desafio, o Amazon Cognito compara os parâmetros com challengeAnswer do usuário.

Esse parâmetro contém todas as informações necessárias para validar a resposta do usuário para o desafio. Essas informações incluem a pergunta que o Amazon Cognito apresenta ao usuário (`publicChallengeParameters`) e as respostas válidas para a pergunta (`privateChallengeParameters`). Somente o acionador do Lambda de verificação da resposta do desafio de autenticação usa esse parâmetro.

challengeAnswer

Esse valor de parâmetro é a resposta do usuário para o desafio.

clientMetadata

Esse parâmetro contém um ou mais pares de chave-valor que você pode fornecer como entrada personalizada à função do Lambda para o acionador de verificação do desafio de autenticação. Para passar esses dados para sua função Lambda, use o ClientMetadata parâmetro nas operações AdminRespondToAuthChallengee da RespondToAuthChallengeAPI. O Amazon Cognito não inclui dados do ClientMetadata parâmetro AdminInitiateAuthe operações de InitiateAuthAPI na solicitação que ele passa para a função de desafio de verificação de autenticação.

Parâmetros de resposta de verificar desafio de autenticação

answerCorrect

Se o usuário concluir o desafio com êxito, o Amazon Cognito definirá esse parâmetro como `true`. Se o usuário não concluir o desafio com êxito, o Amazon Cognito definirá o parâmetro como `false`.

Exemplo de resposta de verificar desafio de autenticação

A função de verificação do desafio de autorização confere se a resposta do usuário a um desafio corresponde à resposta esperada. A resposta do usuário é definida pela entrada da sua aplicação, e a resposta preferencial é definida por `privateChallengeParameters.answer` na resposta da resposta do acionador criar desafio de autenticação. Tanto a resposta correta quanto a resposta dada fazem parte do evento de entrada para essa função.

Neste exemplo, se a resposta do usuário corresponder à resposta esperada, o Amazon Cognito definirá o parâmetro `answerCorrect` como `true`.

Node.js

```
const handler = async (event) => {
  if (
    event.request.privateChallengeParameters.answer ===
    event.request.challengeAnswer
  ) {
    event.response.answerCorrect = true;
  } else {
    event.response.answerCorrect = false;
  }

  return event;
};

export { handler };
```

Acionador do Lambda antes da geração do token

Como o Amazon Cognito invoca esse acionador antes da geração do token, é possível personalizar as declarações em tokens do grupo de usuários. Com os Recursos básicos da primeira versão ou `V1_0` do evento de acionamento de geração pré-token, é possível personalizar o token de identidade (ID). Em grupos de usuários com o plano de recursos Essentials ou Plus, você pode gerar a versão dois ou o evento `V2_0` acionador com personalização do token de acesso e a versão três ou o evento `V3_0` acionador com personalização do token de acesso para concessões de credenciais de cliente machine-to-machine (M2M).

O Amazon Cognito envia um evento `V1_0` como uma solicitação à sua função com dados que seriam gravados no token do ID. Um evento `V2_0` ou `V3_0` é uma solicitação única com os dados

que o Amazon Cognito gravaria nos tokens de identidade e de acesso. Para personalizar os dois tokens é necessário atualizar a função para usar a versão dois ou três do acionador e enviar dados aos dois tokens na mesma resposta.

O Amazon Cognito aplica respostas de eventos da versão dois aos tokens de acesso da autenticação do usuário, em que um usuário humano apresentou credenciais ao seu grupo de usuários. As respostas de eventos da versão três se aplicam aos tokens de acesso da autenticação do usuário e da autenticação da máquina, em que sistemas automatizados autorizam solicitações de token de acesso com segredos do cliente de aplicação. Além das circunstâncias dos tokens de acesso resultantes, os eventos das versões dois e três são idênticos.

Esse acionador do Lambda pode adicionar, remover e modificar algumas declarações em tokens de identidade e de acesso antes que o Amazon Cognito as emita para a aplicação. Para usar esse recurso, associe uma função do Lambda no console de grupos de usuários do Amazon Cognito ou atualize a `LambdaConfig` do grupo de usuários por meio da AWS Command Line Interface (AWS CLI).

Versões de eventos

Seu grupo de usuários pode fornecer diferentes versões de um evento de acionador de pré-geração de token para sua função do Lambda. Um acionador `V1_0` fornece os parâmetros para modificação dos tokens de ID. Um acionador `V2_0` ou `V3_0` fornece parâmetros para o seguinte:

1. As funções de um acionador `V1_0`.
2. A capacidade de personalizar os tokens de acesso.
3. A capacidade de transmitir tipos de dados complexos para valores de reivindicação de ID e token de acesso:
   - String
   - Número
   - Booleano
   - Conjuntos de strings, números, booleanos ou uma combinação de qualquer um desses
   - JSON

> **Note**
>
> No token de ID, você pode preencher objetos complexos com os valores das reivindicações, exceto para `phone_number_verified`, `email_verified`, `updated_at` e `address`.

Os grupos de usuários entregam eventos `V1_0` por padrão. Para configurar seu grupo de usuários para enviar um evento `V2_0`, escolha uma versão do evento de acionador da personalização de recursos básicos + token de acesso para identidades de usuários ao configurar seu acionador no console do Amazon Cognito. Para produzir eventos `V3_0`, escolha Recursos básicos + personalização de token de acesso para identidades de usuários e máquinas. Você também pode definir o valor de `LambdaVersion` nos LambdaConfigparâmetros em uma solicitação de CreateUserPool API UpdateUserPool ou de uma solicitação. As versões um, dois e três do evento estão disponíveis nos planos de recursos Essentials e Plus. As operações M2M para eventos da versão três têm uma estrutura de preços separada da fórmula de usuários ativos mensais (MAU). Para mais informações, consulte Preços do Amazon Cognito.

> **Note**
>
> Grupos de usuários que estavam operacionais com a opção Recursos de segurança avançados em ou antes de 22 de novembro de 2024 às 18:00 GMT e que permanecem no nível de recursos Lite têm acesso às versões um e dois do evento do acionador de pré-geração de tokens. Grupos de usuários nesse nível legado sem recursos avançados de segurança têm acesso à primeira versão do evento. A versão três está disponível somente no Essentials e Plus.

Referência de reivindicações e escopos

O Amazon Cognito limita as declarações e os escopos que você pode adicionar, modificar ou suprimir em tokens de acesso e identidade. A tabela a seguir descreve as declarações que sua função do Lambda pode ou não modificar e os parâmetros do evento de acionamento que afetam a presença ou o valor da reivindicação.

Reivindicar	Tipo de token padrão	Pode adicion?	Pode modific?	Pode suprimi	Parâmetro do evento: adicionar ou modificar	Parâmetro do evento: suprimir	Tipo de identidade	Versões do evento
Qualquer reivindicação que não esteja no esquema de token do grupo de usuários	Nenhur	Sim	Sim	N/D	`claimsTo/ ddOrOver: ide`	`claimsTo: uppress`	Usuário, máquina [1]	Tudo [2]
`scope`	Acessc	Sim	Sim	Sim	`scopesTo/ dd`	`scopesTo: uppress`	Usuário, máquina [1]	`v2_0`, `v3_0`
`cognito:g roups`	ID, Acessc	Sim	Sim	Sim	`groupsTo( verride`	`claimsTo: uppress`	Usuário	Tudo [2]
`cognito:p referred_ role`	ID	Sim	Sim	Sim	`preferre( Role`	`claimsTo: uppress` [3]	Usuário	Todos
`cognito:: oles`	ID	Sim	Sim	Sim	`iamRoles oOverrid(`	`claimsTo: uppress` [3]	Usuário	Todos
`cognito:u sername`	ID	Não	Não	Não	N/D	N/D	Usuário	N/D
Qualquer outra	Nenhur	Não	Não	Não	N/D	N/D	N/D	N/D

Reivindicar	Tipo de token padrão	Pode adicion?	Pode modific?	Pode suprimi	Parâmetro do evento: adicionar ou modificar	Parâmetro do evento: suprimir	Tipo de identidade	Versões do evento
reivindicação com um prefixo `cognito:`								
`username`	Acessc	Não	Não	Não	N/D	N/D	Usuário	`v2_0`, `v3_0`
`sub`	ID, Acessc	Não	Não	Não	N/D	N/D	Usuário	N/D
Atributo OIDC padrão	ID	Sim	Sim	Sim	`claimsTo/ ddOrOver ide`	`claimsTo uppress`	Usuário	Todos
Atributo `custom:`	ID	Sim	Sim	Sim	`claimsTo/ ddOrOver ide`	`claimsTo uppress`	Usuário	Todos
Atributo `dev:`	ID	Não	Não	Sim	N/D	`claimsTo uppress`	Usuário	Todos
`identiti s`	ID	Não	Não	Não	N/D	N/D	Usuário	N/D
`aud`[4]	ID	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`client_i`	Acessc	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D

Reivindicar	Tipo de token padrão	Pode adicion?	Pode modific?	Pode suprimi	Parâmetro do evento: adicionar ou modificar	Parâmetro do evento: suprimir	Tipo de identidade	Versões do evento
`event_id`	Acessc	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`device_k` `y`	Acessc	Não	Não	Não	N/D	N/D	Usuário	N/D
`version`	Acessc	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`acr`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`amr`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`at_hash`	ID	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`auth_time`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`azp`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`exp`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`iat`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`iss`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D

Reivindicar	Tipo de token padrão	Pode adicion?	Pode modific?	Pode suprimi	Parâmetro do evento: adicionar ou modificar	Parâmetro do evento: suprimir	Tipo de identidade	Versões do evento
`jti`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`nbf`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`nonce`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`origin_ji`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D
`token_us`	ID, acesso	Não	Não	Não	N/D	N/D	Usuário, máquina	N/D

[1] Os tokens de acesso para identidades de máquinas só estão disponíveis com o evento de entrada do acionador `v3_0`. A versão três do evento só está disponível nos níveis de recursos Essentials e Plus. Grupos de usuários no nível Lite podem receber eventos `v1_0`. Grupos de usuários no nível Lite com recursos avançados de segurança podem receber eventos `v1_0` e `v2_0`.

[2] Configure seu acionador de pré-geração de tokens para a versão do evento `v1_0` somente para token de ID, `v2_0` para token de ID e de acesso, `v3_0` para token de ID e de acesso com recursos para identidades de máquina.

[3] Para suprimir as reivindicações `cognito:preferred_role` e `cognito:roles`, adicione `cognito:groups` a `claimsToSuppress`.

[4] É possível adicionar uma reivindicação `aud` aos tokens de acesso, mas o valor deve corresponder ao ID do cliente da aplicação da sessão atual. É possível gerar o ID do cliente no evento de solicitação de `event.callerContext.clientId`.

Personalizar o token de identidade

Com todas as versões de eventos do acionador do Lambda de pré-geração de tokens, é possível personalizar o conteúdo de um token de identidade (ID) do grupo de usuários. O token de ID fornece atributos de usuário de uma fonte de identidade confiável para login em uma aplicação web ou móvel. Para obter mais informações sobre tokens, consulte Como entender o token de identidade (ID).

Os usos do gatilho do Lambda de pré-geração de tokens com um token de ID incluem os seguintes:

- Fazer uma alteração em runtime no perfil do IAM que o usuário solicita de um banco de identidades.
- Adicionar atributos do usuário de uma fonte externa.
- Adicionar ou substituir valores de atributos de usuário existentes.
- Suprimir a divulgação de atributos do usuário que, devido aos escopos autorizados do usuário e ao acesso de leitura aos atributos concedido ao cliente da aplicação, seriam transmitidos à aplicação.

Personalizar o token de acesso

Com as versões dois e três de eventos do acionador do Lambda de pré-geração de tokens, é possível personalizar o conteúdo de um token de acesso do grupo de usuários. O token de acesso autoriza os usuários a recuperar informações de recursos protegidos por acesso, como operações de API autorizadas por tokens do Amazon Cognito e de terceiros. APIs Para autorização machine-to-machine (M2M) com concessão de credenciais de cliente, o Amazon Cognito só invoca o gatilho de pré-geração de token quando seu grupo de usuários está configurado para um evento da versão três (). `V3_0` Para obter mais informações sobre tokens de acesso, consulte Como entender o token de acesso.

Os usos do gatilho do Lambda de pré-geração de tokens com um token de acesso incluem os seguintes:

- Adicionar ou suprimir os escopos na reivindicação `scope`. Por exemplo, é possível adicionar escopos a um token de acesso gerado pela autenticação da API de grupos de usuários do Amazon Cognito, que atribui apenas o escopo `aws.cognito.signin.user.admin`.
- Alterar a associação de um usuário em grupos de usuários.
- Adicione declarações que ainda não estão presentes em um token de acesso do Amazon Cognito.

- Suprimir a divulgação de declarações que, de outra forma, seriam transmitidas à aplicação.

Para oferecer compatibilidade com a personalização do acesso no grupo de usuários, é necessário configurar o grupo de usuários para gerar uma versão atualizada da solicitação de gatilho. Atualize o grupo de usuários conforme mostrado no procedimento a seguir.

Console de gerenciamento da AWS

Como oferecer compatibilidade com a personalização do token de acesso em um gatilho do Lambda de pré-geração do tokens

1. Acesse o console do Amazon Cognito e escolha User Pools (Grupos de usuários).
2. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
3. Selecione o menu Extensões e localize os Acionadores do Lambda.
4. Adicione ou edite um Acionador de geração de pré-token.
5. Selecione uma função do Lambda em Atribuir função do Lambda.
6. Escolha uma Versão do evento do acionador de Recursos básicos + personalização do token de acesso para identidades de usuário ou Recursos básicos + personalização do token de acesso para identidades de usuário e de máquinas. Essa configuração atualiza os parâmetros de solicitação que o Amazon Cognito envia à função para incluir campos para personalização do token de acesso.

User pools API

Como oferecer compatibilidade com a personalização do token de acesso em um gatilho do Lambda de pré-geração do tokens

Gere uma solicitação de UpdateUserPoolAPI CreateUserPoolou. Você deve especificar um valor para todos os parâmetros que não deseja definir como padrão. Para obter mais informações, consulte Como atualizar a configuração do grupo de usuários e do cliente da aplicação.

Inclua o conteúdo a seguir no parâmetro `LambdaVersion` da solicitação. Um valor `LambdaVersion` de `V2_0` faz com que o grupo de usuários adicione parâmetros para tokens de acesso, além de aplicar alterações a eles. Um valor `LambdaVersion` de `V3_0` produz o mesmo evento que `V2_0`, mas faz com que seu grupo de usuários também aplique alterações aos tokens de acesso M2M. Para invocar uma versão de função específica, use o ARN de uma função do Lambda com uma versão da função como o valor de `LambdaArn`.

```
"PreTokenGenerationConfig": {
   "LambdaArn": "arn:aws:lambda:us-west-2:123456789012:function:MyFunction",
   "LambdaVersion": "V3_0"
},
```

Metadados do cliente para credenciais do cliente machine-to-machine (M2M)

Você pode transmitir metadados do cliente em solicitações de M2M. Os metadados do cliente são informações adicionais de um usuário ou ambiente de aplicação que podem contribuir para os resultados de um Acionador do Lambda antes da geração do token. Em operações de autenticação com um usuário principal, você pode passar metadados do cliente para o gatilho de pré-geração de token no corpo das solicitações AdminRespondToAuthChallengede RespondToAuthChallengeAPI. Como as aplicações conduzem o fluxo de geração de tokens de acesso para M2M com solicitações diretas ao Endpoint de token, elas têm um modelo diferente. No corpo POST das solicitações de token para credenciais do cliente, transmita um parâmetro `aws_client_metadata` com o objeto de metadados do cliente codificado em URL (`x-www-form-urlencoded`) para string. Para ver um exemplo de solicitação, consulte Credenciais do cliente com autorização básica. Veja a seguir um exemplo de parâmetro que transmite os pares de chave-valor `{"environment": "dev", "language": "en-US"}`.

```
aws_client_metadata=%7B%22environment%22%3A%20%22dev%22,%20%22language%22%3A%20%22en-US%22%7D
```

Mais atributos

- Como personalizar tokens de acesso nos grupos de usuários do Amazon Cognito

Tópicos

- Fontes do acionador do Lambda antes da geração do token
- Parâmetros do acionador do Lambda antes da geração do token
- Exemplo da segunda versão do evento de acionamento pré-token: adicionar e suprimir declarações, escopos e grupos
- Exemplo de evento de geração pré-token da versão dois: adicionar reivindicações com objetos complexos
- Exemplo da primeira versão do evento de geração pré-token: adicionar uma nova declaração e suprimir uma declaração existente

- Exemplo da primeira versão do evento de geração pré-token: modificar a associação do grupo do usuário

Fontes do acionador do Lambda antes da geração do token

Valor de triggerSource	Event
`TokenGeneration_HostedAuth`	Chamado durante a autenticação na página de login do login gerenciado no Amazon Cognito.
`TokenGeneration_Authentication`	Chamado depois de os fluxos de autenticação de usuário concluírem.
`TokenGeneration_NewPasswordChallenge`	Chamado após o usuário ser criado por um admin. Este fluxo é chamado quando o usuário tiver que alterar uma senha temporária.
`TokenGeneration_ClientCredentials`	Chamado após a concessão de credenciais de um cliente M2M. Seu grupo de usuários só envia esse evento quando sua versão do evento é `V3_0`.
`TokenGeneration_AuthenticateDevice`	Chamado no final da autenticação do dispositivo de um usuário.
`TokenGeneration_RefreshTokens`	Chamado quando um usuário tenta atualizar a identidade e acessar tokens.

Parâmetros do acionador do Lambda antes da geração do token

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações. Ao adicionar um gatilho do Lambda de pré-geração de tokens ao grupo de usuários, é possível selecionar uma versão do gatilho. Essa versão determina se o Amazon Cognito transmite uma solicitação para a função do Lambda com parâmetros adicionais para personalização do token de acesso.

Version one

O token da versão um pode definir a associação ao grupo, perfis do IAM e novas reivindicações em tokens de ID. As substituições de associação ao grupo também se aplicam à reivindicação `cognito:groups` em tokens de acesso.

```
{
    "request": {
        "userAttributes": {"string": "string"},
        "groupConfiguration": {
                "groupsToOverride": [
                    "string",
                    "string"
                ],
                "iamRolesToOverride": [
                    "string",
                    "string"
                ],
                "preferredRole": "string"
        },
        "clientMetadata": {"string": "string"}
    },
    "response": {
        "claimsOverrideDetails": {
            "claimsToAddOrOverride": {"string": "string"},
            "claimsToSuppress": [
                "string",
                "string"
            ],
            "groupOverrideDetails": {
                "groupsToOverride": [
                    "string",
                    "string"
                ],
                "iamRolesToOverride": [
                    "string",
                    "string"
                ],
                "preferredRole": "string"
            }
        }
    }
}
```

Versions two and three

Os eventos de solicitação das versões dois e três adicionam campos que personalizam o token de acesso. Os grupos de usuários aplicam alterações dos eventos da versão três aos tokens de acesso para identidades de máquinas. Essas versões também adicionam suporte para tipos de dados `claimsToOverride` complexos no objeto de resposta. Sua função do Lambda pode retornar os seguintes tipos de dados no valor de `claimsToOverride`:

- String
- Número
- Booleano
- Conjuntos de strings, números, booleanos ou uma combinação de qualquer um desses
- JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string"
        },
        "scopes": ["string", "string"],
        "groupConfiguration": {
            "groupsToOverride": ["string", "string"],
            "iamRolesToOverride": ["string", "string"],
            "preferredRole": "string"
        },
        "clientMetadata": {
            "string": "string"
        }
    },
    "response": {
        "claimsAndScopeOverrideDetails": {
            "idTokenGeneration": {
                "claimsToAddOrOverride": {
                    "string": [accepted datatype]
                },
                "claimsToSuppress": ["string", "string"]
            },
            "accessTokenGeneration": {
                "claimsToAddOrOverride": {
                    "string": [accepted datatype]
```

```
                },
                "claimsToSuppress": ["string", "string"],
                "scopesToAdd": ["string", "string"],
                "scopesToSuppress": ["string", "string"]
            },
            "groupOverrideDetails": {
                "groupsToOverride": ["string", "string"],
                "iamRolesToOverride": ["string", "string"],
                "preferredRole": "string"
            }
        }
    }
}
```

Parâmetros de solicitação antes da geração do token

Name (Nome)	Description	Versão mínima do evento de gatilho
userAttributes	Os atributos do seu perfil de usuário no grupo de usuários.	1
groupConfiguration	O objeto de entrada que contém a configuração atual do grupo. O objeto inclui `groupsToOverride` , `iamRolesToOverride` e `preferredRole` .	1
groupsToOverride	Os grupos de usuários dos quais seu usuário é membro.	1
iamRolesToSubstituir	Você pode associar um grupo de grupos de usuários a uma função AWS Identity and Access Management (IAM). Esse elemento é uma lista de todos os perfis do IAM dos grupos dos quais seu usuário é membro.	1
preferredRole	É possível definir uma precedência para grupos de usuários. Esse elemento contém o nome do perfil do IAM do grupo com a maior precedência no elemento `groupsToOverride` .	1

Name (Nome)	Description	Versão mínima do evento de gatilho
clientMetadata	Um ou mais pares de chave-valor que você pode especificar e fornecer como entrada personalizada à função do Lambda para o acionador antes da geração do token. Para passar esses dados para sua função Lambda, use o ClientMetadata parâmetro nas operações AdminRespondToAuthChallenge e da RespondToAuthChallengeAPI. O Amazon Cognito não inclui dados do `ClientMetadata` parâmetro AdminInitiateAuthe operações de InitiateAuthAPI na solicitação que ele passa para a função de pré-geração de token.	1
escopos	Escopos de token de acesso. Os escopos presentes em um token de acesso são os escopos padrão e personalizados do grupo de usuários que o usuário solicitou e que você autorizou o cliente da aplicação a emitir.	2

Parâmetros de resposta antes da geração do token

Name (Nome)	Description	Versão mínima do evento de gatilho
claimsOverrideDetails	Um contêiner para todos os elementos em um evento de acionamento `V1_0`.	1
claimsAndScopeOverrideDetails	Um contêiner para todos os elementos em um evento de acionamento `V2_0` ou `V3_0`.	2
idTokenGeneration	As declarações que você deseja substituir, adicionar ou suprimir no token de ID do usuário. Esses valores de	2

Name (Nome)	Description	Versão mínima do evento de gatilho
	personalização do token pai para ID aparecem somente na versão 2 e posteriores do evento, mas os elementos filhos aparecem nos eventos da versão 1.	
accessTokenGeneration	As declarações e os escopos que você deseja substituir, adicionar ou suprimir no token de acesso do usuário. Esse pai dos valores de personalização do token de acesso aparece somente na versão 2 e posteriores do evento.	2
claimsToAddOrOverride	Um mapa de uma ou mais declarações e os respectivos valores que você deseja adicionar ou modificar. Para declarações relacionadas a grupos, use `groupOverrideDetails` . Na versão 2 e posteriores do evento, esse elemento aparece em `accessTokenGeneration` e `idTokenGeneration` .	1 *
claimsToSuppress	Uma lista de declarações que o Amazon Cognito deve suprimir. Se sua função suprime e substitui um valor de solicitação, o Amazon Cognito suprime a solicitação. Na versão 2 e posteriores do evento, esse elemento aparece em `accessTokenGeneration` e `idTokenGeneration` .	1

Name (Nome)	Description	Versão mínima do evento de gatilho
groupOverrideDetails	O objeto de saída que contém a configuração atual do grupo. O objeto inclui `groupsToOverride` , `iamRolesToOverride` e `preferredRole` . A função substitui o objeto `groupOverrideDetails` pelo objeto fornecido. Se você fornecer um objeto nulo ou vazio na resposta, o Amazon Cognito suprimirá os grupos. Para manter a mesma configuração de grupo existente, copie o valor do objeto `groupConfiguration` da solicitação no objeto `groupOverrideDetails` na resposta. Depois, transmita-o de volta para o serviço. O ID do Amazon Cognito e os tokens de acesso contêm a declaração `cognito:groups` . O objeto `groupOverrideDetails` substitui a declaração `cognito:groups` em tokens de acesso e em tokens de ID. As substituições de grupo são as únicas alterações ao token de acesso que os eventos da versão 1 podem fazer.	1
scopesToAdd	Uma lista de escopos que você deseja adicionar à reivindicação `scope` no token de acesso do usuário. Não é possível adicionar valores de escopo que contenham um ou mais caracteres de espaço em branco.	2
scopesToSuppress	Uma lista de escopos que você deseja remover da reivindicação `scope` no token de acesso do usuário.	2

* Objetos de resposta aos eventos da versão um podem retornar strings. Objetos de resposta aos eventos das versões dois e três podem retornar objetos complexos.

Exemplo da segunda versão do evento de acionamento pré-token: adicionar e suprimir declarações, escopos e grupos

Este exemplo faz as seguintes modificações nos tokens de um usuário.

1. Define `family_name` como `Doe` no token de ID.
2. Impede que as declarações `email` e `phone_number` apareçam no token de ID.
3. Define a declaração `cognito:roles` do token de ID como `"arn:aws:iam::123456789012:role\/sns_callerA","arn:aws:iam::123456789012:role\/sns_callerC","arn:aws:iam::123456789012:role\/sns_callerB"`.
4. Define a declaração `cognito:preferred_role` do token de ID como `arn:aws:iam::123456789012:role/sns_caller`.
5. Adiciona os escopos `openid`, `email` e `solar-system-data/asteroids.add` ao token de acesso.
6. Suprime o escopo `phone_number` e `aws.cognito.signin.user.admin` do token de acesso. A remoção de `phone_number` impede a recuperação do número de telefone do usuário em `userInfo`. A remoção de `aws.cognito.signin.user.admin` impede que as solicitações de API pelo usuário leiam e modifiquem seu próprio perfil com a API de grupos de usuários do Amazon Cognito.

   > **Note**
   >
   > A remoção de `phone_number` dos escopos só impedirá a recuperação do número de telefone de um usuário se os escopos restantes no token de acesso incluírem `openid` e pelo menos mais um escopo padrão. Para obter mais informações, consulte Sobre escopos.

7. Define a declaração `cognito:groups` do token de ID e de acesso como `"new-group-A","new-group-B","new-group-C"`.

JavaScript

```
export const handler = function(event, context) {
  event.response = {
    "claimsAndScopeOverrideDetails": {
      "idTokenGeneration": {
```

```
        "claimsToAddOrOverride": {
          "family_name": "Doe"
        },
        "claimsToSuppress": [
          "email",
          "phone_number"
        ]
      },
      "accessTokenGeneration": {
        "scopesToAdd": [
          "openid",
          "email",
          "solar-system-data/asteroids.add"
        ],
        "scopesToSuppress": [
          "phone_number",
          "aws.cognito.signin.user.admin"
        ]
      },
      "groupOverrideDetails": {
        "groupsToOverride": [
          "new-group-A",
          "new-group-B",
          "new-group-C"
        ],
        "iamRolesToOverride": [
          "arn:aws:iam::123456789012:role/new_roleA",
          "arn:aws:iam::123456789012:role/new_roleB",
          "arn:aws:iam::123456789012:role/new_roleC"
        ],
        "preferredRole": "arn:aws:iam::123456789012:role/new_role",
      }
    }
  };
  // Return to Amazon Cognito
  context.done(null, event);
};
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
    "version": "2",
    "triggerSource": "TokenGeneration_Authentication",
    "region": "us-east-1",
    "userPoolId": "us-east-1_EXAMPLE",
    "userName": "JaneDoe",
    "callerContext": {
        "awsSdkVersion": "aws-sdk-unknown-unknown",
        "clientId": "1example23456789"
    },
    "request": {
        "userAttributes": {
            "sub": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "cognito:user_status": "CONFIRMED",
            "email_verified": "true",
            "phone_number_verified": "true",
            "phone_number": "+12065551212",
            "family_name": "Zoe",
            "email": "Jane.Doe@example.com"
        },
        "groupConfiguration": {
            "groupsToOverride": ["group-1", "group-2", "group-3"],
            "iamRolesToOverride": ["arn:aws:iam::123456789012:role/sns_caller1",
 "arn:aws:iam::123456789012:role/sns_caller2", "arn:aws:iam::123456789012:role/
sns_caller3"],
            "preferredRole": ["arn:aws:iam::123456789012:role/sns_caller"]
        },
        "scopes": [
            "aws.cognito.signin.user.admin", "openid", "email", "phone"
        ]
    },
    "response": {
        "claimsAndScopeOverrideDetails": []
    }
}
```

Exemplo de evento de geração pré-token da versão dois: adicionar reivindicações com objetos complexos

Este exemplo faz as seguintes modificações nos tokens de um usuário.

1. Adiciona reivindicações dos tipos número, string, booleano e JSON ao token de ID. Essa é a única alteração que os eventos de acionador da versão dois disponibilizam para o token de ID.
2. Adiciona reivindicações dos tipos número, string, booleano e JSON ao token de acesso.
3. Adiciona três escopos ao token de acesso.
4. Suprime a reivindicação `email` nos tokens de ID e acesso.
5. Suprime o escopo `aws.cognito.signin.user.admin` no token de acesso.

JavaScript

```
export const handler = function(event, context) {

    var scopes = ["MyAPI.read", "MyAPI.write", "MyAPI.admin"]
    var claims = {}
    claims["aud"]= event.callerContext.clientId;
    claims["booleanTest"] = false;
    claims["longTest"] = 9223372036854775807;
    claims["exponentTest"] = 1.7976931348623157E308;
    claims["ArrayTest"] = ["test", 9223372036854775807, 1.7976931348623157E308,
 true];
    claims["longStringTest"] = "\{\
        \"first_json_block\": \{\
            \"key_A\": \"value_A\",\
            \"key_B\": \"value_B\"\
        \},\
        \"second_json_block\": \{\
            \"key_C\": \{\
                \"subkey_D\": [\
                    \"value_D\",\
                    \"value_E\"\
                ],\
                \"subkey_F\": \"value_F\"\
            \},\
            \"key_G\": \"value_G\"\
        \}\
    \}";
    claims["jsonTest"] = {
     "first_json_block": {
      "key_A": "value_A",
      "key_B": "value_B"
     },
     "second_json_block": {
```

```
      "key_C": {
       "subkey_D": [
        "value_D",
        "value_E"
       ],
       "subkey_F": "value_F"
      },
      "key_G": "value_G"
     }
    };
    event.response = {
        "claimsAndScopeOverrideDetails": {
            "idTokenGeneration": {
                "claimsToAddOrOverride": claims,
                "claimsToSuppress": ["email"]
            },
            "accessTokenGeneration": {
                "claimsToAddOrOverride": claims,
                "claimsToSuppress": ["email"],
                "scopesToAdd": scopes,
                "scopesToSuppress": ["aws.cognito.signin.user.admin"]
            }
        }
    };
    console.info("EVENT response\n" + JSON.stringify(event, (_, v) => typeof v ===
 'bigint' ? v.toString() : v, 2))
    console.info("EVENT response size\n" + JSON.stringify(event, (_, v) => typeof v
 === 'bigint' ? v.toString() : v).length)
    // Return to Amazon Cognito
    context.done(null, event);
};
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
    "version": "2",
    "triggerSource": "TokenGeneration_HostedAuth",
```

```
    "region": "us-west-2",
    "userPoolId": "us-west-2_EXAMPLE",
    "userName": "JaneDoe",
    "callerContext": {
        "awsSdkVersion": "aws-sdk-unknown-unknown",
        "clientId": "1example23456789"
    },
    "request": {
        "userAttributes": {
            "sub": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "cognito:user_status": "CONFIRMED"
            "email_verified": "true",
            "phone_number_verified": "true",
            "phone_number": "+12065551212",
            "email": "Jane.Doe@example.com"
        },
        "groupConfiguration": {
            "groupsToOverride": ["group-1", "group-2", "group-3"],
            "iamRolesToOverride": ["arn:aws:iam::123456789012:role/sns_caller1"],
            "preferredRole": ["arn:aws:iam::123456789012:role/sns_caller1"]
        },
        "scopes": [
            "aws.cognito.signin.user.admin",
            "phone",
            "openid",
            "profile",
            "email"
        ]
    },
    "response": {
        "claimsAndScopeOverrideDetails": []
    }
}
```

Exemplo da primeira versão do evento de geração pré-token: adicionar uma nova declaração e suprimir uma declaração existente

Esse exemplo usa um evento de gatilho 1 de versão com uma função do Lambda de pré-geração de tokens para adicionar uma nova declaração e suprimir uma existente.

Node.js

```
const handler = async (event) => {
  event.response = {
    claimsOverrideDetails: {
      claimsToAddOrOverride: {
        my_first_attribute: "first_value",
        my_second_attribute: "second_value",
      },
      claimsToSuppress: ["email"],
    },
  };

  return event;
};

export { handler };
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código: como o código de exemplo não processa nenhum parâmetro de solicitação, você pode usar um evento de teste com uma solicitação vazia. Para obter mais informações sobre parâmetros de solicitação comuns, consulte Evento de acionador do Lambda do grupo de usuários.

JSON

```
{
  "request": {},
  "response": {}
}
```

Exemplo da primeira versão do evento de geração pré-token: modificar a associação do grupo do usuário

Esse exemplo usa o evento de gatilho 1 de versão com uma função do Lambda de pré-geração de tokens para modificar a associação do grupo do usuário.

Node.js

```
const handler = async (event) => {
  event.response = {
    claimsOverrideDetails: {
      groupOverrideDetails: {
        groupsToOverride: ["group-A", "group-B", "group-C"],
        iamRolesToOverride: [
          "arn:aws:iam::XXXXXXXXXXX:role/sns_callerA",
          "arn:aws:iam::XXXXXXXXX:role/sns_callerB",
          "arn:aws:iam::XXXXXXXXXX:role/sns_callerC",
        ],
        preferredRole: "arn:aws:iam::XXXXXXXXXXX:role/sns_caller",
      },
    },
  };

  return event;
};

export { handler };
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
  "request": {},
  "response": {}
}
```

Migrar o acionador do Lambda do usuário

Quando um usuário não existir no grupo de usuários no momento de login com senha, ou no fluxo de senha esquecida, o Amazon Cognito invocará esse acionador. Depois que a função Lambda retornar com êxito, o Amazon Cognito criará o usuário no grupo de usuários. Para obter detalhes sobre o

fluxo de autenticação com o acionador do Lambda de migração de usuários, consulte Como importar usuários com um acionador do Lambda de migração de usuários.

Para migrar os usuários de seu diretório de usuários existente para grupos de usuários do Amazon Cognito no momento do login ou durante o fluxo de senha esquecida, siga esse acionador do Lambda.

Tópicos

- Fontes do acionador do Lambda de migrar usuário
- Parâmetros do acionador do Lambda de migrar usuário
- Exemplo: migrar um usuário com uma senha existente

Fontes do acionador do Lambda de migrar usuário

Valor de triggerSource	Event
`UserMigration_Authentication` [1]	Migração de usuários no login.
`UserMigration_ForgotPassword`	Migração de usuários durante o fluxo de esquecimento de senha.

[1] O Amazon Cognito não invoca esse acionador quando os usuários autenticam com login sem senha.

Parâmetros do acionador do Lambda de migrar usuário

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "userName": "string",
    "request": {
        "password": "string",
        "validationData": {
            "string": "string",
```

```
            . . .
        },
        "clientMetadata": {
            "string": "string",
       . . .
        }
    },
    "response": {
        "userAttributes": {
            "string": "string",
            . . .
        },
        "finalUserStatus": "string",
        "messageAction": "string",
        "desiredDeliveryMediums": [ "string", . . .],
        "forceAliasCreation": boolean,
        "enableSMSMFA": boolean
    }
}
```

Parâmetros de solicitação de migrar usuário

userName

O nome de usuário que o usuário insere no login.

password

A senha que o usuário insere no login. O Amazon Cognito não envia esse valor em uma solicitação iniciada por um fluxo de senha esquecida.

validationData

Um ou mais pares de chave-valor que contêm os dados de validação na solicitação de login do usuário. Para passar esses dados para sua função Lambda, você pode usar o ClientMetadata parâmetro nas ações InitiateAuthe da AdminInitiateAuthAPI.

clientMetadata

Um ou mais pares de chave-valor que você pode fornecer como entrada personalizada à função do Lambda para o acionador de migração do usuário. Para passar esses dados para sua função Lambda, você pode usar o ClientMetadata parâmetro nas ações AdminRespondToAuthChallengee da ForgotPasswordAPI.

Parâmetros de resposta de migrar usuário

userAttributes

Este campo é obrigatório.

Esse campo deve conter um ou mais pares de nome-valor que o Amazon Cognito armazena no perfil de usuário no grupo de usuários e usa como atributos de usuário. Você pode incluir atributos de usuário padrão e personalizados. Os atributos personalizados exigem o prefixo `custom:` para diferenciá-los dos atributos padrão. Para obter mais informações, consulte Atributos personalizados.

> **Note**
>
> Para redefinir uma senha no fluxo de senha esquecida, o usuário deverá ter um e-mail verificado ou um número de telefone verificado. O Amazon Cognito envia uma mensagem contendo um código de redefinição de senha para o e-mail ou o número de telefone nos atributos do usuário.

Atributos	Requisito
Todos os atributos marcados como necessários quando o grupo de usuários foi criado	Se os atributos necessários estiverem ausentes durante a migração, o Amazon Cognito usará valores padrão.
`username`	Obrigatório se você tiver configurado o grupo de usuários com atributos de alias além do nome de usuário para login e se o usuário tiver inserido um alias válido como nome de usuário. Esse valor de alias pode ser um endereço de e-mail, nome de usuário preferido ou número de telefone. Se a solicitação e o grupo de usuários atenderem aos requisito s de alias, a resposta de sua função deverá atribuir o parâmetro `username` recebido para um atributo alias. Além disso, a resposta deve atribuir seu próprio valor ao atributo `username`. Se o grupo

Atributos	Requisito
	de usuários não atender às condições necessárias para mapear o `username` recebido para um alias, o parâmetro `username` na resposta deverá corresponder exatamente à solicitação ou ser omitido. **Note** `username` deve ser exclusivo no grupo de usuários.

finalUserStatus

Você pode definir esse parâmetro como `CONFIRMED` para confirmar automaticamente seus usuários para que eles possam fazer login com a senha anterior. Quando você define um usuário como `CONFIRMED`, ele não precisa fazer nada além para fazer login. Se você não definir esse atributo como `CONFIRMED`, ele será definido como `RESET_REQUIRED`.

Um `finalUserStatus` do `RESET_REQUIRED` significa que o usuário deve alterar a senha imediatamente após a migração no login, e sua aplicação cliente deve processar o `PasswordResetRequiredException` durante o fluxo de autenticação.

> **Note**
>
> O Amazon Cognito não impõe a política de intensidade de senha que você configurou para o grupo de usuários durante a migração usando o acionador do Lambda. Se a senha não atender à respectiva política que você configurou, o Amazon Cognito ainda assim a aceitará para que ela possa continuar a migrar o usuário. Para impor a política de intensidade da senha e rejeitar senhas que não atendam à política, valide a intensidade da senha no seu código. Em seguida, se a senha não atender à política, finalUserStatus defina como`RESET_REQUIRED`.

messageAction

Você pode definir esse parâmetro como `SUPPRESS` para se recusar a enviar a mensagem de boas-vindas que o Amazon Cognito geralmente envia Amazon novos usuários. Se sua função não retornar esse parâmetro, o Amazon Cognito enviará a mensagem de boas-vindas.

desiredDeliveryMediums

Esse parâmetro pode ser definido como `EMAIL` para enviar a mensagem de boas-vindas por e-mail ou como `SMS` para enviar a mensagem de boas-vindas por SMS. Se sua função não retornar esse parâmetro, o Amazon Cognito enviará a mensagem de boas-vindas por SMS.

forceAliasCreation

Se você definir esse parâmetro como `TRUE` e o número de telefone ou endereço de e-mail no UserAttributes parâmetro já existir como um alias com um usuário diferente, a chamada da API migrará o alias do usuário anterior para o usuário recém-criado. O usuário anterior não pode mais fazer login usando esse alias.

Se você definir esse parâmetro como `FALSE` e o alias existir, o Amazon Cognito não migrará o usuário e retornará um erro para a aplicação cliente.

Se você não retornar esse parâmetro, o Amazon Cognito assumirá que seu valor é “false”.

enableSMSMFA

Defina esse parâmetro como `true` para exigir que o usuário migrado conclua a autenticação multifator (MFA) por mensagem de texto SMS para fazer login. Seu grupo de usuários deve ter a MFA habilitada. Os atributos do usuário nos parâmetros da solicitação devem incluir um número de telefone; do contrário, a migração desse usuário falhará.

Exemplo: migrar um usuário com uma senha existente

Esse exemplo de função Lambda migra o usuário com uma senha existente e suprime a mensagem de boas-vindas do Amazon Cognito.

Node.js

```
exports.handler = (event, context, callback) => {
  var user;

  if (event.triggerSource == "UserMigration_Authentication") {
    // authenticate the user with your existing user directory service
    user = authenticateUser(event.userName, event.request.password);
    if (user) {
      event.response.userAttributes = {
        email: user.emailAddress,
        email_verified: "true",
```

```
      };
      event.response.finalUserStatus = "CONFIRMED";
      event.response.messageAction = "SUPPRESS";
      context.succeed(event);
    } else {
      // Return error to Amazon Cognito
      callback("Bad password");
    }
  } else if (event.triggerSource == "UserMigration_ForgotPassword") {
    // Lookup the user in your existing user directory service
    user = lookupUser(event.userName);
    if (user) {
      event.response.userAttributes = {
        email: user.emailAddress,
        // required to enable password-reset code to be sent to user
        email_verified: "true",
      };
      event.response.messageAction = "SUPPRESS";
      context.succeed(event);
    } else {
      // Return error to Amazon Cognito
      callback("Bad password");
    }
  } else {
    // Return error to Amazon Cognito
    callback("Bad triggerSource " + event.triggerSource);
  }
};
```

Acionador do Lambda de mensagem personalizada

Quando você tiver um padrão externo para as mensagens de e-mail e SMS que deseja enviar aos seus usuários, ou quando quiser aplicar sua própria lógica em tempo de execução à formatação das mensagens do usuário, adicione um acionador de mensagem personalizada ao grupo de usuários. A mensagem personalizada do Lambda recebe o conteúdo de todas as mensagens de e-mail e SMS antes que seu grupo de usuários as envie. Sua função do Lambda então tem a oportunidade de modificar o conteúdo e o assunto da mensagem.

O Amazon Cognito invoca esse acionador antes de enviar um e-mail, uma mensagem de verificação de telefone ou um código de autenticação multifator (MFA). Você pode personalizar a mensagem dinamicamente com o acionador de mensagem personalizado.

A solicitação inclui `codeParameter`. Essa string funciona como espaço reservado no código que o Amazon Cognito fornece ao usuário. Insira a string `codeParameter` no corpo da mensagem, na posição em que você deseja que o código de verificação apareça. Quando o Amazon Cognito recebe essa resposta, ele substitui a string `codeParameter` pelo código de verificação real.

> **Note**
>
> O evento de entrada de uma função do Lambda de mensagem personalizada com o acionador `CustomMessage_AdminCreateUser` retorna um nome de usuário e um código de verificação. Como um usuário criado pelo administrador deve receber tanto o nome de usuário quanto o código, a resposta da função deve incluir ambos variáveis de espaços reservados para o nome de usuário e o código. Os espaços reservados para sua mensagem são os valores de `request.usernameParameter` e `request.codeParameter`. Esses valores são normalmente `{username}` e `{####}`. Como prática recomendada, referenciam os valores de entrada em vez de codificar rigidamente os nomes das variáveis.

Tópicos

Fontes do acionador do Lambda de mensagem personalizada

Valor de triggerSource	Event
`CustomMessage_SignUp`	Custom message – Para enviar o código de confirmação após cadastro.
`CustomMessage_AdminCreateUser`	Custom message – Para enviar a senha temporária a um novo usuário.
`CustomMessage_ResendCode`	Custom message – Para reenviar o código de confirmação a um usuário existente.

Valor de triggerSource	Event
`CustomMessage_ForgotPassword`	Custom message – Para enviar o código de confirmação da solicitação de esquecimento de senha.
`CustomMessage_UpdateUserAttribute`	Custom message – Quando um e-mail ou número de telefone de um usuário for alterado, esse trigger enviará um código de verificação automaticamente ao usuário. Não pode ser usado para outros atributos.
`CustomMessage_VerifyUserAttribute`	Mensagem personalizada – Este trigger envia um código de verificação ao usuário quando solicitado manualmente para um novo e-mail ou número de telefone.
`CustomMessage_Authentication`	Custom message – Para enviar o código MFA durante a autenticação.

Parâmetros do acionador do Lambda de mensagem personalizada

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "request": {
        "userAttributes": {
            "string": "string",
            . . .
        }
        "codeParameter": "####",
        "usernameParameter": "string",
        "clientMetadata": {
            "string": "string",
            . . .
```

```
        }
    },
    "response": {
        "smsMessage": "string",
        "emailMessage": "string",
        "emailSubject": "string"
    }
}
```

Parâmetros de solicitação de mensagem personalizada

userAttributes

Um ou mais pares de nome-valor que representam atributos de usuário.

codeParameter

Uma string a ser usada como espaço reservado do código de verificação na mensagem personalizada.

usernameParameter

O nome do usuário. O Amazon Cognito inclui esse parâmetro em solicitações geradas por usuários criados pelo administrador.

clientMetadata

Um ou mais pares de chave-valor que você pode fornecer como entrada personalizada para a função Lambda especificada para o acionador de mensagem personalizada. A solicitação que invoca uma função de mensagem personalizada não inclui dados transmitidos no ClientMetadata parâmetro AdminInitiateAuthe operações de InitiateAuthAPI. Para passar esses dados para sua função Lambda, você pode usar o ClientMetadata parâmetro nas seguintes ações de API:

- AdminResetUserPassword
- AdminRespondToAuthChallenge
- AdminUpdateUserAttributes
- ForgotPassword
- GetUserAttributeVerificationCode
- ResendConfirmationCode
- SignUp
- UpdateUserAttributes

Parâmetros de resposta de mensagem personalizada

Na resposta, especifique o texto personalizado a ser usado em mensagens para seus usuários. Para as restrições de string que o Amazon Cognito aplica a esses parâmetros, consulte. MessageTemplateType

smsMessage

A mensagem SMS personalizada a ser enviada a seus usuários. Deve incluir o valor de `codeParameter` recebido na solicitação.

emailMessage

A mensagem de e-mail personalizada a ser enviada a seus usuários. Você pode usar a formatação HTML no parâmetro `emailMessage`. Deve incluir o valor de `codeParameter` recebido na solicitação como a variável `{####}`. O Amazon Cognito pode usar o parâmetro `emailMessage` somente se o atributo `EmailSendingAccount` do grupo de usuários for `DEVELOPER`. Se o atributo `EmailSendingAccount` do grupo de usuários não for `DEVELOPER` e um parâmetro `emailMessage` for retornado, o Amazon Cognito vai gerar um código de erro 400 `com.amazonaws.cognito.identity.idp.model.InvalidLambdaResponseException`. Ao escolher o Amazon Simple Email Service (Amazon SES) para enviar mensagens de e-mail, o atributo `EmailSendingAccount` de um grupo de usuários é `DEVELOPER`. Do contrário, o valor será `COGNITO_DEFAULT`.

emailSubject

A linha de assunto da mensagem personalizada. Você só pode usar o `emailSubject` parâmetro se o EmailSendingAccount atributo do grupo de usuários for`DEVELOPER`. Se o atributo `EmailSendingAccount` do grupo de usuários não for `DEVELOPER` e o Amazon Cognito retornar um parâmetro `emailSubject`, o Amazon Cognito vai gerar um código de erro 400 `com.amazonaws.cognito.identity.idp.model.InvalidLambdaResponseException`. O atributo `EmailSendingAccount` de um grupo de usuários é `DEVELOPER` ao escolher o Amazon Simple Email Service (Amazon SES) para enviar mensagens de e-mail. Do contrário, o valor será `COGNITO_DEFAULT`.

Exemplo de mensagem personalizada de cadastro

Esse exemplo de função do Lambda personaliza um e-mail ou mensagem SMS quando o serviço requer que uma aplicação envie um código de verificação ao usuário.

O Amazon Cognito pode invocar um acionador do Lambda em vários eventos: no pós-registro, ao reenviar um código de verificação, ao recuperar uma senha esquecida ou ao verificar um atributo de usuário. A resposta inclui mensagens para SMS e e-mail. A mensagem deve incluir o parâmetro de código `"####"`. Esse parâmetro é o espaço reservado do código de verificação que o usuário recebe.

A mensagem de e-mail tem um comprimento máximo de 20 mil caracteres UTF-8. Esse tamanho inclui o código de verificação. Você pode usar etiquetas HTML nessas mensagens de e-mail.

A mensagem SMS tem um comprimento máximo de 140 caracteres UTF-8. Esse tamanho inclui o código de verificação.

Node.js

```
const handler = async (event) => {
  if (event.triggerSource === "CustomMessage_SignUp") {
    const message = `Thank you for signing up. Your confirmation code is
 ${event.request.codeParameter}.`;
    event.response.smsMessage = message;
    event.response.emailMessage = message;
    event.response.emailSubject = "Welcome to the service.";
  }
  return event;
};

export { handler };
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
 "version": "1",
 "region": "us-west-2",
 "userPoolId": "us-west-2_EXAMPLE",
 "userName": "test-user",
 "callerContext": {
  "awsSdkVersion": "aws-sdk-unknown-unknown",
```

```
  "clientId": "1example23456789"
 },
 "triggerSource": "CustomMessage_SignUp",
 "request": {
  "userAttributes": {
   "sub": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
   "cognito:user_status": "CONFIRMED",
   "email_verified": "true",
   "phone_number_verified": "true",
   "phone_number": "+12065551212",
   "email": "test-user@example.com"
  },
  "codeParameter": "{####}",
  "linkParameter": "{##Click Here##}",
  "usernameParameter": "None"
 },
 "response": {
  "smsMessage": "None",
  "emailMessage": "None",
  "emailSubject": "None"
 }
}
```

Exemplo de mensagem personalizada para criação de usuário pelo administrador

A solicitação que o Amazon Cognito enviou para este exemplo de mensagem personalizada da função do Lambda tem um valor `triggerSource` e um nome de usuário `CustomMessage_AdminCreateUser` com uma senha temporária. A função é preenchida com `${event.request.codeParameter}` a partir da senha temporária na solicitação e com `${event.request.usernameParameter}` a partir do nome de usuário na solicitação.

Suas mensagens personalizadas devem inserir os valores de `codeParameter` e `usernameParameter` dentro `smsMessage` e `emailMessage` no objeto de resposta. Neste exemplo, a função grava a mesma mensagem nos campos de resposta `event.response.smsMessage` e `event.response.emailMessage`.

A mensagem de e-mail tem um comprimento máximo de 20 mil caracteres UTF-8. Esse tamanho inclui o código de verificação. Você pode usar etiquetas HTML nesses e-mails. A mensagem SMS tem um comprimento máximo de 140 caracteres UTF-8. Esse tamanho inclui o código de verificação.

A resposta inclui mensagens para SMS e e-mail.

Node.js

```
const handler = async (event) => {
  if (event.triggerSource === "CustomMessage_AdminCreateUser") {
    const message = `Welcome to the service. Your user name is
 ${event.request.usernameParameter}. Your temporary password is
 ${event.request.codeParameter}`;
    event.response.smsMessage = message;
    event.response.emailMessage = message;
    event.response.emailSubject = "Welcome to the service";
  }
  return event;
};

export { handler };
```

O Amazon Cognito transmite informações de evento para a função do Lambda. A função retorna o mesmo objeto de evento para o Amazon Cognito, com as alterações na resposta. No console do Lambda, você pode configurar um evento de teste com dados relevantes para o acionador do Lambda. A seguir, é mostrado um evento de teste para esse exemplo de código:

JSON

```
{
  "version": 1,
  "triggerSource": "CustomMessage_AdminCreateUser",
  "region": "<region>",
  "userPoolId": "<userPoolId>",
  "userName": "<userName>",
  "callerContext": {
      "awsSdk": "<calling aws sdk with version>",
      "clientId": "<apps client id>",
      ...
  },
  "request": {
      "userAttributes": {
          "phone_number_verified": false,
          "email_verified": true,
           ...
      },
      "codeParameter": "####",
      "usernameParameter": "username"
```

```
  },
  "response": {
      "smsMessage": "<custom message to be sent in the message with code parameter
 and username parameter>"
      "emailMessage": "<custom message to be sent in the message with code parameter
 and username parameter>"
      "emailSubject": "<custom email subject>"
  }
}
```

Acionadores do Lambda remetente personalizado

Os acionadores do Lambda `CustomEmailSender` e `CustomSMSSender` permitem notificações por e-mail e SMS de terceiros em grupos de usuários. Você pode escolher provedores de SMS e e-mail para enviar notificações aos usuários de dentro do código de sua função do Lambda. Quando o Amazon Cognito envia convites, códigos de MFA, códigos de confirmação, códigos de verificação e senhas temporárias aos usuários, os eventos ativam suas funções configuradas do Lambda. O Amazon Cognito envia o código e senhas temporárias (segredos) para suas funções ativadas do Lambda. O Amazon Cognito criptografa esses segredos com uma chave gerenciada pelo AWS KMS cliente e o. AWS Encryption SDK AWS Encryption SDK É uma biblioteca de criptografia do lado do cliente que ajuda você a criptografar e descriptografar dados genéricos.

CustomEmailSender

O Amazon Cognito invoca esse acionador para enviar notificações por e-mail aos usuários.

PersonalizadoSMSSender

O Amazon Cognito invoca esse acionador para enviar notificações SMS aos usuários.

Conceitos de criptografia

O Amazon Cognito não envia códigos de usuários em texto simples nos eventos que envia para acionadores personalizados do remetente. As funções do Lambda devem descriptografar códigos nos eventos. Os conceitos a seguir são a arquitetura de criptografia que sua função deve usar para obter códigos que possam ser entregues aos usuários.

AWS KMS

AWS KMS é um serviço gerenciado para criar e controlar AWS KMS chaves. Essas chaves criptografam seus dados. Para obter mais informações, consulte O que é o AWS Key Management Service?.

Chave KMS

Uma chave do KMS é uma representação lógica de uma chave criptográfica. A chave do KMS inclui metadados, como o ID da chave, a data de criação, a descrição e o estado da chave. A chave do KMS também contém o material de chave usado para criptografar e descriptografar dados. Para obter mais informações, consulte, Chaves do AWS KMS.

Chaves simétricas do KMS

Uma chave simétrica do KMS é uma chave de criptografia de 256 bits que não sai do AWS KMS sem ser criptografada. Para usar uma chave KMS simétrica, você deve ligar. AWS KMS O Amazon Cognito usa chaves simétricas. A mesma chave criptografa e descriptografa. Para obter mais informações, consulte Chaves simétricas do KMS.

O que é importante saber sobre acionadores do Lambda de remetente personalizado

- Para configurar seus grupos de usuários para usar esses acionadores do Lambda, é possível usar a AWS CLI ou o SDK. Essas configurações não estão disponíveis no console do Amazon Cognito.

  A operação `UpdateUserPool` define a configuração do Lambda. As solicitações para essa operação exige todos os parâmetros do grupo de usuários e os parâmetros que você deseja modificar. Se você não fornecer todos os parâmetros relevantes, o Amazon Cognito assumirá os valores padrão para todos os parâmetros ausentes. Conforme demonstrado no exemplo de AWS CLI a seguir, inclua entradas para todas as funções do Lambda que você deseja adicionar ou manter em seu grupo de usuários. Para obter mais informações, consulte Como atualizar a configuração do grupo de usuários e do cliente da aplicação.

```
   #Send this parameter in an 'aws cognito-idp update-user-pool' CLI command,
including any existing
   #user pool configurations. This snippet also includes a pre sign-up trigger for
syntax reference. The pre sign-up trigger
   #doesn't have a role in custom sender triggers.

     --lambda-config "PreSignUp=lambda-arn, \
```

```
                        CustomSMSSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                        CustomEmailSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                        KMSKeyID=key-id"
```

Para solicitações que usam o corpo JSON do `UpdateUserPool`, o trecho `LambdaConfig` a seguir atribui funções personalizadas de remetente de SMS e e-mail.

```
"LambdaConfig": {
   "KMSKeyID": "arn:aws:kms:us-
east-1:111122223333:key/a6c4f8e2-0c45-47db-925f-87854bc9e357",
   "CustomEmailSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:111122223333:function:MyFunction",
      "LambdaVersion": "V1_0"
   },
   "CustomSMSSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:111122223333:function:MyFunction",
      "LambdaVersion": "V1_0"
   }
```

- Para remover um gatilho Lambda personalizado do remetente com `update-user-pool` AWS CLI um comando, omita `CustomSMSSender` o parâmetro `CustomEmailSender` `--lambda-config` or e inclua todos os outros gatilhos que você deseja usar com seu grupo de usuários.

  Para remover um acionador do Lambda de remetente personalizado com uma solicitação da API `UpdateUserPool`, omita o parâmetro `CustomSMSSender` ou `CustomEmailSender` do corpo da solicitação que contém o restante da configuração do grupo de usuários.
- O Amazon Cognito faz escapes de caracteres reservados de HTML < (`<`) e > (`>`) na senha temporária do usuário. Esses caracteres podem aparecer em senhas temporárias que o Amazon Cognito envia para à função personalizada de remetente de e-mail, mas não aparecem nos códigos de verificação temporários. Para enviar senhas temporárias, a função do Lambda deve liberar esses caracteres depois de decifrar a senha e antes de enviar a mensagem ao usuário.

Ativar acionadores do Lambda de remetente personalizado

Para usar lógica personalizada para enviar mensagens de SMS ou e-mail ao grupo de usuários, configure acionadores de remetente personalizado. O procedimento a seguir atribui um acionador de SMS personalizado, um acionador de e-mail personalizado ou ambos ao seu grupo de usuários. Depois de adicionar o acionador de remetente de SMS personalizado, o Amazon Cognito sempre

envia atributos do usuário, incluindo o número de telefone e o código único para a função do Lambda, em vez do comportamento padrão que envia uma mensagem SMS ou de e-mail.

1. Crie uma chave de criptografia simétrica em AWS Key Management Service (AWS KMS). O Amazon Cognito gera segredos (senhas temporárias, códigos de verificação, senhas de autenticação de uso único e códigos de autorização) e usa essa chave do KMS para criptografá-los com AWS Encryption SDK. Em seguida, você pode usar o AWS Encryption SDK em sua função Lambda para descriptografar os segredos e enviá-los ao usuário em texto simples.
2. A entidade principal do IAM que cria ou atualiza seu grupo de usuários cria uma concessão única com base na chave KMS que o Amazon Cognito usa para criptografar o código. Conceda essas permissões `CreateGrant` da entidade principal à sua chave KMS. Para que esse exemplo de política de chaves do KMS seja efetivo, o administrador que atualiza o grupo de usuários deve estar conectado com uma sessão de perfil assumido para o perfil do IAM `arn:aws:iam::111222333444:role/my-example-administrator-role`.

   Aplique a política baseada em recursos a seguir, modificada para seu ambiente, à chave do KMS.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::111122223333:role/my-example-administrator-
role"
        },
        "Action": "kms:CreateGrant",
        "Resource": "arn:aws:kms:us-
west-2:111122223333:key/1example-2222-3333-4444-999example",
        "Condition": {
            "StringEquals": {
               "kms:EncryptionContext:userpool-id": "us-west-2_EXAMPLE"
            }
        }
    },
    {
        "Sid": "Allow Lambda to decrypt",
```

```
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::111122223333:role/my-lambda-function-role"
        },
        "Action": "kms:Decrypt",
        "Resource": "*"
    }]
}
```

3. Crie uma função do Lambda para o acionador de remetente personalizado. O Amazon Cognito usa o SDK de criptografia da AWS para criptografar os segredos, as senhas temporárias e os códigos que autorizam as solicitações de API dos usuários.

   a. Atribua um perfil de execução do Lambda que tenha, no mínimo, as permissões `kms:Decrypt` para a chave do KMS.

   b. Componha o código da função do Lambda para enviar as mensagens. O evento de entrada para sua função contém um segredo. Em sua função, decifre o segredo com o AWS Encryption SDK e processe todos os metadados relevantes. Depois, envie o código, sua própria mensagem personalizada e o número de telefone de destino para a API personalizada que entrega a mensagem.

   c. Adicione o AWS Encryption SDK à sua função Lambda. Para ter mais informações, consulte Linguagens de programação do AWS Encryption SDK. Para atualizar o pacote do Lambda, conclua as etapas a seguir.

      i. Exporte a função do Lambda como um arquivo .zip no Console de gerenciamento da AWS.

      ii. Abra sua função e adicione AWS Encryption SDK o. Para ter mais informações e links de download, consulte Linguagens de programação do AWS Encryption SDK no Guia do desenvolvedor do AWS Encryption SDK .

      iii. Compacte a função com as dependências do SDK e faça upload da função para o Lambda. Para obter mais informações, consulte Implantar funções do Lambda como arquivos .zip no Guia do desenvolvedor do AWS Lambda .

4. Conceda à entidade principal `cognito-idp.amazonaws.com` do serviço do Amazon Cognito acesso para invocar a função do Lambda.

   O AWS CLI comando a seguir concede ao Amazon Cognito permissão para invocar sua função Lambda:

```
aws lambda add-permission --function-name lambda_arn --statement-id
 "CognitoLambdaInvokeAccess" --action lambda:InvokeFunction --principal cognito-
idp.amazonaws.com
```

5. Gere uma solicitação de UpdateUserPoolAPI com um `LambdaConfig` parâmetro que adiciona gatilhos Lambda personalizados do remetente. Não é possível adicionar acionadores desse tipo no console do Amazon Cognito. Os acionadores de remetente personalizado exigem parâmetros `LambdaConfig` de `KMSKeyID` e `CustomSMSSender` ou `CustomEmailSender` (ou ambos).

Acionador do Lambda de remetente de e-mail personalizado

Quando você atribui um acionador de remetente de e-mail personalizado ao grupo de usuários, o Amazon Cognito invoca uma função do Lambda em vez do comportamento padrão quando um evento do usuário exige que ele envie uma mensagem de e-mail. Com um gatilho de remetente personalizado, sua AWS Lambda função pode enviar notificações por e-mail para seus usuários por meio de um método e provedor de sua escolha. O código personalizado da função deve processar e entregar todas as mensagens de e-mail do grupo de usuários.

Esse acionador serve para cenários em que talvez você queira ter mais controle sobre como seu grupo de usuários envia mensagens de e-mail. Sua função do Lambda pode personalizar a chamada para as operações da API do Amazon SES, por exemplo, quando você deseja gerenciar várias identidades verificadas ou Regiões da AWS cruzadas. Sua função também pode redirecionar mensagens para outro meio de entrega ou serviço de terceiros.

Para saber mais sobre como configurar um acionador de remetente de e-mail personalizado, consulte Ativar acionadores do Lambda de remetente personalizado.

Fontes do acionador do Lambda de remetente de e-mail personalizado

A tabela a seguir mostra o evento de acionamento de fontes de acionadores de e-mail personalizado no código do Lambda.

`TriggerSource value`	Event
`CustomEmailSender_SignUp`	Um usuário se cadastra e o Amazon Cognito envia uma mensagem de boas-vindas.

TriggerSource value	Event
CustomEmailSender_Authentication	Um usuário faz login e o Amazon Cognito envia um código OTP ou MFA do e-mail.
CustomEmailSender_ForgotPassword	Um usuário solicita um código para redefinir a senha.
CustomEmailSender_ResendCode	Um usuário solicita um código de confirmação da conta de substituição.
CustomEmailSender_UpdateUserAttribute	Um usuário atualiza um endereço de e-mail ou um atributo de número de telefone e o Amazon Cognito envia um código para verificar o atributo.
CustomEmailSender_VerifyUserAttribute	Um usuário cria um endereço de e-mail ou um atributo de número de telefone e o Amazon Cognito envia um código para verificar o atributo.
CustomEmailSender_AdminCreateUser	Você cria um usuário em seu grupo de usuários e o Amazon Cognito envia uma senha temporária.
CustomEmailSender_AccountTakeOverNotification	O Amazon Cognito detecta uma tentativa de tomada de controle de uma conta de usuário e envia uma notificação ao usuário.

Parâmetros do acionador do Lambda de remetente personalizado de e-mail

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "request": {
```

```
        "type": "customEmailSenderRequestV1",
        "code": "string",
        "clientMetadata": {
            "string": "string",
             . . .
            },
        "userAttributes": {
            "string": "string",
            . . .
         }
}
```

Parâmetros de solicitação do remetente personalizado de e-mail

type

A versão da solicitação. Para um evento de remetente personalizado de e-mail, o valor dessa string é sempre `customEmailSenderRequestV1`.

código

O código criptografado que sua função pode descriptografar e enviar ao usuário.

clientMetadata

Um ou mais pares de chave-valor que você pode fornecer como entrada personalizada à função do Lambda para o acionador de migração do usuário. Para passar esses dados para sua função Lambda, você pode usar o ClientMetadata parâmetro nas ações AdminRespondToAuthChallengee da RespondToAuthChallengeAPI. O Amazon Cognito não inclui dados do ClientMetadata parâmetro AdminInitiateAuthe operações de InitiateAuthAPI na solicitação que ele passa para a função de pós-autenticação.

> **Note**
>
> O Amazon Cognito envia `ClientMetadata` para funções personalizadas de acionador de e-mail personalizado em eventos com as seguintes fontes de acionador:
>
> - `CustomEmailSender_ForgotPassword`
> - `CustomEmailSender_SignUp`
> - `CustomEmailSender_Authentication`

O Amazon Cognito não envia `ClientMetadata` em eventos de acionador com `CustomEmailSender_AccountTakeOverNotification` de origem.

userAttributes

Um ou mais pares de chave-valor que representam atributos de usuário.

Parâmetros de resposta do remetente personalizado de e-mail

O Amazon Cognito não espera nenhuma outra informação de retorno na resposta do remetente personalizado de e-mail. Sua função do Lambda deve interpretar o evento, descriptografar o código e, em seguida, entregar o conteúdo da mensagem. Uma função típica reúne uma mensagem de e-mail e a direciona para um retransmissor SMTP de terceiros.

Exemplo de código

O exemplo de Node.js a seguir processa um evento de mensagem de e-mail na função do Lambda de remetente personalizado de e-mail. Esse exemplo pressupõe que a função tenha duas variáveis de ambiente definidas.

`KEY_ID`

O ID da chave do KMS que você deseja usar para criptografar e descriptografar os códigos dos usuários.

`KEY_ARN`

O nome do recurso da Amazon (ARN) da chave do KMS que você deseja usar para criptografar e descriptografar os códigos dos usuários.

Como implantar essa função

1. Instale a versão mais recente do NodeJS na área de trabalho do desenvolvedor.
2. Crie um novo projeto NodeJS em seu espaço de trabalho.
3. Inicialize o projeto com `npm init -y`.
4. Crie o script da função do Lambda: `touch index.mjs`.
5. Cole o conteúdo do exemplo abaixo em `index.mjs`.

6. Baixe a dependência do projeto, AWS Encryption SDK:`npm install @aws-crypto/client-node`.
7. Compacte o diretório de projeto em um arquivo: `zip -r my_deployment_package.zip .`.
8. Implante o arquivo ZIP à função.

Essa função de exemplo descriptografa o código e, para eventos de inscrição, simula o envio de uma mensagem de e-mail para o endereço de e-mail do usuário.

```
import { KmsKeyringNode, buildClient, CommitmentPolicy } from '@aws-crypto/client-
node';

// Configure the encryption SDK client with the KMS key from the environment variables
const { encrypt, decrypt } = buildClient(
    CommitmentPolicy.REQUIRE_ENCRYPT_ALLOW_DECRYPT
);

const generatorKeyId = process.env.KEY_ID;
const keyIds = [process.env.KEY_ARN];
const keyring = new KmsKeyringNode({ generatorKeyId, keyIds });

// Example function to simulate sending email.
// This example logs message details to CloudWatch Logs from your Lambda function.
// Update this function with custom logic that sends an email message to 'emailaddress'
 with body 'message'.
const sendEmail = async (emailAddress, message) => {
    // Log the destination with the email address masked.
    console.log(`Simulating email send to ${emailAddress.replace(/[^@.]/g, '*')}`);
    // Log the message with the code masked.
    console.log(`Message content: ${message.replace(/\b\d{6,8}\b/g, '********')}`);
    // Simulate API delay
    await new Promise(resolve => setTimeout(resolve, 100));
    console.log('Email sent successfully');
    return true;
};

export const handler = async (event) => {
    try {
        // Decrypt the secret code using encryption SDK
        let plainTextCode;
        if (event.request.code) {
```

```
            const { plaintext, messageHeader } = await decrypt(keyring,
 Buffer.from(event.request.code, 'base64'));
            plainTextCode = Buffer.from(plaintext).toString('utf-8');
        }

        // Handle different trigger sources
        if (event.triggerSource == 'CustomEmailSender_SignUp') {
            const emailAddress = event.request.userAttributes.email;
            const message = `Welcome! Your verification code is: ${plainTextCode}`;
            await sendEmail(emailAddress, message);
        }
        else if (event.triggerSource == 'CustomEmailSender_ResendCode') {
            // Handle resend code
        }
        else if (event.triggerSource == 'CustomEmailSender_ForgotPassword') {
            // Handle forgot password
        }
        else if (event.triggerSource == 'CustomEmailSender_UpdateUserAttribute') {
            // Handle update attribute
        }
        else if (event.triggerSource == 'CustomEmailSender_VerifyUserAttribute') {
            // Handle verify attribute
        }
        else if (event.triggerSource == 'CustomEmailSender_AdminCreateUser') {
            // Handle admin create user
        }
        else if (event.triggerSource == 'CustomEmailSender_Authentication') {
            // Handle authentication
        }
        else if (event.triggerSource ==
 'CustomEmailSender_AccountTakeOverNotification') {
            // Handle account takeover notification
        }

        return;
    } catch (error) {
        console.error('Error in custom email sender:', error);
        throw error;
    }
};
```

Acionador do Lambda de remetente personalizado de SMS

Quando você atribui um acionador de remetente de SMS personalizado ao grupo de usuários, o Amazon Cognito invoca uma função do Lambda em vez do comportamento padrão quando um evento do usuário exige que ele envie uma mensagem SMS. Com um gatilho de remetente personalizado, sua AWS Lambda função pode enviar notificações por SMS para seus usuários por meio de um método e provedor de sua escolha. O código personalizado da função deve processar e entregar todas as mensagens SMS do grupo de usuários.

Esse acionador serve para cenários em que talvez você queira ter mais controle sobre como seu grupo de usuários envia mensagens SMS. Sua função Lambda pode personalizar a chamada para as operações de API do Amazon SNS, por exemplo, quando você quiser gerenciar várias IDs originações ou cruzamentos. Regiões da AWS Sua função também pode redirecionar mensagens para outro meio de entrega ou serviço de terceiros.

Para saber mais sobre como configurar um acionador de remetente de e-mail personalizado, consulte Ativar acionadores do Lambda de remetente personalizado.

Fontes de acionador do Lambda remetente personalizado de SMS

A tabela a seguir mostra o evento de acionamento de fontes de acionadores de SMS personalizado no código do Lambda.

`TriggerSource value`	Event
`CustomSMSSender_SignUp`	Um usuário se cadastra e o Amazon Cognito envia uma mensagem de boas-vindas.
`CustomSMSSender_ForgotPassword`	Um usuário solicita um código para redefinir a senha.
`CustomSMSSender_ResendCode`	Um usuário solicita um novo código para confirmar seu registro.
`CustomSMSSender_VerifyUserAttribute`	Um usuário cria um endereço de e-mail ou um atributo de número de telefone e o Amazon Cognito envia um código para verificar o atributo.

TriggerSource value	Event
CustomSMSSender_UpdateUserAttribute	Um usuário atualiza um endereço de e-mail ou um atributo de número de telefone e o Amazon Cognito envia um código para verificar o atributo.
CustomSMSSender_Authentication	Um usuário faz login e o Amazon Cognito envia um código OTP ou MFA do SMS.
CustomSMSSender_AdminCreateUser	Você cria um usuário em seu grupo de usuários e o Amazon Cognito envia uma senha temporária.

Parâmetros do acionador do Lambda de remetente personalizado de SMS

A solicitação que o Amazon Cognito transmite para essa função do Lambda é uma combinação dos parâmetros abaixo e dos parâmetros comuns que o Amazon Cognito adiciona a todas as solicitações.

JSON

```
{
    "request": {
        "type": "customSMSSenderRequestV1",
        "code": "string",
        "clientMetadata": {
            "string": "string",
             . . .
            },
        "userAttributes": {
            "string": "string",
            . . .
         }
}
```

Parâmetros de solicitação do remetente personalizado de SMS

type

A versão da solicitação. Para um evento de remetente personalizado de SMS, o valor dessa string é sempre `customSMSSenderRequestV1`.

código

O código criptografado que sua função pode descriptografar e enviar ao usuário.

clientMetadata

Um ou mais pares de chave-valor que você pode fornecer como entrada personalizada ao acionador da função do Lambda de remetente personalizado de SMS. Para passar esses dados para sua função Lambda, você pode usar o ClientMetadata parâmetro nas ações AdminRespondToAuthChallengee da RespondToAuthChallengeAPI. O Amazon Cognito não inclui dados do ClientMetadata parâmetro AdminInitiateAuthe operações de InitiateAuthAPI na solicitação que ele passa para a função de pós-autenticação.

userAttributes

Um ou mais pares de chave-valor que representam atributos de usuário.

Parâmetros de resposta do remetente personalizado de SMS

O Amazon Cognito não espera nenhuma outra informação de retorno na resposta. Sua função pode usar operações de API para consultar e modificar seus recursos ou registrar metadados de eventos em um sistema externo.

Exemplo de código

O exemplo do Node.js a seguir processar um evento de mensagem SMS na função do Lambda de remetente personalizado de SMS. Esse exemplo pressupõe que a função tenha duas variáveis de ambiente definidas.

`KEY_ID`

O ID da chave do KMS que você deseja usar para criptografar e descriptografar os códigos dos usuários.

KEY_ARN

O nome do recurso da Amazon (ARN) da chave do KMS que você deseja usar para criptografar e descriptografar os códigos dos usuários.

Como implantar essa função

1. Instale a versão mais recente do NodeJS na área de trabalho do desenvolvedor.
2. Crie um novo projeto NodeJS em seu espaço de trabalho.
3. Inicialize o projeto com `npm init -y`.
4. Crie o script da função do Lambda: `touch index.mjs`.
5. Cole o conteúdo do exemplo abaixo em `index.mjs`.
6. Baixe a dependência do projeto, AWS Encryption SDK:`npm install @aws-crypto/client-node`.
7. Compacte o diretório de projeto em um arquivo: `zip -r my_deployment_package.zip .`.
8. Implante o arquivo ZIP à função.

```
import { KmsKeyringNode, buildClient, CommitmentPolicy } from '@aws-crypto/client-
node';

// Configure the encryption SDK client with the KMS key from the environment variables
const { encrypt, decrypt } = buildClient(
    CommitmentPolicy.REQUIRE_ENCRYPT_ALLOW_DECRYPT
);

const generatorKeyId = process.env.KEY_ID;
const keyIds = [process.env.KEY_ARN];
const keyring = new KmsKeyringNode({ generatorKeyId, keyIds });

// Example function to simulate sending SMS.
// This example logs message details to CloudWatch Logs from your Lambda function.
// Update this function with custom logic that sends an SMS message to 'phoneNumber'
 with body 'message'.
const sendSMS = async (phoneNumber, message) => {
    // Log the destination with the phone number masked.
    console.log(`Simulating SMS send to ${phoneNumber.replace(/[^+]/g, '*')}`);
    // Log the message with the code masked.
    console.log(`Message content: ${message.replace(/\b\d{6,8}\b/g, '********')}`);
```

```
    // Simulate API delay
    await new Promise(resolve => setTimeout(resolve, 100));
    console.log('SMS sent successfully');
    return true;
};

export const handler = async (event) => {
    try {
        // Decrypt the secret code using encryption SDK
        let plainTextCode;
        if (event.request.code) {
            const { plaintext, messageHeader } = await decrypt(keyring,
 Buffer.from(event.request.code, 'base64'));
            plainTextCode = Buffer.from(plaintext).toString('utf-8');
        }

        // Handle different trigger sources
        if (event.triggerSource == 'CustomSMSSender_SignUp') {
            const phoneNumber = event.request.userAttributes.phone_number;
            const message = `Welcome! Your verification code is: ${plainTextCode}`;
            await sendSMS(phoneNumber, message);
        }
        else if (event.triggerSource == 'CustomSMSSender_ResendCode') {
            // Handle resend code
        }
        else if (event.triggerSource == 'CustomSMSSender_ForgotPassword') {
            // Handle forgot password
        }
        else if (event.triggerSource == 'CustomSMSSender_UpdateUserAttribute') {
            // Handle update attribute
        }
        else if (event.triggerSource == 'CustomSMSSender_VerifyUserAttribute') {
            // Handle verify attribute
        }
        else if (event.triggerSource == 'CustomSMSSender_AdminCreateUser') {
            // Handle admin create user
        }
        return;
    } catch (error) {
        console.error('Error in custom SMS sender:', error);
        throw error;
    }
};
```

Tópicos

Avaliar os recursos de mensagem SMS com uma função de remetente personalizado de SMS

Uma função do Lambda de remetente personalizado de SMS aceitará as mensagens SMS que seu grupo de usuários enviar e fornecerá o conteúdo com base em sua lógica personalizada. O Amazon Cognito envia o Parâmetros do acionador do Lambda de remetente personalizado de SMS para sua função. Sua função pode fazer o que você quiser com essas informações. Por exemplo, você pode enviar o código a um tópico do Amazon Simple Notification Service (Amazon SNS). Um assinante de tópicos do Amazon SNS pode ser uma mensagem SMS, um endpoint HTTPS ou um endereço de e-mail.

Para criar um ambiente de teste para mensagens SMS do Amazon Cognito com uma função Lambda personalizada do remetente de SMS, amazon-cognito-user-poolconsulte development-and-testing-with - sms-redirected-to-email - na biblioteca aws-samples em. GitHub O repositório contém AWS CloudFormation modelos que podem criar um novo grupo de usuários ou trabalhar com um grupo de usuários que você já tem. Esses modelos criam funções do Lambda e um tópico do Amazon SNS. A função do Lambda que o modelo atribui como um acionador de remetente personalizado de SMS redireciona para o tópico do Amazon SNS as mensagens SMS que você envia aos assinantes.

Quando você implanta essa solução em um grupo de usuários, todas as mensagens que o Amazon Cognito geralmente envia pelo sistema de mensagens SMS são enviadas pela função do Lambda a um endereço de e-mail central. Use essa solução para personalizar e visualizar mensagens SMS e testar os eventos do grupo de usuários que fazem com que o Amazon Cognito envie uma mensagem SMS. Depois de concluir seus testes, reverta a CloudFormation pilha ou remova a atribuição personalizada da função de remetente de SMS do seu grupo de usuários.

> **Important**
>
> Não use os modelos em amazon-cognito-user-pool- development-and-testing-with - sms-redirected-to-email para criar um ambiente de produção. A função do Lambda de remetente personalizado de SMS na solução simula mensagens SMS, mas envia todas elas a um único endereço de e-mail central. Antes de enviar mensagens SMS em um grupo de usuários do Amazon Cognito de produção, você deve preencher os requisito s mostrados em Configurações de mensagens SMS para grupos de usuários do Amazon Cognito.

Como gerenciar usuários em seu grupo de usuários

Depois de criar um grupo de usuários, você poderá criar, confirmar e gerenciar contas de usuários. Com os grupos de usuários do Amazon Cognito, você pode gerenciar seus usuários e o acesso deles a recursos mapeando funções do IAM para grupos.

O gerenciamento de usuários em seu grupo de usuários do Amazon Cognito envolve várias opções de configuração e tarefas administrativas. Os grupos de usuários podem ser escalados para milhões de usuários. Um diretório de usuários dessa escala requer ferramentas administrativas igualmente escaláveis e reproduzíveis. Você pode querer criar vários perfis de usuário, gerenciar usuários inativos, produzir relatórios de governança e conformidade ou configurar ferramentas de autoatendimento em que os usuários façam a maior parte do trabalho. Depois de criar um grupo de usuários, você pode controlar como os usuários se cadastram e confirmam suas contas, incluindo solicitar verificação de e-mail ou número de telefone. Os administradores também podem criar contas de usuário diretamente e personalizar as mensagens de boas-vindas e os requisitos de senha.

Nos grupos de usuários, você pode gerenciar o acesso aos recursos com base na associação de um usuário ao grupo. Você pode atribuir perfis do IAM a esses grupos para gerenciar o acesso aos Serviços da AWS com bancos de identidades. A associação ao grupo de usuários está presente nos tokens de ID e de acesso. Com essas informações, você pode tomar decisões de controle de acesso em tempo de execução na aplicação ou com um mecanismo de políticas como o Amazon Verified Permissions.

Os grupos de usuários geralmente têm muitos usuários. Você geralmente pesquisará e atualizará contas de usuário. O console e a API do Amazon Cognito oferecem suporte à consulta de usuários com base em atributos padrão, como nome de usuário, e-mail e número de telefone. Os administradores também podem redefinir senhas, desativar contas e visualizar o histórico de eventos do usuário.

Para migrar dados de usuários existentes, o Amazon Cognito tem opções para importar usuários de um arquivo CSV e usar um acionador do Lambda para migrar automaticamente os usuários quando eles fizerem login pela primeira vez. Essas opções oferecem suporte a transições de usuários de outros diretórios para o seu grupo de usuários.

Você pode usar os recursos de gerenciamento de usuários nos grupos de usuários para ter um controle refinado sobre o ciclo de vida do usuário e a experiência de autenticação. A combinação de cadastro de autoatendimento, contas criadas pelo administrador, grupos e ferramentas de migração faz dos grupos de usuários do Amazon Cognito um diretório flexível.

Tópicos

Configurar políticas para a criação de usuários

O grupo de usuários pode permitir que os usuários se inscrevam ou você pode criá-los como administrador. Também é possível controlar quanto do processo de verificação e de confirmação após o cadastro os usuários podem realizar. Por exemplo, talvez você queira revisar os cadastros e aceitá-los com base em um processo de validação externo. Essa configuração ou a política de criação de usuários do administrador também define a quantidade de tempo antes da qual um usuário não pode mais confirmar sua conta de usuário.

O Amazon Cognito pode atender às necessidades de seus clientes públicos como a plataforma de gerenciamento de identidade e acesso de clientes (CIAM) para seu software. Um grupo de usuários que aceita se cadastrar e tem um cliente da aplicação, com ou sem login gerenciado, cria um perfil de usuário para qualquer pessoa na Internet que conheça seu ID de cliente da aplicação que pode ser descoberto publicamente e solicite o cadastro. Um perfil de usuário cadastrado pode receber tokens de acesso e identidade e acessar recursos que você autorizou para a aplicação. Antes de ativar o cadastro em seu grupo de usuários, revise suas opções e certifique-se de que sua configuração esteja em conformidade com seus padrões de segurança. Defina Habilitar autorregistro e `AllowAdminCreateUserOnly`, descritos nos procedimentos a seguir, com cuidado.

Console de gerenciamento da AWS

O menu Cadastrar-se do grupo de usuários contém algumas das configurações de cadastro e criação administrativa de usuários no grupo de usuários.

Como configurar a experiência de cadastro

1. Em Verificação e confirmação assistidas pelo Cognito, escolha se deseja Permitir que o Cognito envie mensagens automaticamente para verificar e confirmar. Com essa configuração ativada, o Amazon Cognito envia um e-mail ou mensagem SMS para novos usuários com um código que eles devem apresentar ao grupo de usuários. Isso confirma a propriedade do endereço de e-mail ou do número de telefone, configurando o atributo equivalente como verificado e confirmando a conta do usuário para login. Os Atributos a serem verificados escolhidos determinam os métodos de entrega e os destinos das mensagens de verificação.
2. A Verificação das alterações de atributos não é significativa quando você está criando usuários, mas está relacionada à verificação dos atributos. É possível permitir que os usuários que alteraram, mas ainda não verificaram, seus atributos de login continuem fazendo login com o novo valor de atributo ou com o original. Para obter mais informações, consulte Como verificar quando usuários alteram o e-mail ou o número de telefone.
3. A opção Atributos obrigatórios exibe os atributos que devem receber um valor antes que um usuário possa se cadastrar ou que você possa criar um usuário. Os atributos obrigatórios só podem ser definidos na criação de um grupo de usuários.
4. Os Atributos personalizados são importantes para o processo de criação e cadastro do usuário porque, ao criar um usuário pela primeira vez, você só pode definir um valor para atributos personalizados imutáveis. Para obter mais informações sobre atributos personalizados, consulte Atributos personalizados.
5. Em Cadastro por autoatendimento, selecione Habilitar autorregistro se desejar que os usuários possam gerar uma nova conta com a API `SignUp` não autenticada. Se você desativar o autorregistro, só poderá criar novos usuários como administrador, no console do Amazon Cognito ou AdminCreateUsercom solicitações de API. Em um grupo de usuários em que o autorregistro está inativo, as solicitações de SignUpAPI retornam `NotAuthorizedException` e o login gerenciado não exibe um link de inscrição.

Para grupos de usuários nos quais você planeja criar usuários como administrador, é possível configurar a duração das senhas temporárias no menu Métodos de autenticação em Senhas temporárias definidas por administradores expiram em.

Outro elemento importante da criação de usuários como administrador é a mensagem de convite. Ao criar um novo usuário, o Amazon Cognito envia uma mensagem ao usuário com um link para

a sua aplicação para que o usuário possa fazer login pela primeira vez. Personalize esse modelo de mensagem no menu Métodos de autenticação, em Modelos de mensagens.

É possível configurar clientes de aplicações confidenciais, geralmente aplicações da Web, com um segredo de cliente que impede o cadastro sem o segredo do cliente da aplicação. Como prática recomendada de segurança, não distribua segredos de clientes de aplicações em clientes de aplicações públicas, geralmente aplicativos móveis. É possível criar clientes da aplicação com segredos do cliente no menu Clientes da aplicação do console do Amazon Cognito.

Amazon Cognito user pools API

Você pode definir programaticamente os parâmetros para a criação de usuários em um grupo de usuários em uma solicitação CreateUserPoolde UpdateUserPoolAPI.

O AdminCreateUserConfigelemento define valores para as seguintes propriedades de um grupo de usuários.

1. Habilitar cadastro por autoatendimento
2. A mensagem de convite que você envia aos novos usuários criados por administrador

O exemplo a seguir, quando adicionado ao corpo de uma solicitação completa de API, define um grupo de usuários com cadastro por autoatendimento inativo e um e-mail de convite básico.

```
"AdminCreateUserConfig": {
      "AllowAdminCreateUserOnly": true,
      "InviteMessageTemplate": {
         "EmailMessage": "Your username is {username} and temporary password is
 {####}.",
         "EmailSubject": "Welcome to ExampleApp",
         "SMSMessage": "Your username is {username} and temporary password is
 {####}."
      }
   }
```

Os seguintes parâmetros adicionais de uma solicitação de UpdateUserPoolAPI CreateUserPoolou API governam a criação de novos usuários.

AutoVerifiedAttributes

Os atributos, endereços de e-mail ou números de telefone, aos quais você deseja enviar uma mensagem automaticamente ao registrar um novo usuário.

Políticas

A política de senha do grupo de usuários.

Esquema

Os atributos personalizados do grupo de usuários. Esses são importantes para o processo de criação e cadastro do usuário porque, ao criar um usuário pela primeira vez, você só pode definir um valor para atributos personalizados imutáveis.

Esse parâmetro também define os atributos necessários para o grupo de usuários. O texto a seguir, quando inserido no elemento `Schema` no corpo completo de uma solicitação da API, define o atributo `email` conforme necessário.

```
{
            "Name": "email",
            "Required": true
}
```

Como cadastrar e confirmar contas de usuários

Contas de usuários são adicionadas ao grupo de usuários de uma das seguintes formas:

- O usuário se cadastra aplicação cliente do grupo de usuários. Pode ser um aplicativo móvel ou uma aplicação Web.
- Você pode importar a conta de usuário para o grupo de usuários. Para obter mais informações, consulte Como importar usuários para grupos de usuários com base em um arquivo CSV.
- Você pode criar a conta de usuário em seu grupo e convidá-lo a login. Para obter mais informações, consulte Como criar contas de usuário como administrador.

Os usuários que se inscrevem precisam primeiro ser confirmados para que possam fazer login. Usuários importados e criados já estão confirmados, mas eles precisam criar uma senha própria na primeira vez em que fizerem login. As seções a seguir explicam o processo de confirmação e verificação de e-mail e telefone.

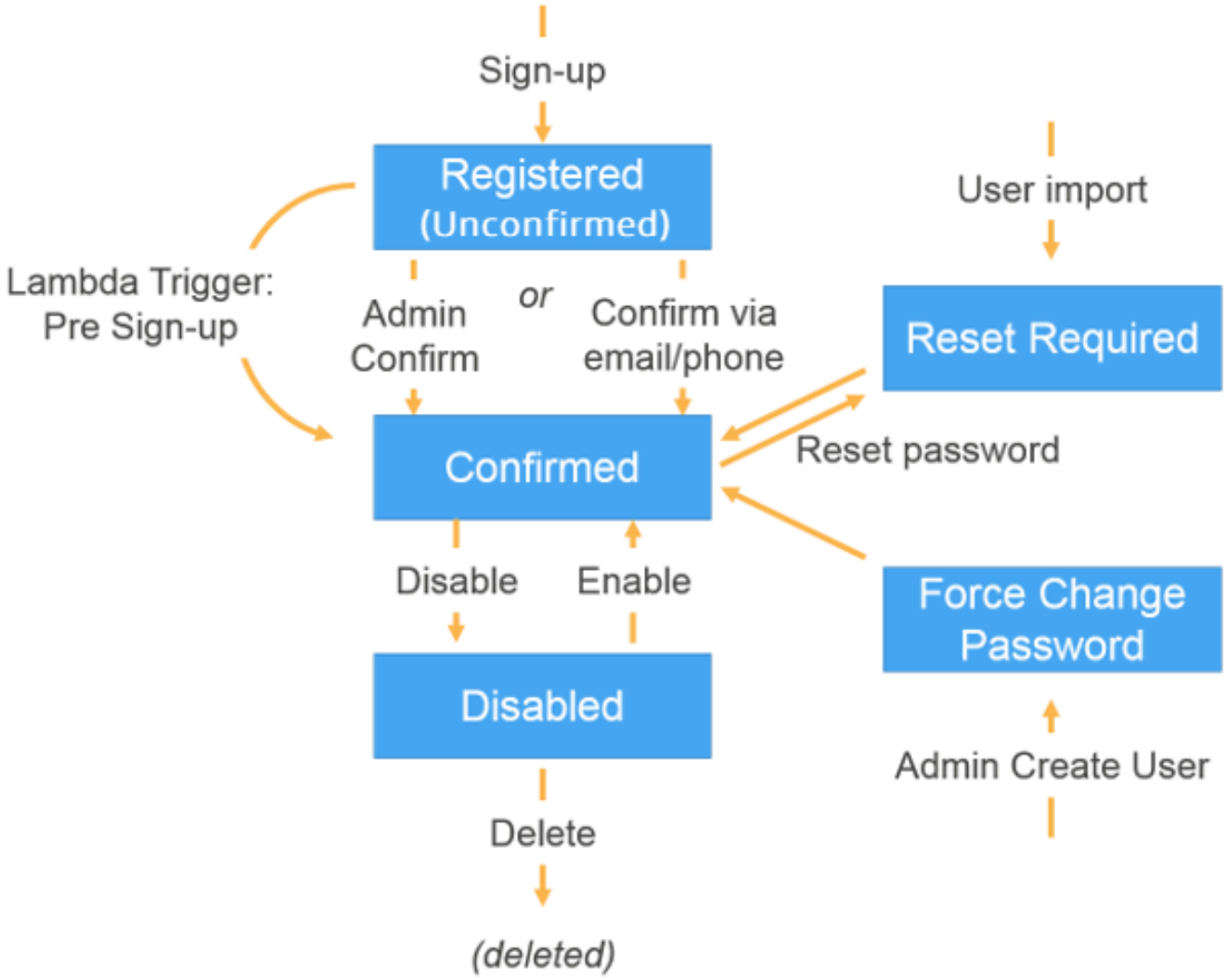
Senhas no cadastro

O Amazon Cognito exige senhas de todos os usuários quando eles se cadastram, exceto sob as condições abaixo. Se todas essas condições forem atendidas, você poderá omitir as senhas nas operações de cadastro.

1. O Login sem senha está ativo em seu grupo de usuários e cliente de aplicação.
2. Seu aplicativo é personalizado com módulos de autenticação em um AWS SDK. O login gerenciado e a IU hospedada sempre exigem senhas.
3. Os usuários fornecem valores de atributos para os métodos de login sem senha — senhas de uso único por e-mail ou mensagem SMS () — que você permite. OTPs Por exemplo, se você permitir o login com uma OTP enviada por e-mail e telefone, os usuários poderão fornecer um número de telefone ou endereço de e-mail. Porém, se permitir somente o login com e-mail, eles deverão fornecer um endereço de e-mail.
4. Seu grupo de usuários verifica automaticamente os atributos que os usuários podem usar com o login sem senha.
5. Para qualquer SignUpsolicitação, o usuário não fornece um valor para o parâmetro Password.

Visão geral da confirmação de conta de usuário

O diagrama a seguir ilustra o processo de confirmação:



Uma conta de usuário pode estar em qualquer um dos seguintes estados:

Registrado (não confirmado)

O usuário foi registrado com êxito, mas não pode fazer login até que a conta seja confirmada. O usuário está habilitado, mas não está confirmado nesse estado.

Novos usuários que se cadastram começam nesse estado.

Confirmado

A conta de usuário está confirmada e o usuário pode fazer login. Quando um usuário insere um código ou segue um link de e-mail para confirmar sua conta de usuário, o e-mail ou o número de telefone é automaticamente confirmado. O código ou link é válido por 24 horas.

Se a conta de usuário tiver sido confirmada pelo administrador ou por um acionador de pré-cadastro do Lambda, é possível que não haja um número de telefone ou um e-mail verificado associado à conta.

É necessário redefinir a senha

A conta de usuário está confirmada, mas o usuário deve solicitar um código e redefinir sua senha para poder fazer login.

As contas de usuários que são importados por um administrador ou desenvolvedor começam nesse estado.

Forçar alteração de senha

A conta de usuário está confirmada e o usuário pode fazer login usando uma senha temporária. No entanto, no primeiro login, ele deve alterar a senha para um novo valor antes de fazer qualquer coisa.

As contas de usuários que são criadas por um administrador ou desenvolvedor começam nesse estado.

Desabilitado

Antes de excluir uma conta de usuário, você precisa desabilitar o acesso de login para esse usuário.

Mais atributos

- Detectar e corrigir contas de usuários inativas com o Amazon Cognito

Como verificar informações de contato no cadastro

Quando novos usuários se cadastram no seu aplicativo, você provavelmente deseja que eles forneçam pelo menos um método de contato. Por exemplo, com as informações de contato dos usuários, você pode:

- Enviar uma senha temporária quando um usuário decide redefinir a senha.
- Notificar os usuários quando as informações pessoais ou financeiras deles forem atualizadas.
- Enviar mensagens promocionais, como descontos ou ofertas especiais.
- Enviar resumos da conta ou lembretes de faturas.

Para casos de uso como esses, é importante que você envie suas mensagens para um destino verificado. Caso contrário, suas mensagens podem ser enviadas para um endereço de e-mail inválido ou para um número de telefone que foi digitado incorretamente. Ou pior, você pode enviar informações confidenciais para agentes maldosos que se passam por seus usuários.

Para ajudar a garantir que você envie mensagens apenas aos indivíduos certos, configure o grupo de usuários do Amazon Cognito de modo que os usuários tenham que fornecer o seguinte, quando se cadastrarem:

a. Um endereço de e-mail ou número de telefone.
b. Um código de verificação que o Amazon Cognito envia para esse endereço de e-mail ou número de telefone. Se tiverem passado 24 horas e o código ou link do seu usuário não for mais válido, chame a operação da ResendConfirmationCodeAPI para gerar e enviar um novo código ou link.

Ao fornecer o código de verificação, um usuário comprova que tem acesso à caixa de correio ou ao telefone que recebeu o código. Depois que o usuário fornece o código, o Amazon Cognito atualiza as informações sobre o usuário no grupo de usuários das seguintes maneiras:

- Definindo o status do usuário como `CONFIRMED`.
- Atualizando os atributos do usuário para indicar que o endereço de e-mail ou número de telefone é verificado.

Para visualizar essas informações, você pode usar o console do Amazon Cognito. Ou você pode usar a operação de `AdminGetUser` API, o `admin-get-user` comando com AWS CLI o. ou uma ação correspondente em um dos AWS SDKs.

Se um usuário tiver um método de contato verificado, o Amazon Cognito enviará automaticamente uma mensagem ao usuário quando ele solicitar uma redefinição de senha.

Outras ações que confirmam e verificam os atributos do usuário

A atividade do usuário a seguir verifica os atributos do usuário. Você não precisa definir esses atributos para verificação automática: as ações listadas os marcam como verificados em todos os casos.

Endereço de e-mail

1. Concluir com sucesso a autenticação sem senha com uma senha de uso único (OTP) enviada por e-mail.

2. Concluir com sucesso a autenticação multifator (MFA) com uma OTP enviada por e-mail.

Número de telefone

1. Concluir com sucesso a autenticação sem senha com uma OTP por SMS.
2. Concluir com sucesso a MFA com uma OTP por SMS.

Para configurar o grupo de usuários para exigir a verificação de e-mail ou telefone

Ao confirmar os endereços de e-mail e os números de telefone, você garante que possa entrar em contato com seus usuários. Conclua as etapas a seguir Console de gerenciamento da AWS para configurar seu grupo de usuários para exigir que seus usuários confirmem seus endereços de e-mail ou números de telefone.

> **Note**
>
> Se você ainda não tiver um grupo de usuários em sua conta, consulte Conceitos básicos dos grupos de usuários.

Para configurar o grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. No painel de navegação, escolha User Pools (Grupos de usuários). Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
3. Clique no menu Cadastrar-se e localize Verificação de atributo e confirmação da conta do usuário. Escolha Editar.
4. Em Verificação e confirmação assistidas pelo Cognito, escolha se deseja Permitir que o Cognito envie mensagens automaticamente para verificar e confirmar. Com essa configuração ativada, o Amazon Cognito envia mensagens para os atributos de contato dos usuários que você escolhe quando um usuário se cadastra ou cria um perfil de usuário. Para verificar os atributos e confirmar os perfis de usuários para login, o Amazon Cognito envia um código ou link em mensagens para os usuários. Os usuários devem inserir o código na interface do usuário para que a aplicação possa confirmá-lo em uma solicitação de API `AdminConfirmSignUp` e `ConfirmSignUp`.

> **Note**
>
> Também é possível desabilitar Cognito-assisted verification and confirmation (Verificação e confirmação assistidas pelo Cognito) e usar ações de API ou acionadores do Lambda autenticados para verificar atributos e confirmar usuários.
> Se você escolher essa opção, o Amazon Cognito não enviará códigos de verificação quando o usuário se cadastrar. Escolha essa opção se você estiver usando um fluxo de autenticação personalizado que verifica pelo menos um método de contato sem usar os códigos de verificação do Amazon Cognito. Por exemplo, você pode usar um gatilho de pré-cadastro do Lambda que verifica automaticamente endereços de e-mail que pertencem a um domínio específico.
> Se você não verificar as informações de contato dos usuários, talvez eles não consigam usar a aplicação. Lembre-se de que os usuários precisam de informações de contato verificadas para:
>
> - Redefinir as próprias senhas: quando um usuário seleciona uma opção na sua aplicação que chama a ação de API `ForgotPassword`, o Amazon Cognito envia uma senha temporária para o endereço de e-mail ou número de telefone do usuário. O Amazon Cognito só enviará essa senha se o usuário tiver pelo menos um método de contato verificado.
> - Fazer login usando um endereço de e-mail ou número de telefone como um alias: se você configurar o grupo de usuários para permitir esses aliases, um usuário só poderá fazer o acesso com um alias se o alias for verificado. Para obter mais informações, consulte Personalização dos atributos de login.

5. Selecione os Attributes to verify (Atributos para verificar):

   Enviar mensagem de SMS, verificar o número de telefone

   O Amazon Cognito envia um código de verificação em uma mensagem de SMS quando o usuário se cadastra. Selecione essa opção se você normalmente se comunicar com os usuários por SMS. Por exemplo, você precisará usar números de telefone verificados se enviar notificações de entrega, confirmações de compromissos ou alertas. Os números de telefone do usuário serão o atributo verificado quando as contas forem confirmadas; você deve adotar medidas adicionais para verificar e se comunicar com endereços de e-mail do usuário.

Enviar mensagem de e-mail, verificar endereço de e-mail

O Amazon Cognito envia um código de verificação por meio de uma mensagem de SMS quando o usuário se cadastra. Escolha essa opção se você normalmente se comunica com os usuários por e-mail. Por exemplo, você precisará usar endereços de e-mail verificados se enviar faturas, resumos de pedidos ou ofertas especiais. Os endereços de e-mail do usuário serão o atributo verificado quando as contas forem confirmadas; você deve adotar medidas adicionais para verificar e se comunicar com números de telefone do usuário.

Enviar mensagem de SMS se o número de telefone estiver disponível, caso contrário, enviar uma mensagem de e-mail

Escolha essa opção se você não exigir que todos os usuários tenham o mesmo método de contato verificado. Nesse caso, a página de cadastro em seu aplicativo pode solicitar que os usuários verifiquem apenas o método de contato que preferirem. Quando o Amazon Cognito envia um código de verificação, ele envia o código para o método de contato fornecido na solicitação de `SignUp` da sua aplicação. Se um usuário fornecer um endereço de e-mail e um número de telefone, e a aplicação fornecer os dois métodos de contato na solicitação de `SignUp`, o Amazon Cognito enviará um código de verificação somente para o número de telefone.

Se você exige que os usuários verifiquem um endereço de e-mail e um número de telefone, escolha esta opção. O Amazon Cognito verifica um método de contato quando o usuário se cadastra e sua aplicação precisará verificar o outro método de contato depois que o usuário fizer login. Para obter mais informações, consulte Se você necessitar que os usuários confirmem tanto endereços de e-mail como números de telefone.

6. Escolha Salvar alterações.

Fluxo de autenticação com verificação por e-mail ou telefone

Se o grupo de usuários exigir que os usuários verifiquem as informações de contato, seu aplicativo deverá facilitar o seguinte fluxo quando um usuário se cadastrar:

1. Um usuário se inscreve no seu aplicativo inserindo um nome de usuário, número de telefone, endereço de and/or e-mail e possivelmente outros atributos.
2. O serviço do Amazon Cognito recebe a solicitação de cadastro da aplicação. Depois de verificar se a solicitação contém todos os atributos necessários para o cadastro, o serviço conclui o

processo de cadastro e envia um código de confirmação para o telefone do usuário (em uma mensagem SMS) ou o e-mail. O código é válido por 24 horas.

3. O serviço retorna para o aplicativo a informação de que o cadastro está concluído e que a conta de usuário está aguardando confirmação. A resposta contém informações sobre o destino para onde o código de confirmação foi enviado. Nesse ponto, a conta de usuário está em um estado não confirmado, e o endereço de e-mail e número de telefone do usuário não estão verificados.
4. O aplicativo agora pode solicitar que o usuário insira o código de confirmação. O usuário não precisa inserir o código imediatamente. No entanto, o usuário não poderá fazer login até que ele insira o código de confirmação.
5. O usuário insere o código de confirmação no aplicativo.
6. A aplicação chama `ConfirmSignUp` para enviar o código para o serviço do Amazon Cognito, que verifica o código e, se ele estiver correto, definirá a conta de usuário para o estado confirmado. Depois que a conta de usuário for confirmada com êxito, o serviço do Amazon Cognito marcará automaticamente como verificado o atributo que foi usado para a confirmação (e-mail ou número de telefone). A menos que o valor do atributo seja alterado, o usuário não precisará fazer a verificação novamente.
7. Nesse momento, como a conta de usuário está no estado confirmado, o usuário pode fazer login.

Se você necessitar que os usuários confirmem tanto endereços de e-mail como números de telefone

O Amazon Cognito verifica apenas um método de contato quando um usuário se cadastra. Nos casos em que o Amazon Cognito precise escolher entre confirmar um endereço de e-mail ou um número de telefone, ele opta por confirmar o número de telefone enviando um código de confirmação por SMS. Por exemplo, se você configurar o grupo de usuários para permitir que os usuários verifiquem tanto endereços de e-mail como números de telefone, e se a aplicação fornecer ambos os atributos no cadastro, o Amazon Cognito verificará apenas o número de telefone. Depois que um usuário verifica o número de telefone dele, o Amazon Cognito define o respetivo status como `CONFIRMED` e ele tem permissão para fazer login na aplicação.

Depois que o usuário fizer login, o aplicativo poderá fornecer a opção de verificar o método de contato que não foi verificado durante o cadastro. Para verificar esse segundo método, o aplicativo chama a ação de API `VerifyUserAttribute`. Observe que essa ação requer um parâmetro `AccessToken` e o Amazon Cognito só fornece tokens de acesso para usuários autenticados. Portanto, você poderá verificar o segundo método de contato somente depois que o usuário fizer login.

Se você exigir que os usuários verifiquem tanto endereços de e-mail como números de telefone, faça o seguinte:

1. Configure o grupo de usuários para permitir que os usuários verifiquem endereços de e-mail ou números de telefone.
2. No fluxo de cadastro do aplicativo, exija que os usuários forneçam tanto um endereço de e-mail como um número de telefone. Chame a ação de API `SignUp` e forneça o endereço de e-mail e o número de telefone para o parâmetro `UserAttributes`. Nesse momento, o Amazon Cognito envia um código de verificação para o telefone do usuário.
3. Na interface do aplicativo, é apresentada uma página de confirmação onde o usuário insere o código de verificação. Confirme o usuário chamando a ação de API `ConfirmSignUp`. Nesse ponto, o status do usuário é `CONFIRMED`, e o número de telefone do usuário é verificado, mas o endereço de e-mail não é verificado.
4. Apresente a página de login e autentique o usuário chamando a ação de API `InitiateAuth`. Depois que o usuário é autenticado, o Amazon Cognito retorna um token de acesso para a aplicação.
5. Chame a ação de API `GetUserAttributeVerificationCode`. Especifique os seguintes parâmetros na solicitação:

   - `AccessToken`: o token de acesso retornado pelo Amazon Cognito quando o usuário fez login.
   - `AttributeName`: especifique `"email"` como o valor do atributo.

   O Amazon Cognito envia um código de verificação para o endereço de e-mail do usuário.
6. Apresente uma página de confirmação onde o usuário insere o código de verificação. Quando o usuário enviar o código, chame a ação de API `VerifyUserAttribute`. Especifique os seguintes parâmetros na solicitação:

   - `AccessToken`: o token de acesso retornado pelo Amazon Cognito quando o usuário fez login.
   - `AttributeName`: especifique `"email"` como o valor do atributo.
   - `Code`: o código de verificação que o usuário forneceu.

   Nesse ponto, o endereço de e-mail é verificado.

Permitir que os usuários se inscrevam na aplicação, mas mediante confirmação deles como administradores do grupo de usuários

Talvez você não queira que o grupo de usuários envie automaticamente mensagens de verificação no grupo de usuários, mas ainda queira que qualquer pessoa se inscreva em uma conta. Esse modelo deixa espaço, por exemplo, para análise humana de novas solicitações de inscrição e para validação em lote e processamento de inscrições. Você pode confirmar novas contas de usuário no console do Amazon Cognito ou com a operação de API autenticada pelo IAM. AdminConfirmSignUp Você pode confirmar contas de usuário como administrador, independentemente de o grupo de usuários enviar ou não mensagens de verificação.

Você só pode confirmar a inscrição de autoatendimento de um usuário com essa técnica. Para confirmar um usuário que você criou como administrador, crie uma solicitação de AdminSetUserPasswordAPI com `Permanent` definido como`True`.

1. Um usuário se inscreve no seu aplicativo inserindo um nome de usuário, número de telefone, endereço de and/or e-mail e possivelmente outros atributos.
2. O serviço do Amazon Cognito recebe a solicitação de cadastro da aplicação. Após verificar se a solicitação contém todos os atributos necessários para o cadastramento, o serviço conclui o processo e retorna para o aplicativo a informação de que o cadastramento está concluído e aguarda confirmação. Nesse ponto, a conta de usuário está em um estado não confirmado. O usuário não pode fazer login até que a conta esteja confirmada.
3. Confirme a conta do usuário. Você deve fazer login Console de gerenciamento da AWS ou assinar sua solicitação de API com AWS credenciais para confirmar a conta.

    a. Para confirmar um usuário no console do Amazon Cognito, navegue até o menu Usuários, selecione o usuário que deseja confirmar e, no menu Ações, clique em Confirmar.

    b. Para confirmar um usuário na AWS API ou na CLI, crie uma solicitação de AdminConfirmSignUpAPI ou admin-confirm-sign-upno. AWS CLI
4. Nesse momento, como a conta de usuário está no estado confirmado, o usuário pode fazer login.

Computar valores de hash de segredo

Atribua um segredo do cliente ao cliente da aplicação confidencial como prática recomendada. Quando você atribui um segredo de cliente ao cliente da aplicação, as solicitações de API de grupos de usuários do Amazon Cognito devem incluir um hash que inclua o segredo do cliente no corpo

da solicitação. Para validar seu conhecimento do segredo do cliente para as operações de API nas listas a seguir, concatene o segredo do cliente com o ID do cliente da aplicação e o nome de usuário; depois, codifique essa string em base64.

Quando a aplicação conecta usuários a um cliente que tem um hash secreto, é possível utilizar o valor de qualquer atributo de login do grupo de usuários como o elemento de nome de usuário do hash secreto. Quando a aplicação solicita novos tokens em uma operação de autenticação com `REFRESH_TOKEN_AUTH`, o valor do elemento de nome de usuário depende dos seus atributos de login. Quando o grupo de usuários não tiver `username` como atributo de login, defina o valor do hash secreto do nome do usuário na declaração `sub` do usuário em seu token de acesso ou ID. Quando `username` é um atributo de login, defina o valor do nome de usuário de hash secreto da declaração `username`.

Os seguintes grupos de usuários do Amazon Cognito APIs aceitam um valor de hash secreto do cliente em um parâmetro. `SecretHash`

- ConfirmForgotPassword
- ConfirmSignUp
- ForgotPassword
- ResendConfirmationCode
- SignUp

Além disso, os itens a seguir APIs aceitam um valor de hash secreto do cliente em um `SECRET_HASH` parâmetro, seja em parâmetros de autenticação ou em uma resposta de desafio.

Operação de API	Parâmetro pai para SECRET_HASH
InitiateAuth	AuthParameters
AdminInitiateAuth	AuthParameters
RespondToAuthChallenge	ChallengeResponses
AdminRespondToAuthChallenge	ChallengeResponses

O valor do hash de segredo é um código de autenticação de mensagem baseado em hash (HMAC) de chave codificado em Base64 calculado com o uso da chave secreta de um cliente do grupo

de usuários e do nome de usuário mais o ID do cliente na mensagem. O pseudocódigo a seguir mostra como esse valor é calculado. Nesse pseudocódigo, + indica concatenação, `HMAC_SHA256` representa uma função que produz um valor HMAC usando Hmac e `Base64` representa uma função que produz uma versão codificada em SHA256 Base-64 da saída de hash.

```
Base64 ( HMAC_SHA256 ( "Client Secret Key", "Username" + "Client Id" ) )
```

Para obter uma visão geral detalhada de como calcular e usar o `SecretHash` parâmetro, consulte Como soluciono os erros “Não é possível verificar o hash secreto para o cliente” na minha API de grupos de usuários do Amazon Cognito<client-id>? no Centro de AWS Conhecimento.

Você pode usar os exemplos de código a seguir no código da aplicação do lado do servidor.

Shell

```
echo -n "[username][app client ID]" | openssl dgst -sha256 -hmac [app client secret]
 -binary | openssl enc -base64
```

Java

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;

public static String calculateSecretHash(String userPoolClientId, String
 userPoolClientSecret, String userName) {
    final String HMAC_SHA256_ALGORITHM = "HmacSHA256";

    SecretKeySpec signingKey = new SecretKeySpec(
            userPoolClientSecret.getBytes(StandardCharsets.UTF_8),
            HMAC_SHA256_ALGORITHM);
    try {
        Mac mac = Mac.getInstance(HMAC_SHA256_ALGORITHM);
        mac.init(signingKey);
        mac.update(userName.getBytes(StandardCharsets.UTF_8));
        byte[] rawHmac =
 mac.doFinal(userPoolClientId.getBytes(StandardCharsets.UTF_8));
        return Base64.getEncoder().encodeToString(rawHmac);
    } catch (Exception e) {
        throw new RuntimeException("Error while calculating ");
    }
}
```

Python

```
import sys
import hmac, hashlib, base64
username = sys.argv[1]
app_client_id = sys.argv[2]
key = sys.argv[3]
message = bytes(sys.argv[1]+sys.argv[2],'utf-8')
key = bytes(sys.argv[3],'utf-8')
secret_hash = base64.b64encode(hmac.new(key, message,
 digestmod=hashlib.sha256).digest()).decode()
print("SECRET HASH:",secret_hash)
```

Como confirmar contas de usuários sem verificar e-mail ou número de telefone

O acionador de pré-cadastro do Lambda pode ser usado para confirmar automaticamente as contas de usuário no momento do cadastro sem a necessidade de um código de confirmação nem da verificação do e-mail ou do número de telefone. Usuários que são confirmados dessa maneira podem imediatamente fazer login sem a necessidade de receber um código.

Você também pode marcar um número de telefone ou e-mail do usuário como verificado por meio desse trigger.

> **Note**
>
> Embora essa abordagem seja conveniente para os usuários quando eles estão dando os primeiros passos, recomendamos a verificação automática de pelo menos um dos dois (e-mail ou número de telefone). Caso contrário, o usuário pode ficar impossibilitado de recuperar a senha caso a esqueça.

Se você não exigir que o usuário receba e insira um código de confirmação no cadastro e não verifique automaticamente o e-mail e o número de telefone no acionador de pré-cadastro do Lambda, você correrá o risco de não ter um endereço de e-mail nem um número de telefone verificado para essa conta de usuário. O usuário pode confirmar o endereço de e-mail ou o número de telefone posteriormente. No entanto, se o usuário esquecer a senha e não tiver um número de telefone nem um endereço de e-mail verificado, ele será bloqueado da conta porque o fluxo de senha esquecida exige um número de telefone ou um e-mail verificado para enviar um código de verificação ao usuário.

Como verificar quando usuários alteram o e-mail ou o número de telefone

Os usuários podem inserir um número de telefone ou endereço de e-mail como nome de usuário no login nos grupos de usuários configurados com vários nomes de login. Quando eles atualizam o endereço de e-mail ou número de telefone na sua aplicação, o Amazon Cognito pode enviar imediatamente uma mensagem com um código que confirma a propriedade do novo valor do atributo. Para habilitar o envio automático desses códigos de verificação, consulte Como configurar verificação de e-mail ou telefone.

Os usuários que receberem um código de verificação devem devolvê-lo ao Amazon Cognito em uma VerifyUserAttributesolicitação. Após fornecerem o código, o atributo será marcado como verificado. Normalmente, quando os usuários atualizam seu endereço de e-mail ou número de telefone, é necessário confirmar que eles são proprietários do novo valor antes que possam usá-lo para fazer login e receber mensagens. Os grupos de usuários têm uma opção configurável que define se os usuários devem verificar as atualizações em seu endereço de e-mail ou número de telefone.

Essa opção é a propriedade do grupo de usuários `AttributesRequireVerificationBeforeUpdate`. Configure-o em uma UpdateUserPoolsolicitação CreateUserPoolor ou com a configuração Manter o valor do atributo original ativo quando uma atualização estiver pendente no menu de inscrição do console do Amazon Cognito.

A forma como seu grupo de usuários trata as atualizações de endereços de e-mail e números de telefone está conectada à configuração do nome de usuário do grupo de usuários. Os nomes de usuário do grupo de usuários podem estar em uma configuração de atributos de nome de usuário em que os nomes de login são endereço de e-mail, número de telefone ou ambos. Eles também podem estar em uma configuração de atributos de alias em que o atributo `username` é um nome de login com endereço de e-mail, número de telefone ou nome de usuário preferencial como nomes de login alternativos. Para obter mais informações, consulte Personalização dos atributos de login.

Você também pode usar um acionador de mensagem personalizada do Lambda para personalizar a mensagem de verificação. Para obter mais informações, consulte Acionador do Lambda de mensagem personalizada. Quando o endereço de e-mail ou o número de telefone do usuário não estiver verificado, sua aplicação deverá informar ao usuário que ele precisa verificar o atributo e fornecer um botão ou um link para que ele insira o código de verificação.

A tabela a seguir descreve como `AttributesRequireVerificationBeforeUpdate` e as configurações de alias determinam o resultado quando os usuários alteram o valor de seus atributos de login.

Configuração do nome de usuário	Comportamento quando os usuários precisam verificar novos atributos	Comportamento quando os usuários não precisam verificar novos atributos
Atributos do nome de usuário	O atributo original permanece verificado, elegível para login e com o valor original. Quando o usuário verifica um novo valor, o Amazon Cognito atualiza o valor do atributo, marca-o como verificado e o torna elegível para login.	O Amazon Cognito atualiza o atributo para um novo valor. O novo valor é elegível para login. Quando o usuário verifica um novo valor, o Amazon Cognito o marca como verificado.
Atributos de alias	O atributo original permanece verificado, elegível para login e com o valor original. Quando o usuário verifica um novo valor, o Amazon Cognito atualiza o valor do atributo, marca-o como verificado e o torna elegível para login.	O Amazon Cognito atualiza o atributo para um novo valor. Nem o valor original nem o novo valor do atributo são elegíveis para login. Quando o usuário verifica um novo valor, o Amazon Cognito atualiza o valor do atributo, marca-o como verificado e o torna elegível para login.

Exemplo 1

O usuário 1 faz login na sua aplicação com o endereço de e-mail `user1@example.com` e tem o nome de usuário `user1` (atributos de alias). Seu grupo de usuários está configurado para verificar as atualizações nos atributos de login e enviar mensagens de verificação automaticamente. Ele solicita a atualização do endereço de e-mail para `user1+foo@example.com`. Ele recebe um e-mail de verificação em `user1+foo@example.com` e só pode fazer login novamente com o endereço de e-mail `user1@example.com`. Posteriormente, ele insere o código de verificação e só pode fazer login novamente com o endereço de e-mail `user1+foo@example.com`.

Exemplo 2

O usuário 2 faz login na sua aplicação com o endereço de e-mail `user2@example.com` e tem um nome de usuário (atributos de alias). Seu grupo de usuários está configurado para não verificar as atualizações nos atributos de login e para enviar mensagens de verificação automaticamente. Ele solicita a atualização do endereço de e-mail para `user2+bar@example.com`. Ele recebe um e-mail de verificação em `user2+bar@example.com` e não conseguem fazer login novamente. Posteriormente, ele insere o código de verificação e só pode fazer login novamente com o endereço de e-mail `user2+bar@example.com`.

Exemplo 3

O usuário 3 faz login na sua aplicação com o endereço de e-mail `user3@example.com` e não tem um nome de usuário (atributos de nome de usuário). Seu grupo de usuários está configurado para não verificar as atualizações nos atributos de login e para enviar mensagens de verificação automaticamente. Ele solicita a atualização do endereço de e-mail para `user3+baz@example.com`. Ele recebe um e-mail de verificação em `user3+baz@example.com`, mas pode fazer login imediatamente sem nenhuma ação adicional com o código de verificação.

Processos de confirmação e verificação para contas de usuários criadas por administradores ou desenvolvedores

As contas de usuários que foram criadas por um administrador ou desenvolvedor já ficam no estado confirmado para que os usuários não precisem inserir um código de confirmação. A mensagem de convite que o serviço do Amazon Cognito envia para esses usuários inclui o nome de usuário e uma senha temporária. O usuário precisa alterar a senha antes de fazer login. Para obter mais informações, consulte Personalizar mensagens de e-mail e de SMS em Como criar contas de usuário como administrador e o trigger de mensagem personalizada em Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda.

Processos de confirmação e verificação para contas de usuários importadas

As contas de usuário criadas usando o recurso de importação de usuários na Console de gerenciamento da AWS CLI ou na API (consulteComo importar usuários para grupos de usuários com base em um arquivo CSV) já estão no estado confirmado, portanto, os usuários não precisam inserir um código de confirmação. Nenhuma mensagem de convite é enviada. No entanto, as contas de usuário importadas exigem que os usuários primeiro solicitem um código chamando a API `ForgotPassword` e, em seguida, criem uma senha usando o código entregue chamando a API `ConfirmForgotPassword` antes de fazer login. Para obter mais informações, consulte Solicitação de redefinição de senha aos usuários importados.

Quando a conta de usuário é importada, o número de telefone ou o e-mail do usuário deve ser marcado como confirmado de modo que nenhuma verificação seja necessária quando o usuário fizer login.

Como enviar e-mails enquanto testa sua aplicação

O Amazon Cognito envia e-mails aos usuários quando eles criam e gerenciam suas contas na aplicação cliente para o grupo de usuários. Se você configurar o grupo de usuários para solicitar verificação por e-mail, o Amazon Cognito enviará um e-mail quando:

- Um usuário se cadastrar.
- Um usuário atualizar o endereço de e-mail.
- Um usuário realizar uma ação que chama a ação de API `ForgotPassword`.
- Você criar uma conta de usuário como um administrador.

Dependendo da ação que inicia o e-mail, o e-mail contém um código de verificação ou uma senha temporária. Os usuários devem receber esses e-mails e compreender a mensagem. Caso contrário, eles podem não conseguir fazer login e usar seu aplicativo.

Para garantir que os e-mails sejam enviados com êxito e que a mensagem pareça correta, teste na sua aplicação as ações que iniciam entregas de e-mail no Amazon Cognito. Por exemplo, usando a página de cadastro do seu aplicativo, ou usando a ação de API `SignUp`, é possível iniciar um e-mail cadastrando-se com um endereço de e-mail de teste. Ao testar dessa forma, lembre-se do seguinte:

> **Importante**
>
> Ao usar um endereço de e-mail para testar ações que iniciam e-mails no Amazon Cognito, não use um endereço de e-mail falso (um que não tenha caixa de correio). Use um endereço de e-mail real que receberá o e-mail do Amazon Cognito sem criar uma devolução definitiva. Uma devolução definitiva ocorre quando o Amazon Cognito deixa de entregar o e-mail para a caixa de correio do destinatário, o que sempre ocorre se a caixa de correio não existe.
> O Amazon Cognito limita o número de e-mails que podem ser enviados por AWS contas que incorrem em rejeições difíceis de forma persistente.

Ao testar ações que iniciam e-mails, use um dos seguintes endereços de e-mail para evitar devoluções definitivas:

- Um endereço para uma conta de e-mail que você possui e usa para testes. Ao usar seu próprio endereço de e-mail, você recebe o e-mail que o Amazon Cognito envia. Com esse e-mail, você pode usar o código de verificação para testar a experiência de cadastro no seu aplicativo. Se você personalizou a mensagem de e-mail para o grupo de usuários, pode verificar se as personalizações estão corretas.
- O endereço do simulador de caixa postal, success@simulator.amazonses.com. Se você usar o endereço do simulador, o Amazon Cognito enviará o e-mail com êxito, mas você não conseguirá visualizá-lo. Essa opção é útil quando você não precisa usar o código de verificação e não precisa verificar a mensagem de e-mail.
- O endereço do simulador de caixa postal com a adição de um rótulo arbitrário, como success+user1@simulator.amazonses.com ou success+user2@simulator.amazonses.com. O Amazon Cognito enviará e-mails para esses endereços com êxito, mas você não conseguirá visualizá-los. Essa opção é útil quando você deseja testar o processo de cadastro adicionando vários usuários de teste ao grupo de usuários, e cada usuário de teste tem um endereço de e-mail exclusivo.

Como configurar verificação de e-mail ou telefone

Você pode selecionar as configurações para verificação de e-mail ou telefone no menu Métodos de autenticação. Para obter mais informações sobre a autenticação multifator (MFA), consulte MFA de mensagem de texto SMS.

O Amazon Cognito usa o Amazon SNS para enviar mensagens SMS. Se você ainda não enviou uma mensagem SMS do Amazon Cognito ou de qualquer outra AWS service (Serviço da AWS) , o Amazon SNS pode colocar sua conta na sandbox de SMS. Recomendamos que você envie uma mensagem de teste para um número de telefone verificado antes de remover sua conta da sandbox para a produção. Além disso, se você pretende enviar mensagens SMS para números de telefone dos EUA, deve obter um ID de origem ou de remetente do Amazon Pinpoint. Para configurar seu grupo de usuários do Amazon Cognito para mensagens SMS, consulte Configurações de mensagens SMS para grupos de usuários do Amazon Cognito.

O Amazon Cognito pode verificar automaticamente os endereços de e-mail ou os números de telefone. Para fazer essa verificação, o Amazon Cognito envia um código de verificação ou um link de verificação. No caso dos endereços de e-mail, o Amazon Cognito envia um código ou um link em uma mensagem de e-mail. Você pode escolher um Tipo de verificação de Código ou Link ao editar seu modelo de Mensagem de verificação no menu Modelos de mensagens no console do Amazon Cognito. Para obter mais informações, consulte Personalizar mensagens de verificação de e-mail.

Para números de telefone, o Amazon Cognito envia um código em uma mensagem de texto SMS.

O Amazon Cognito deve verificar um número de telefone ou endereço de e-mail para confirmar os usuários e ajudá-los a recuperar senhas esquecidas. Como alternativa, você pode confirmar automaticamente os usuários com o gatilho Lambda de pré-inscrição ou usar AdminConfirmSignUpa operação da API. Para obter mais informações, consulte Como cadastrar e confirmar contas de usuários.

O código de verificação ou link tem validade de 24 horas.

Se você optar por exigir verificação de um endereço de e-mail ou número de telefone, o Amazon Cognito enviará automaticamente o código de verificação ou o link quando um usuário se cadastrar. Se o grupo de usuários tiver um Acionador do Lambda de remetente personalizado de SMS ou Acionador do Lambda de remetente de e-mail personalizado configurado, essa função será invocada.

Observações

- O Amazon SNS cobra separadamente por mensagens de texto SMS que ele usa para verificar números de telefone. Não há encargos para envio de mensagens de e-mail. Para obter informações sobre preços do Amazon SNS, consulte Worldwide SMS pricing (Preço global de SMS). Para obter a lista atualizada de países nos quais o sistema de mensagens SMS está disponível, consulte Supported regions and countries (Países e regiões compatíveis).
- Quando você testar ações na aplicação que geram mensagens de e-mail do Amazon Cognito, use um endereço de e-mail real acessível ao Amazon Cognito sem devoluções definitivas. Para obter mais informações, consulte the section called “Como enviar e-mails enquanto testa sua aplicação”.
- O fluxo de senha esquecida requer o e-mail ou o número de telefone do usuário para verificar o usuário.

Important

Se um usuário se cadastrar com um número de telefone e um endereço de e-mail e suas configurações de grupo de usuários exigirem a verificação dos dois atributos, o Amazon Cognito enviará um código de verificação ao telefone por uma mensagem SMS. O Amazon Cognito ainda não verificou o endereço de e-mail, então seu aplicativo deve ligar

GetUserpara ver se um endereço de e-mail aguarda verificação. Se precisar de verificação, o aplicativo deverá ligar GetUserAttributeVerificationCodepara iniciar o fluxo de verificação de e-mail. Em seguida, ele deve enviar o código de verificação ligando VerifyUserAttribute.

Você pode ajustar sua cota de gastos de mensagens SMS para uma Conta da AWS e para mensagens individuais. Os limites se aplicam somente ao custo do envio de mensagens SMS. Para obter mais informações, consulte O que são cotas de gastos em nível de conta e de mensagem e como elas funcionam? no Amazon SNS FAQs.

O Amazon Cognito envia mensagens SMS usando recursos do Amazon SNS no local em que você criou Região da AWS o grupo de usuários ou em uma região alternativa legada do Amazon SNS da tabela a seguir. A exceção são grupos de usuários do Amazon Cognito na região da Ásia-Pacífico (Seul). Esses grupos de usuários usam a configuração do Amazon SNS na região da Ásia-Pacífico (Tóquio). Para obter mais informações, consulte Escolha o Região da AWS para mensagens SMS.

Região do Amazon Cognito	Região alternativa herdada do Amazon SNS
Leste dos EUA (Ohio)	Leste dos EUA (Norte da Virgínia)
Ásia-Pacífico (Mumbai)	Ásia-Pacífico (Singapura)
Ásia-Pacífico (Seul)	Ásia-Pacífico (Tóquio)
Canadá (Central)	Leste dos EUA (Norte da Virgínia)
Europa (Frankfurt)	Europa (Irlanda)
Europa (Londres)	Europa (Irlanda)

Exemplo:se o grupo de usuários do Amazon Cognito estiver na Ásia-Pacífico (Mumbai) e você tiver aumentado o limite de gastos em ap-southeast-1, é provável que não queira solicitar um aumento separado em ap-south-1. Em vez disso, você pode usar os recursos do Amazon SNS na Ásia-Pacífico (Singapura).

Verificar atualizações de endereços de e-mail e números de telefone

Um atributo de endereço de e-mail ou número de telefone pode se tornar ativo e não verificado imediatamente depois que o usuário alterar o respectivo valor. O Amazon Cognito também pode

exigir que seu usuário verifique o novo valor antes que o Amazon Cognito atualize o atributo. Quando você exigir que seus usuários confirmem primeiro o novo valor, eles poderão usar o valor original para fazer login e receber mensagens até confirmarem o novo valor.

Quando os usuários podem usar o endereço de e-mail ou o número de telefone como um alias de login no grupo de usuários, o nome de login para um atributo atualizado é condicionado à exigência de verificação de atributos atualizados. Quando você exigir que os usuários confirmem um atributo atualizado, um usuário poderá fazer login com o valor do atributo original até verificar o novo valor. Quando você não exigir que os usuários confirmem um atributo atualizado, um usuário não poderá fazer login nem receber mensagens no valor do atributo novo nem no original até confirmar o novo valor.

Por exemplo, seu grupo de usuários permite o login com um alias de endereço de e-mail e exige que os usuários confirmem seu endereço de e-mail quando realizam uma atualização. Sue, que faz login como `sue@example.com`, quer alterar o endereço de e-mail para`sue2@example.com`, mas faz login como **s**`sue2@example.com` acidentalmente. Sue não recebe o e-mail de confirmação, então não consegue confirmar o e-mail `ssue2@example.com`. Sue faz login como `sue@example.com` e reenvia o formulário em sua aplicação para atualizar o endereço de e-mail para `sue2@example.com`. Ela recebe esse e-mail, fornece o código de verificação à aplicação e começa a fazer login como `sue2@example.com`.

Quando o usuário atualiza um atributo e o grupo de usuários verifica novos valores de atributos

- É possível fazer login com o valor do atributo original antes de confirmar o código para verificar o novo valor.
- É possível fazer login somente com o novo valor do atributo depois de confirmar o código para verificar o novo valor.
- Se você `phone_number_verified` definir `email_verified` `true` ou ativar uma solicitação de AdminUpdateUserAttributesAPI, eles poderão fazer login antes de confirmarem o código que o Amazon Cognito enviou a eles.

Quando um usuário atualiza um atributo e o grupo de usuários não verifica novos valores de atributos

- Não é possível fazer login nem receber mensagens com o valor do atributo original.
- Não é possível fazer login nem receber mensagens que não sejam um código de confirmação com o valor do novo atributo antes de confirmar o código para verificar o novo valor.

- Se você `phone_number_verified` definir `email_verified` `true` ou ativar uma solicitação de AdminUpdateUserAttributesAPI, eles poderão fazer login antes de confirmarem o código que o Amazon Cognito enviou a eles.

Para exigir verificação de atributos quando os usuários atualizam o endereço de e-mail ou o número de telefone

1. Faça login no console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. No painel de navegação, escolha User Pools (Grupos de usuários) e escolha o grupo de usuários que deseja editar.
3. No menu Cadastrar-se, selecione Editar em Verificação de atributo e confirmação da conta do usuário.
4. Selecione Keep original attribute value active when an update is pending (Manter valor do atributo original ativo quando uma atualização estiver pendente).
5. Em Active attribute values when an update is pending (Valores de atributos ativos quando uma atualização está pendente), escolha os atributos que você deseja exigir que seus usuários confirmem antes que o Amazon Cognito atualize o valor.
6. Escolha Salvar alterações.

Para exigir a verificação da atualização de atributos com a API do Amazon Cognito, você pode definir o `AttributesRequireVerificationBeforeUpdate` parâmetro em uma UpdateUserPoolsolicitação.

Como autorizar o Amazon Cognito a enviar mensagens SMS em seu nome

Para enviar mensagens SMS aos usuários em seu nome, o Amazon Cognito precisa da sua permissão. Para conceder essa permissão, você pode criar uma função AWS Identity and Access Management (IAM). No menu Métodos de autenticação do console do Amazon Cognito em SMS, selecione Editar para definir uma função.

Configurar mensagens de MFA, autenticação, verificação e convite

Com o Amazon Cognito, você pode personalizar mensagens de autenticação, verificação e convite de usuários por SMS e e-mail para aprimorar a segurança e a experiência do usuário da aplicação. Para algumas mensagens, você pode escolher entre verificações baseadas em código e verificações por link de acesso com um clique. Este tópico aborda como você pode personalizar as comunicações de autenticação e verificação no console do Amazon Cognito.

No menu Modelos de mensagens, você pode personalizar:

- Seus modelos de e-mail e SMS para autenticação por senha de uso único (OTP) e autenticação multifator (MFA)
- Mensagens de verificação de SMS e e-mail
- O tipo de verificação para e-mail: código ou link

> **Note**
>
> O Amazon Cognito envia links com seu modelo baseado em links nas mensagens de verificação quando os usuários se cadastram ou reenviam um código de confirmação. Os e-mails das operações de atualização de atributos e redefinição de senha usam o modelo de código.

- Mensagens de convite a usuários
- Endereços de e-mail FROM (Remetente) e REPLY-TO (Responder para) para e-mails que passam pelo seu grupo de usuários

> **Note**
>
> Os modelos de mensagem de verificação por SMS e e-mail só serão exibidos se você tiver optado por exigir a verificação de número de telefone e e-mail. De maneira semelhante, o modelo de mensagem SMS de MFA só aparece se a configuração de MFA for required (obrigatória) ou optional (opcional).

Tópicos

Modelos de mensagens

É possível usar modelos de mensagens para inserir espaços reservados em suas mensagens. O Amazon Cognito substitui os espaços reservados pelos valores correspondentes. Você pode consultar Espaços reservados para modelos universais em modelos de mensagem de qualquer tipo, embora esses valores não estejam presentes em todos os tipos de mensagens.

Espaços reservados para modelos universais

Description	Token	Tipo de mensagem
Código de verificação	`{####}`	Mensagens de verificação, confirmação e MFA
Senha temporária	`{####}`	Mensagens de senha esquecida e de convite
Nome do usuário	`{username}`	Mensagens de convite e de segurança avançada

Uma das respostas automatizadas disponíveis com proteção contra ameaças é notificar o usuário de que o Amazon Cognito detectou atividades possivelmente mal-intencionadas. Você pode usar espaços reservados de modelo de segurança avançada para fazer o seguinte:

- Inclua detalhes específicos sobre um evento, como endereço IP, cidade, país, hora do login e nome do dispositivo. A proteção contra ameaças do Amazon Cognito pode analisar esses detalhes.
- Verificar se um link de um clique é válido.
- Use o ID do evento, o token de feedback e o nome de usuário para seu próprio link de um clique.

> **Note**
>
> Para gerar links com um clique e usar os espaços reservados `{one-click-link-valid}` e `{one-click-link-invalid}` em modelos de e-mail de segurança avançada, será necessário ter um domínio já configurado para o grupo de usuários.

A proteção contra ameaças adiciona os espaços reservados abaixo que você pode inserir nos modelos de mensagem. Esses espaços reservados se aplicam às Mensagens de autenticação adaptável, notificações que o Amazon Cognito envia aos usuários cujas sessões foram avaliadas quanto ao nível de risco. Para configurar modelos de mensagens com essas variáveis, atualize a configuração de função completa da sua proteção contra ameaças no console do Amazon Cognito ou envie modelos em SetRiskConfigurationuma solicitação.

Espaços reservados de modelo de segurança avançada

Description	Token
IP address (endereço de IP)	`{ip-address}`
Cidade	`{city}`
País	`{country}`
Tempo de login	`{login-time}`
Nome do dispositivo	`{device-name}`
O link de um clique é válido	`{one-click-link-valid}`
O link de um clique não é válido	`{one-click-link-invalid}`
ID do evento	`{event-id}`
Token do feedback	`{feedback-token}`

Personalizar mensagens de MFA enviadas por e-mail e SMS

Para personalizar as mensagens de autenticação multifator (MFA) enviadas por SMS e e-mail, edite Mensagem de MFA no menu Modelos de mensagens no console de grupos de usuários do Amazon Cognito.

> **Important**
>
> Sua mensagem personalizada deve conter o espaço reservado `{####}`. Esse espaço reservado é substituído pelo código de autenticação antes de a mensagem ser enviada.

O Amazon Cognito define um limite máximo de 140 caracteres UTF-8 para mensagens SMS, incluindo o código de autenticação.

Como personalizar mensagens SMS de verificação

Para personalizar a mensagem SMS para verificação do número de telefone, edite o modelo Mensagem de verificação no menu Modelos de mensagens do seu grupo de usuários.

> **Important**
>
> Sua mensagem personalizada deve conter o espaço reservado `{####}`. Esse espaço reservado é substituído pelo código de verificação antes de a mensagem ser enviada.

O comprimento máximo da mensagem, incluindo o código de verificação, é de 140 caracteres em UTF-8.

Personalizar mensagens de verificação de e-mail

Para verificar o endereço de e-mail de um usuário em seu grupo de usuários com o Amazon Cognito, você pode enviar ao usuário uma mensagem de e-mail com um link que pode ser selecionado ou enviar um código que possa ser inserido.

Para personalizar o assunto do e-mail e o conteúdo das mensagens de verificação de endereço de e-mail, edite o modelo de Mensagem de verificação no menu Modelos de mensagens de seu grupo de usuários. Você pode selecionar um Tipo de verificação de Código ou Link ao editar o modelo de Mensagem de verificação.

Se você escolher Código como tipo de verificação, sua mensagem personalizada deverá conter o espaço reservado `{####}`. Ao enviar a mensagem, o código de verificação substitui esse espaço reservado.

Se você escolher Link como o tipo de verificação, sua mensagem personalizada deverá conter o espaço reservado no formato `{##Verify Your Email##}`. Você pode alterar a string de texto entre os caracteres do espaço reservado, por exemplo, `{##Click here##}`. Um link de verificação intitulado Verify Your Email (Verificar seu e-mail) substitui esse espaço reservado.

O link para uma mensagem de verificação por e-mail direciona o usuário para um URL como no exemplo a seguir.

```
https://<your user pool domain>/confirmUser/?
client_id=abcdefg12345678&user_name=emailtest&confirmation_code=123456
```

O comprimento máximo da mensagem, incluindo o código de verificação (se estiver presente), é de 20.000 caracteres em UTF-8. Você pode usar etiquetas HTML nessa mensagem para formatar o conteúdo.

Como personalizar mensagens de convite a usuários

É possível personalizar a mensagem de convite a usuários enviada pelo Amazon Cognito aos novos usuários por SMS ou e-mail editando o modelo de Mensagens de convite no menu Modelos de mensagens.

> **Important**
>
> Sua mensagem personalizada deve conter os espaços reservados `{username}` e `{####}`. Quando o Amazon Cognito envia a mensagem de convite, ele substitui esses espaços reservados pelo nome de usuário pela senha do usuário.

O comprimento máximo da mensagem SMS, incluindo o código de verificação, é de 140 caracteres em UTF-8. O comprimento máximo da mensagem de e-mail, incluindo o código de verificação, é de 20.000 caracteres em UTF-8. Você pode usar etiquetas HTML nas suas mensagem de e-mail para formatar o conteúdo.

Personalizar o endereço de e-mail

Por padrão, o Amazon Cognito envia mensagens de e-mail aos usuários dos seus grupos de usuários do endereço no-reply@verificationemail.com. É possível escolher especificar endereços de e-mail FROM (Remetente) e REPLY-TO (Responder para) personalizados em vez de no-reply@verificationemail.com.

Como personalizar os endereços de e-mail FROM e REPLY-TO

1. Acesse o console do Amazon Cognito e escolha User Pools (Grupos de usuários).
2. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
3. Clique no menu Métodos de autenticação. Em Email (E-mail), escolha Edit (Editar).
4. Escolha uma SES Region (Região do SES).

5. Escolha um FROM email address (Endereço de e-mail do remetente) na lista de endereços de e-mail que você verificou com o Amazon SES na SES Region (Região do SES) selecionada por você. Para usar um endereço de e-mail de um domínio verificado, defina as configurações de e-mail na AWS Command Line Interface ou na AWS API. Para mais informações, consulte Verificar endereços de e-mail e domínios no Amazon SES no Guia do desenvolvedor do Amazon Simple Email Service.
6. Escolha um Configuration set (Conjunto de configurações) na lista de conjuntos de configurações na SES Region (Região SES) escolhida.
7. Insira um FROM sender name (Nome do remetente) amigável para suas mensagens de e-mail, no formato `John Stiles <johnstiles@example.com>`.
8. Para personalizar o endereço de e-mail do destinatário, insira um endereço de e-mail válido no campo Endereço de e-mail do destinatário.

Como autorizar o Amazon Cognito a enviar e-mails do Amazon SES em seu nome (de um endereço de e-mail remetente personalizado)

É possível configurar o Amazon Cognito para enviar e-mails de um endereço de e-mail remetente personalizado ao invés do seu endereço padrão. Para usar um endereço personalizado, você deve conceder permissão para que o Amazon Cognito envie mensagens de e-mail de uma identidade verificada do Amazon SES. Na maioria dos casos, é possível conceder essa permissão criando uma política de autorização de envio. Para mais informações, consulte Usar autorização de envio com o Amazon SES no Guia do desenvolvedor do Amazon Simple Email Service.

Quando você configura um grupo de usuários para usar o Amazon SES para mensagens de e-mail, o Amazon Cognito cria a função `AWSServiceRoleForAmazonCognitoIdpEmailService` em sua conta para conceder acesso ao Amazon SES. Nenhuma política de autorização de envio é necessária quando a função vinculada ao serviço `AWSServiceRoleForAmazonCognitoIdpEmailService` é usada. Você só precisa adicionar uma política de autorização de envio quando usar a funcionalidade de e-mail padrão em seu grupo de usuários e uma identidade verificada do Amazon SES como o endereço remetente.

Para obter mais informações sobre a função vinculada ao serviço criada pelo Amazon Cognito, consulte Como usar funções vinculadas a serviço para o Amazon Cognito.

O exemplo de política de autorização de envio a seguir concede ao Amazon Cognito uma capacidade limitada de usar uma identidade verificada do Amazon SES. O Amazon Cognito só pode enviar mensagens de e-mail quando o fizer em nome do grupo de usuários na condição

`aws:SourceArn` e da conta na condição `aws:SourceAccount`. Para mais exemplos, consulte Exemplos de política de autorização de envio do Amazon SES no Guia do desenvolvedor do Amazon Simple Email Service.

> **Note**
>
> Neste exemplo, o valor "Sid" é uma string arbitrária que identifica exclusivamente a instrução. Para mais informações sobre sintaxe de políticas, consulte Políticas de autorização de envio do Amazon SES no Guia do desenvolvedor do Amazon Simple Email Service.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "stmnt1234567891234",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "email.cognito-idp.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "arn:aws:ses:us-
east-1:111122223333:identity/support@example.com",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:cognito-idp:us-
east-1:111122223333:userpool/us-east-1_EXAMPLE"
                }
            }
        }
    ]
```

```
}
```

O console do Amazon Cognito adiciona uma política semelhante para você quando você seleciona uma identidade do Amazon SES do menu suspenso. Se usar a CLI ou a API para configurar o grupo de usuários, você deverá anexar uma política estruturada como o exemplo anterior à sua identidade do Amazon SES.

Como criar contas de usuário como administrador

Os grupos de usuários não são apenas um diretório de usuários do gerenciamento de identidade e acesso do cliente (CIAM), em que qualquer pessoa na Internet pode cadastrar um perfil de usuário na aplicação. Você pode desativar o cadastro por autoatendimento. Talvez você já conheça seus clientes e queira admitir apenas aqueles que foram autorizados com antecedência. Você pode colocar barreiras de proteção na autenticação manual da aplicação com um provedor de identidades SAML 2.0 ou OIDC privado, importando usuários, examinando usuários no momento do cadastro ou criando usuários com operações administrativas de API. Seu fluxo de trabalho para criação administrativa de usuários pode ser programático, provisionando usuários após o registro em outro sistema, ou pode ser baseado em testes case-by-case ou no console do Amazon Cognito.

Quando você cria usuários como administrador, o Amazon Cognito define uma senha temporária para eles e envia uma mensagem de boas-vindas ou convite. Eles podem seguir o link na mensagem de convite e fazer login pela primeira vez, definindo uma senha e confirmando a conta. A página a seguir descreve como criar usuários e configurar a mensagem de boas-vindas. Para obter mais informações sobre a criação de usuários com a API de grupos de usuários e um AWS SDK ou CDK, consulte. AdminCreateUser

Depois de criar seu grupo de usuários, você pode criar usuários usando a Console de gerenciamento da AWS, bem como a API do AWS Command Line Interface Amazon Cognito. Você pode criar um perfil para um novo usuário em um grupo de usuários e enviar uma mensagem de boas-vindas com instruções de cadastro por SMS ou e-mail.

Veja a seguir alguns exemplos de como os administradores podem gerenciar usuários em grupos de usuários.

- Crie um novo perfil de usuário no console do Amazon Cognito ou com a operação de API `AdminCreateUser`.
- Disponibilize fluxos de autenticação sem senha username-and-password, chave de acesso e personalizados para seu grupo de usuários e cliente de aplicativos.

- Defina os valores dos atributos do usuário.
- Crie atributos personalizados.
- Defina o valor dos atributos personalizados imutáveis nas solicitações de API `AdminCreateUser`. Esse recurso não está disponível no console do Amazon Cognito.
- Especifique uma senha temporária, crie um usuário sem senha ou permita que o Amazon Cognito gere automaticamente uma senha.
- Crie novos usuários e confirme automaticamente suas contas, verifique seus endereços de e-mail ou verifique seus números de telefone.
- Especifique mensagens personalizadas de convite por SMS e e-mail para novos usuários por meio dos acionadores Console de gerenciamento da AWS ou Lambda, como mensagem personalizada, remetente de SMS personalizado e remetente de e-mail personalizado.
- Especifique se as mensagens de convite serão enviadas por SMS, e-mail ou ambos.
- Reenvie a mensagem de boas-vindas a um usuário existente chamando a API `AdminCreateUser`, especificando `RESEND` para o parâmetro `MessageAction`.
- Suprima o envio da mensagem de convite quando o usuário for criado.
- Especifique um limite de tempo de expiração de até 90 dias para novas contas de usuário.
- Permita que os usuários se cadastrem ou exija que novos usuários sejam adicionados apenas pelo administrador.

Os administradores também podem conectar usuários com AWS credenciais em um aplicativo do lado do servidor. Para obter mais informações, consulte Modelos de autorização para autenticação de API e SDK.

Fluxos de autenticação de usuários e criação de usuários

A criação administrativa de usuários tem opções que diferem com base na configuração do seu grupo de usuários. Os fluxos de autenticação, ou métodos disponíveis aos usuários para login e MFA, podem alterar a forma como você cria usuários e as mensagens enviadas a eles. Veja a seguir alguns fluxos de autenticação disponíveis em grupos de usuários.

- Nome de usuário e senha
- Chaves de acesso
- Faça login com terceiros IdPs
- Sem senha com senhas de uso único por e-mail e SMS () OTPs

- Autenticação multifatorial com e-mail, SMS e aplicativo autenticador OTPs
- Autenticação personalizada com acionadores do Lambda

Para obter mais informações sobre como configurar esses fatores de login, consulte Autenticação com grupos de usuários do Amazon Cognito.

Criar usuários sem senhas

Se o login sem senha estiver habilitado para seu grupo de usuários, você poderá criar usuários sem senhas. Forneça valores de atributos para um fator de login sem senha disponível para criar um usuário sem uma senha. Por exemplo, se o login sem senha com OTP enviada por e-mail estiver disponível em seu grupo de usuários, você poderá criar um usuário sem senha e com um atributo de endereço de e-mail. Se os únicos fluxos de autenticação disponíveis para novos usuários exigirem uma senha, por exemplo, chave de acesso ou nome de usuário e senha, você deverá criar ou gerar uma senha temporária para cada novo usuário.

Como criar um novo usuário sem uma senha

- Selecione Não defina uma senha no console do Amazon Cognito
- Omita ou deixe em branco o parâmetro `TemporaryPassword` da solicitação de API `AdminCreateUser`

Usuários sem senhas são confirmados automaticamente

Normalmente, novos usuários recebem uma senha temporária e entram em um status `FORCE_CHANGE_PASSWORD` ao serem criados. Quando você cria usuários sem senhas, eles imediatamente entram em um estado `CONFIRMED`. Você não pode reenviar códigos de confirmação para esses usuários no estado `CONFIRMED`.

As mensagens de convite mudam para usuários sem senhas.

Por padrão, o Amazon Cognito envia uma mensagem de convite para novos usuários que diz: `Your username is {userName} and your password is {####}.` Quando você cria usuários sem senha, a mensagem diz: `Your username is {userName}.` Personalize sua mensagem de convite para refletir se você definirá senhas para os usuários. Omita a variável de senha `{####}` nos modelos de autenticação sem senha.

Não é possível gerar senhas automaticamente quando fatores sem senha estão disponíveis

Se configurou o grupo de usuários para oferecer suporte ao login sem senha com OTP enviada por e-mail ou telefone, você não conseguirá gerar uma senha automaticamente. Para os usuários que terão uma senha, você deverá definir uma senha temporária ao criar os perfis deles.

Os usuários sem senha devem ter valores para todos os atributos obrigatórios

Quando você cria um usuário sem uma senha, sua solicitação só é bem-sucedida se o usuário fornecer valores para todos os atributos que você marcou como obrigatórios em seu grupo de usuários. Isso se aplica a qualquer atributo obrigatório, não somente aos atributos de número de telefone e e-mail obrigatórios para a entrega da OTP.

Criar usuários que fornecerão valores de atributos obrigatórios posteriormente

É recomendável exigir atributos em seu grupo de usuários. No entanto, colete esses atributos após a criação administrativa de usuários, durante a interação do usuário na aplicação. Os administradores podem omitir valores para os atributos obrigatórios ao criar usuários com senhas temporárias. Você não pode omitir valores de atributos obrigatórios para usuários sem senha.

Os usuários com valores ausentes para os atributos obrigatórios e uma senha temporária recebem um desafio NEW_PASSWORD_REQUIRED no primeiro login. Em seguida, eles podem informar um valor para os atributos obrigatórios ausentes no parâmetro `requiredAttributes`. Você pode criar usuários com senhas e sem atributos obrigatórios somente se todos os atributos obrigatórios forem mutáveis. Os usuários só podem concluir o login com desafios `NEW_PASSWORD_REQUIRED` e valores de atributos obrigatórios se esses atributos forem graváveis pelo cliente de aplicação que eles usam para fazer login.

Ao definir uma senha permanente para um usuário criado pelo administrador, o status muda para `CONFIRMED` e seu grupo de usuários não solicita uma nova senha ou atributos obrigatórios no primeiro login.

Criando um novo usuário no Console de gerenciamento da AWS

É possível definir requisitos de senha de usuário, configurar as mensagens de convite e verificação enviadas aos usuários e adicionar novos usuários com o console do Amazon Cognito.

Definir uma política de senha e habilitar a autoinscrição

Você pode definir as configurações para a complexidade mínima da senha e se os usuários podem se inscrever usando public APIs em seu grupo de usuários.

Configurar uma política de senhas

1. Acesse o console do Amazon Cognito e escolha User Pools (Grupos de usuários).
2. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
3. Clique no menu Métodos de autenticação e localize Política de senha. Escolha Editar.
4. Selecione o Password policy mode (Modo de política de senha) Custom (Personalizado).
5. Selecione um Password minimum length (Comprimento mínimo da senha). Para os limites do requisito de tamanho da senha, consulte Cotas de recursos de grupos de usuários.
6. Selecione um requisito de Password complexity (Complexidade de senha).
7. Escolha por quanto tempo a senha definida pelos administradores deve ser válida.
8. Escolha Salvar alterações.

Permitir cadastro por autoatendimento

1. Acesse o console do Amazon Cognito e escolha User Pools (Grupos de usuários).
2. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
3. Clique no menu Cadastrar-se e localize Cadastro por autoatendimento. Selecione Edit (Editar).
4. Escolha se deseja Enable self-registration (Habilitar a autoinscrição). O autorregistro geralmente é usado com clientes de aplicativos públicos que precisam registrar novos usuários em seu grupo de usuários sem distribuir um segredo de cliente ou credenciais de API AWS Identity and Access Management (IAM).

   > **Como desabilitar a autoinscrição**
   >
   > Se você não habilitar a autoinscrição, os novos usuários deverão ser criados por ações administrativas da API usando credenciais da API do IAM ou mediante login com provedores federados.

5. Escolha Salvar alterações.

Personalizar mensagens de e-mail e de SMS

Personalizar mensagens de usuário

É possível personalizar as mensagens que o Amazon Cognito envia aos seus usuários quando você os convida para o acesso, eles se cadastram em uma conta de usuário ou acessam e são solicitados a fazer a autenticação multifator (MFA).

> **Note**
>
> Uma Invitation message (Mensagem de convite) é enviada quando você cria um usuário em seu grupo de usuários e o convida a acessar. O Amazon Cognito envia informações iniciais de acesso para o endereço de e-mail ou o número de telefone do usuário.
> Uma Verification message (Mensagem de verificação) é enviada quando um usuário se cadastra em uma conta no grupo de usuários. O Amazon Cognito envia um código para o usuário. Quando o usuário fornece o código ao Amazon Cognito, ele verifica suas informações de contato e confirma sua conta para acesso. Códigos de verificação são válidos por 24 horas.
> Uma MFA message (Mensagem de MFA) é enviada quando você habilita o SMS de MFA em seu grupo de usuários e um usuário que tenha configurado o SMS de MFA faz o acesso e a MFA é solicitada.

1. Acesse o console do Amazon Cognito e escolha User Pools (Grupos de usuários).
2. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
3. Escolha o menu Modelos de mensagens e selecione Mensagem de verificação, Mensagem de convite ou Mensagem de MFA e clique em Editar.
4. Personalize as mensagens para o tipo de mensagem escolhido.

   > **Note**
   >
   > Todas as variáveis nos modelos de mensagens devem ser incluídas quando você personaliza a mensagem. Se a variável, por exemplo {####}, não for incluída, seu usuário não terá informações suficientes para concluir a ação da mensagem.
   > Para mais informações, consulte Modelos de mensagens.

5. a. Mensagens de verificação

i. Selecione um Verification type (Tipo de verificação) para mensagens de Email (E-mail). Um verificação por Code (Código) envia um código numérico que o usuário deve inserir. Uma verificação por Link (Link) envia um link no qual o usuário pode clicar para verificar suas informações de contato. O texto na variável para uma mensagem de Link é exibido como texto com hiperlink. Por exemplo, um modelo de mensagem usando a variável {##Clique aqui##} é exibido como Clique aqui na mensagem de e-mail.

ii. Insira um Email subject (Assunto do e-mail) para mensagens de Email (E-mail).

iii. Insira um modelo personalizado de Email message (Mensagem de e-mail) para mensagens de Email (E-mail). Você pode personalizar esse modelo usando código em HTML.

iv. Insira um modelo personalizado de SMS message (Mensagem SMS) para mensagens de SMS (SMS).

v. Escolha Salvar alterações.

b. Mensagens de convite

i. Insira um Email subject (Assunto do e-mail) para mensagens de Email (E-mail).

ii. Insira um modelo personalizado de Email message (Mensagem de e-mail) para mensagens de Email (E-mail). Você pode personalizar esse modelo usando código em HTML.

iii. Insira um modelo personalizado de SMS message (Mensagem SMS) para mensagens de SMS (SMS).

iv. Escolha Salvar alterações.

c. Mensagens de MFA

i. Insira um modelo personalizado de SMS message (Mensagem SMS) para mensagens de SMS (SMS).

ii. Escolha Salvar alterações.

Criar um usuário

Criar um usuário

Você pode criar novos usuários para seu grupo de usuários diretamente do console do Amazon Cognito. Normalmente, os usuários podem fazer login depois que eles definem uma senha. Para acesso com um endereço de e-mail, o usuário precisa verificar o atributo `email`. Para fazer login

com um número de telefone, o usuário deve verificar o atributo `phone_number`. Para confirmar contas como administrador, você também pode usar a API AWS CLI ou ou criar perfis de usuário com um provedor de identidade federado. Para mais informações, consulte a Referência de API do Amazon Cognito.

1. Acesse o console do Amazon Cognito e escolha User Pools (Grupos de usuários).
2. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
3. Clique no menu Usuários e, em seguida, selecione Criar um usuário.
4. Revise os User pool sign-in and security requirements (Requisitos de acesso e segurança do grupo de usuários) para obter orientações sobre requisitos de senha, métodos de recuperação de conta disponíveis e atributos de alias para seu grupo de usuários.
5. Escolha como você deseja enviar uma Invitation message (Mensagem de convite). Escolha mensagem SMS, mensagem de e-mail ou ambos. Para suprimir a mensagem de convite, clique em Não enviar um convite.

   > **Note**
   >
   > Para que você possa enviar mensagens de convite, configure um remetente e uma Região da AWS com o Amazon Simple Notification Service e o Amazon Simple Email Service no menu Métodos de autenticação do grupo de usuários. Aplicam-se tarifas de mensagens e dados do destinatário da mensagem. A cobrança de mensagens de e-mail pelo Amazon SES e de mensagens SMS pelo Amazon SNS é feita separadamente.

6. Selecione um Username (Nome de usuário) para o novo usuário.
7. Escolha Create a password (Criar uma senha) ou Generate a password (Gerar uma senha) se desejar que o Amazon Cognito gere uma senha para o usuário. A opção de gerar uma senha não estará disponível se o login sem senha estiver habilitado no grupo de usuários. Qualquer senha temporária deve aderir à política de senha do grupo de usuários.
8. Escolha Criar.
9. Clique no menu Usuários e selecione o Nome de usuário correspondente. Adicione e edite User attributes (Atributos do usuário) e Group memberships (Associações de grupo). Examine User event history (Histórico de eventos do usuário).

Como adicionar grupos a um grupo de usuários

O suporte a grupos de usuários no Amazon Cognito permite que você crie e gerencie grupos, adicione usuários a grupos e remova usuários de grupos. Use grupos a fim de criar coleções de usuários para gerenciar suas permissões ou representar diferentes tipos de usuários. Você pode atribuir uma função AWS Identity and Access Management (IAM) a um grupo para definir as permissões dos membros de um grupo.

Você pode usar grupos para criar um conjunto de usuários em um grupo de usuários, que normalmente é feito para definir as permissões para esses usuários. Por exemplo, você pode criar grupos separados para os usuários que são leitores, colaboradores e editores do seu site e aplicativo. Usando a função do IAM associada a um grupo, você também pode definir permissões diferentes para os diferentes grupos de modo que apenas colaboradores possam colocar conteúdo no Amazon S3 e apenas editores possam publicar conteúdo por meio de uma API no Amazon API Gateway.

O Amazon Cognito cria um grupo de usuários para cada OIDC e provedor de identidade social (IdP) que você adiciona ao seu grupo de usuários. SAMl O nome do grupo está no formato `[user pool ID]_[IdP name]`, por exemplo, `us-east-1_EXAMPLE_MYSSO` ou `us-east-1_EXAMPLE_Google`. Cada perfil de usuário exclusivo do IdP gerado automaticamente é adicionado automaticamente a esse grupo. Os usuários vinculados não são adicionados automaticamente a esse grupo, mas você pode adicionar os perfis ao grupo em outro processo.

Você pode criar e gerenciar grupos em um grupo de usuários a partir da Console de gerenciamento da AWS APIs, da e da CLI. Como desenvolvedor (usando AWS credenciais), você pode criar, ler, atualizar, excluir e listar os grupos de um grupo de usuários. Você também pode adicionar e remover usuários dos grupos.

Não há custo adicional para usar grupos dentro de um grupo de usuários. Para obter mais informações, consulte Preço do Amazon Cognito.

Como atribuir funções do IAM a grupos

É possível usar grupos para controlar permissões aos recursos usando uma função do IAM. As funções do IAM incluem políticas de confiança e políticas de permissão. A política de confiança da função especifica quem pode usar a função. As políticas de permissão especificam as ações e os recursos que os membros do grupo podem acessar. Ao criar uma função do IAM, configure a política de confiança da função a fim de permitir que os usuários do grupo assumam a função. Nas políticas de permissão da função, especifique as permissões que você deseja que o grupo tenha.

Ao criar um grupo no Amazon Cognito, especifique uma função do IAM fornecendo o ARNda função. Quando membros do grupo fazem login usando o Amazon Cognito, eles podem receber credenciais temporárias dos grupos de identidades. Suas permissões são determinadas pela função do IAM associada.

Usuários individuais podem estar em vários grupos. Como desenvolvedor, você tem as seguintes opções para escolher automaticamente a função do IAM quando um usuário estiver em vários grupos:

- Você pode atribuir valores de precedência para cada grupo. O grupo com a melhor (mais baixa) precedência será escolhido e sua função do IAM associada será aplicada.
- Seu aplicativo também pode escolher entre as funções disponíveis ao solicitar AWS credenciais para um usuário por meio de um grupo de identidades, especificando um ARN de função no parâmetro. GetCredentialsForIdentity`CustomRoleARN` A função do IAM especificada deve corresponder a uma função que esteja disponível para o usuário.

Como atribuir valores de precedência a grupos

Um usuário pode pertencer a mais de um grupo. Nos tokens de ID e acesso do usuário, a afirmação `cognito:groups` contém a lista de todos os grupos aos quais um usuário pertence. A requisição `cognito:roles` contém a lista de funções correspondentes aos grupos.

Como um usuário pode pertencer a mais de um grupo, uma precedência pode ser atribuída a cada grupo. Esse é um valor inteiro não negativo que especifica a precedência desse grupo em relação aos outros grupos aos quais um usuário pertence no grupo de usuários. Zero é o principal valor de precedência. Os grupos com os menores valores precedência prevalecem sobre grupos com valores de precedência nulos ou superiores. Se um usuário pertencer a dois ou mais grupos, o grupo com o menor valor de precedência será o que terá a função do IAM aplicada à declaração `cognito:preferred_role` no token de ID do usuário.

Dois grupos podem ter o mesmo valor de precedência. Se isso acontecer, nenhum dos grupos terá precedência sobre o outro. Se dois grupos com o mesmo valor de precedência tiverem a mesma função ARN, essa função será usada na requisição `cognito:preferred_role` em tokens de ID para os usuários em cada grupo. Se os dois grupos tiverem funções diferentes ARNs, a `cognito:preferred_role` reivindicação não será definida nos tokens de ID dos usuários.

Como usar grupos para controlar permissões com o Amazon API Gateway

Você pode usar grupos em um grupo de usuários para controlar permissões com o Amazon API Gateway. Os grupos dos quais um usuário é membro estão incluídos no token de ID e no token de acesso de um grupo de usuários na declaração `cognito:groups`. É possível enviar tokens de ID ou de acesso com solicitações para o Amazon API Gateway e usar um autorizador de grupo de usuários do Amazon Cognito para uma API REST. Para mais informações, consulte Controlar o acesso a uma API REST usando um grupo de usuários do Amazon Cognito como autorizador no Guia do desenvolvedor do API Gateway.

Também é possível autorizar o acesso a uma API HTTP do Amazon API Gateway com um autorizador JWT personalizado. Para obter mais informações, consulte Controle do acesso ao HTTP APIs com autorizadores JWT no Guia do desenvolvedor do API Gateway.

Limitações nos grupos

Grupos de usuários estão sujeitas às seguintes limitações:

- O número de grupos que você pode criar é limitado pelas cotas de serviço do Amazon Cognito.
- Grupos não podem ser aninhados.
- Você não pode pesquisar usuários em um grupo.
- Você não pode pesquisar grupos por nome, mas pode listá-los.

Criando um novo grupo no Console de gerenciamento da AWS

Siga o procedimento abaixo para criar um novo grupo.

Para criar um novo grupo

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Grupos e selecione Criar um grupo.
5. Na página Create a group (Criar um grupo), em Group name (Nome do grupo), insira um nome para o novo grupo.
6. Opcionalmente, é possível fornecer informações adicionais sobre esse grupo usando qualquer um dos seguintes campos:

- Description (Descrição): insira detalhes sobre o uso planejado para esse novo grupo.
- Precedence (Precedência): o Amazon Cognito avalia e aplica todas as permissões de grupo para um determinado usuário com base nos grupos aos quais eles pertencem que têm um valor de precedência menor. O grupo com a precedência mais baixa será escolhido e sua função do IAM associada será aplicada. Para obter mais informações, consulte Como atribuir valores de precedência a grupos.
- IAM role (Função do IAM): é possível atribuir uma função do IAM ao grupo quando precisar controlar permissões aos recursos. Se você estiver integrando um grupo de usuários a um grupo de identidades, a configuração de IAM role (Função do IAM) determinará qual função estará atribuída no token de ID do usuário se o grupo de identidades estiver configurado para selecionar a função a partir do token. Para obter mais informações, consulte Como atribuir funções do IAM a grupos.
- Add users to this group (Adicionar usuários a esse grupo): adicione usuários existentes como membros desse grupo após sua criação.

7. Selecione Create (Criar) para confirmar.

Como gerenciar e pesquisar contas de usuários

Os grupos de usuários podem conter milhões de usuários. Trabalhar com um conjunto de dados desse porte é um desafio para os administradores. O Amazon Cognito tem ferramentas para encontrar e modificar perfis de usuário. Os principais métodos para encontrar usuários são o menu Usuários do console do Amazon Cognito e com. ListUsers Dos métodos que recuperam informações sobre usuários, essas são as opções que não têm um impacto nos custos, ao contrário de, por exemplo, AdminGetUser.

Esta seção do guia contém informações sobre como encontrar e atualizar perfis de usuário em um grupo de usuários.

Como visualizar atributos do usuário

Siga o procedimento abaixo para visualizar atributos do usuário no console do Amazon Cognito.

Para visualizar atributos do usuário

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).

3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Usuários e escolha um usuário na lista.
5. Na página de detalhes do usuário, em User attributes (Atributos do usuário), você pode ver quais atributos estão associados ao usuário.

Como redefinir uma senha do usuário

Siga o procedimento abaixo para redefinir uma senha do usuário no console do Amazon Cognito.

Para redefinir uma senha do usuário

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Usuários e escolha um usuário na lista.
5. Na página de detalhes do usuário, escolha Actions (Ações), Reset password (Redefinir senha).
6. Na caixa de diálogo Reset password (Redefinir senha), leia as informações e, quando estiver pronto, escolha Reset (Redefinir).

   Essa ação resulta imediatamente no envio de um código de confirmação para o usuário e desabilita a senha atual do usuário, ao alterar o estado do usuário para `RESET_REQUIRED`. O código Reset password (Redefinir senha) é válido por 1 hora.

Habilitar, desabilitar e excluir contas de usuário

Você pode excluir perfis de usuário não utilizados ou, se quiser impedir temporariamente o acesso, desabilitá-los. Os usuários podem excluir as próprias contas, mas somente os administradores do grupo de usuários podem habilitar e desabilitar contas de usuário.

Efeito da exclusão

Os usuários não podem fazer login com contas de usuário excluídas e, para recuperar o acesso, devem se cadastrar ou criar uma nova conta.

Efeito da desabilitação de contas

Quando você desabilita uma conta de usuário, o Amazon Cognito invalida automaticamente todas as sessões autenticadas, desativa a conta do usuário para login e revoga os tokens de acesso

e atualização. O Amazon Cognito retorna um erro `invalid_request` com a mensagem `User is not enabled` quando um usuário tenta fazer login em uma conta que você desabilitou. Esse comportamento não muda com as configurações de divulgação de existência do usuário para o cliente de aplicação. É possível desabilitar as contas de usuário locais e os perfis locais das contas de usuário federado. Quando os usuários fazem login com o login gerenciado ou com a IU hospedada clássica, você desabilita a conta deles e eles tentam fazer login novamente com o cookie do navegador que mantém a sessão autenticada, o Amazon Cognito os redireciona para a página de login.

Efeito da habilitação de contas

Os usuários podem fazer login imediatamente nas contas após você habilitá-las. As contas de usuário são habilitadas por padrão. Os atributos e senhas dos usuários permanecem os mesmos de antes da desabilitação da conta. Os tokens que sua aplicação revogou, independentemente de você ter desabilitado a conta do usuário ou revogado separadamente o token de atualização, permanecem inválidos depois que você habilita a conta de usuário que tinha o token.

Delete a user account (console)

Como excluir uma conta de usuário

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Usuários e marque o botão de seleção ao lado do nome de usuário de um usuário na lista.
5. Escolha Excluir.
6. Selecione Desabilitar acesso do usuário.
7. Escolha Excluir.

Delete a user account (API)

Os usuários podem excluir suas contas com a operação da access-token-authorized DeleteUserAPI de autoatendimento. Veja a seguir um exemplo de corpo da solicitação `DeleteUser`.

```
{
```

```
   "AccessToken": "eyJra456defEXAMPLE"
}
```

Os administradores podem excluir contas de usuário com a operação de API autorizada pelo IAM AdminDeleteUser. Veja a seguir um exemplo de corpo da solicitação `AdminDeleteUser`.

```
{
   "Username": "testuser",
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Disable a user account (console)

Como desabilitar uma conta de usuário

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Usuários e escolha o nome de usuário de um usuário na lista.
5. Na página de detalhes do usuário, selecione Ações e Desabilitar acesso do usuário.
6. Na caixa de diálogo criada, clique em Desabilitar.

Disable a user account (API)

Os administradores podem desativar contas de usuário com a operação de API autorizada pelo IAM AdminDisableUser. Veja a seguir um exemplo de corpo da solicitação `AdminDisableUser`.

```
{
   "Username": "testuser",
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Enable a user account (console)

Como habilitar uma conta de usuário

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).

3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Usuários e escolha o nome de usuário de um usuário na lista.
5. Na página de detalhes do usuário, selecione Ações e Habilitar acesso do usuário.
6. Na caixa de diálogo criada, clique em Habilitar.

Enable a user account (API)

Os administradores podem habilitar contas de usuário com a operação de API autorizada pelo IAM AdminEnableUser. Veja a seguir um exemplo de corpo da solicitação `AdminEnableUser`.

```
{
   "Username": "testuser",
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Como pesquisar atributos de usuários

Se você já tiver criado um grupo de usuários, poderá pesquisar no painel Users (Usuários) no Console de gerenciamento da AWS. Você também pode usar a ListUsers API do Amazon Cognito, que aceita um parâmetro Filter.

Você pode pesquisar qualquer um dos seguintes atributos padrão: Atributos personalizados não podem ser pesquisados.

- username (diferencia maiúsculas de minúsculas)
- e-mail
- phone_number
- name
- given_name
- family_name
- preferred_username
- cognito: user_status (chamado Status no console) (diferencia maiúsculas de minúsculas)
- status (chamado Enabled (Habilitado) no console) (diferencia maiúsculas de minúsculas)
- sub

> **Note**
>
> Você também pode listar usuários usando um filtro no lado do cliente. O filtro no lado do servidor não encontra correspondência com mais de um atributo. Para pesquisa avançada, use um filtro no lado do cliente com o parâmetro `--query` da ação `list-users` na AWS Command Line Interface. Quando você usa um filtro do lado do cliente, ListUsers retorna uma lista paginada de zero ou mais usuários. Você pode receber várias páginas consecutivas com zero resultados. Repita a consulta com cada token de paginação retornado até que você receba um valor de token de paginação nulo, em seguida, revise o resultado combinado.
> Para obter mais informações sobre filtragem do lado do servidor e do lado do cliente, consulte AWS CLI Filtragem de saída no Guia do usuário. AWS Command Line Interface

Pesquisando usuários com o Console de gerenciamento da AWS

Se você já tiver criado um grupo de usuários, poderá pesquisar no painel Users (Usuários) no Console de gerenciamento da AWS.

Console de gerenciamento da AWS as pesquisas são sempre pesquisas com prefixo (“começa com”).

Para pesquisar um usuário no console do Amazon Cognito

1. Acesse o console do Amazon Cognito. Você pode ser solicitado a fornecer suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Usuários e digite o nome de usuário no campo de pesquisa. Observe que alguns valores de atributo diferenciam maiúsculas de minúsculas (por exemplo, Username).

   Você também pode encontrar usuários ajustando o filtro de pesquisa para restringir o escopo para outras propriedades do usuário, como Email (E-mail), Phone number (Número de telefone) ou Last name (Sobrenome).

Pesquisar usuários usando o API `ListUsers`

Para pesquisar usuários do seu aplicativo, use a API do Amazon CognitoListUsers . Esta API usa os seguintes parâmetros:

- `AttributesToGet`: uma matriz de strings, onde cada string é o nome de um atributo de usuário a ser retornados para cada usuário nos resultados da pesquisa. Para recuperar todos os atributos, não inclua o parâmetro `AttributesToGet` nem a solicitação `AttributesToGet` com um valor da string literal `null`.
- `Filter`: uma string de filtro do formulário "`AttributeName Filter-Type "AttributeValue`"". Aspas dentro da string de filtro devem ser evitadas usando o caractere de barra invertida (\). Por exemplo, ."`family_name = \"Reddy\"`" Se a string de filtro estiver vazia, `ListUsers` retorna todos os usuários no grupo de usuários.
  - `AttributeName`: o nome do atributo a ser pesquisado. Você só pode pesquisar um atributo por vez.

    > **Note**
    >
    > Você só pode pesquisar atributos padrão. Atributos personalizados não podem ser pesquisados. Isso é porque somente atributos indexados são pesquisáveis, e atributos personalizados não podem ser indexados.

  - `Filter-Type`: para obter uma correspondência exata, use =, por exemplo, `given_name = "Jon"`. Para uma correspondência de prefixo ("começa com"), use ^=, por exemplo, `given_name ^= "Jon"`.
  - `AttributeValue`: o valor de atributo que deve ser correspondido por cada usuário.
- `Limit`: o número máximo de usuários a serem retornados.
- `PaginationToken`: um token para obter mais resultados de uma pesquisa anterior. O Amazon Cognito encerra a validade do token de paginação após uma hora.
- `UserPoolId`: a ID de grupo de usuários para o grupo de usuários na qual a pesquisa deve ser realizada.

Todas as pesquisas diferenciam maiúsculas de minúsculas. Os resultados da pesquisa são classificados pelo atributo nomeado pela string `AttributeName`, em ordem ascendente.

Exemplos de uso da API **`ListUsers`**

O exemplo a seguir retorna todos os usuários e inclui todos os atributos.

```
{
    "AttributesToGet": null,
    "Filter": "",
    "Limit": 10,
    "UserPoolId": "us-east-1_samplepool"
}
```

O exemplo a seguir retorna todos os usuários cujos números de telefone começam com "+1312" e inclui todos os atributos.

```
{
    "AttributesToGet": null,
    "Filter": "phone_number ^= \"+1312\"",
    "Limit": 10,
    "UserPoolId": "us-east-1_samplepool"
}
```

O exemplo a seguir retorna os primeiros 10 usuários que têm "Reddy" como sobrenome. Para cada usuário, os resultados da pesquisa incluem nome do usuário, número de telefone e endereço de e-mail. Se houver mais de 10 usuários correspondentes no grupo de usuários, a resposta incluirá um token de paginação.

```
{
    "AttributesToGet": [
        "given_name",
        "phone_number",
        "email"
    ],
    "Filter": "family_name = \"Reddy\"",
    "Limit": 10,
    "UserPoolId": "us-east-1_samplepool"
}
```

Se o exemplo anterior retornar um token de paginação, o exemplo a seguir retornará os próximos 10 usuários que correspondam à mesma string de filtro.

```
{
    "AttributesToGet": [
        "given_name",
        "phone_number",
        "email"
    ],
    "Filter": "family_name = \"Reddy\"",
    "Limit": 10,
    "PaginationToken": "pagination_token_from_previous_search",
    "UserPoolId": "us-east-1_samplepool"
}
```

Senhas, recuperação de contas e políticas de senha

Todos os usuários que fazem login em um grupo de usuários, até mesmo usuários federados, têm senhas atribuídas aos seus perfis de usuário. Usuários locais e usuários vinculados devem inserir uma senha ao fazerem login. Os usuários federados não usam senhas de grupos de usuários, mas fazem login usando o provedor de identidades (IdP). Você pode permitir que os usuários redefinam suas próprias senhas, redefinam ou alterem senhas como administrador e definam políticas para complexidade e histórico de senhas.

O Amazon Cognito não armazena senhas de usuários em texto simples. Em vez disso, ele armazena um hash da senha de cada usuário com um salt específico do usuário. Por esse motivo, você não pode recuperar senhas existentes dos perfis de usuário em seus grupos de usuários. Como prática recomendada, não armazene senhas de usuário em texto simples em nenhum lugar. Redefina as senhas quando os usuários as esquecerem.

Redefinição e recuperação de senha

Os usuários esquecem suas senhas. Você pode permitir que eles mesmos as redefinam, ou exigir que um administrador redefina a senha para eles. Os grupos de usuários do Amazon Cognito têm opções para os dois modelos. Esta parte do guia aborda as configurações do grupo de usuários e as operações de API para redefinição de senha.

A operação ForgotPasswordda API e a opção de login gerenciado Esqueceu sua senha? enviam aos usuários um código que, quando eles confirmam que têm o código correto, lhes dá a oportunidade

de definir uma nova senha com ConfirmForgotPassword. Esse é o modelo de recuperação de senha por autoatendimento.

Recuperação de usuários não verificados

É possível enviar mensagens de recuperação para usuários que verificaram seu endereço de e-mail ou número de telefone. Se eles não tiverem um e-mail ou telefone de recuperação confirmado, um administrador do grupo de usuários poderá marcar o endereço de e-mail ou número de telefone como verificado. Edite os Atributos de usuário do usuário no console do Amazon Cognito e marque a caixa de seleção ao lado de Marcar número de telefone como verificado ou Marcar endereço de e-mail como verificado. Você também pode `phone_number_verified` definir `email_verified` ou como verdadeiro em uma AdminUpdateUserAttributessolicitação. Para novos usuários, a operação da ResendConfirmationCodeAPI envia um novo código para seu endereço de e-mail ou número de telefone e eles podem concluir a confirmação e a verificação por autoatendimento.

Redefinir senhas como administrador

As operações AdminSetUserPassworde a AdminResetUserPasswordAPI são os métodos de redefinição de senha iniciados pelo administrador. `AdminSetUserPassword`define uma senha temporária ou permanente e `AdminResetUserPassword` envia aos usuários um código de redefinição de senha da mesma forma que. `ForgotPassword`

Configurar redefinição e recuperação de senha

O Amazon Cognito seleciona automaticamente suas opções de recuperação de conta com base nos atributos e opções de login obrigatórios escolhidos ao criar um grupo de usuários no console. Você pode modificar essas configurações padrão.

O método de MFA preferido de um usuário influencia os métodos que ele pode usar para recuperar a senha. Os usuários cujo MFA preferencial é por mensagem de e-mail não podem receber um código de redefinição de senha por e-mail. Os usuários cujo MFA preferencial é por mensagem SMS não podem receber um código de redefinição de senha por SMS.

Suas configurações de recuperação de senha devem fornecer uma opção alternativa quando os usuários não estão qualificados para usar o método de redefinição de senha de sua preferência. Por exemplo, seus mecanismos de recuperação podem ter o e-mail como prioridade e o MFA do e-mail pode ser opcional no seu grupo de usuários. Nesse caso, adicione a recuperação da conta de mensagens SMS como uma segunda opção ou use operações administrativas da API para redefinir as senhas desses usuários.

O Amazon Cognito responde às solicitações de redefinição de senha de usuários que não têm um método de recuperação válido com uma resposta de erro `InvalidParameterException`.

> **Note**
>
> Os usuários não podem receber códigos de redefinição de senha e de MFA no mesmo endereço de e-mail ou número de telefone. Se eles usarem senhas de uso único (OTPs) de mensagens de e-mail para MFA, deverão usar mensagens SMS para recuperação da conta. Se OTPs usarem mensagens SMS para MFA, deverão usar mensagens de e-mail para recuperação da conta. Em grupos de usuários com MFA, talvez os usuários não consigam concluir a recuperação de senha por autoatendimento se tiverem o endereço de e-mail cadastrado, mas não tiverem o número de telefone, ou vice-versa.
>
> Para evitar que os usuários não consigam redefinir as senhas em grupos de usuários com essa configuração, defina os atributos `email` e `phone_number` como obrigatórios. Como alternativa, é possível configurar processos que sempre coletam e definem esses atributos quando os usuários se cadastram ou quando seus administradores criam perfis de usuário. Quando os usuários têm ambos os atributos, o Amazon Cognito envia automaticamente códigos de redefinição de senha para o destino que não é o fator de MFA do usuário.

O procedimento a seguir configura a recuperação de contas de autoatendimento em um grupo de usuários.

Configure self-service password reset (API/SDK)

O `AccountRecoverySetting` parâmetro é o parâmetro do grupo de usuários que define os métodos que os usuários podem usar para recuperar sua senha em solicitações de ForgotPasswordAPI ou quando selecionam Esqueceu a senha? no login gerenciado. `ForgotPassword`envia um código de recuperação para um e-mail verificado ou um número de telefone verificado. O código de recuperação é válido por uma hora. Quando você especifica uma AccountRecoverySetting para o grupo de usuários, o Amazon Cognito escolhe o destino de entrega de código com base na prioridade definida por você.

Quando você define `AccountRecoverySetting` e um usuário tem o MFA SMS configurado, o SMS não pode ser usado como um mecanismo de recuperação de conta. A prioridade dessa configuração é determinada com 1 sendo da prioridade mais alta. O Amazon Cognito envia uma verificação para apenas um dos métodos especificados. O exemplo de `AccountRecoverySetting` a seguir define endereços de e-mail como o destino principal

dos códigos de recuperação de conta, recorrendo à mensagem SMS se o usuário não tiver um endereço de e-mail cadastrado.

```
"AccountRecoverySetting": {
   "RecoveryMechanisms": [
      {
         "Name": "verified_email",
         "Priority": 1
      },
      {
         "Name": "verified_phone_number",
         "Priority": 2
      }
   ]
}
```

O valor `admin_only` desativa a recuperação de contas de autoatendimento, exigindo que os usuários entrem em contato com o administrador para redefinir a senha. Você não pode usar `admin_only` com nenhum outro mecanismo de recuperação de conta. O seguinte exemplo…

```
"AccountRecoverySetting": {
   "RecoveryMechanisms": [
      {
         "Name": "admin_only",
         "Priority": 1
      }
   ]
}
```

Se você não especificar `AccountRecoverySetting`, o Amazon Cognito enviará o código de recuperação primeiro para um número de telefone verificado e, se o usuário não tiver um número de telefone cadastrado, para um endereço de e-mail verificado.

Para obter mais informações sobre a `AccountRecoverySetting`, consulte CreateUserPool e UpdateUserPool.

Configure self-service password reset (console)

Configure as opções de recuperação de conta e redefinição de senha no menu Fazer login do seu grupo de usuários.

Como configurar a recuperação da conta de usuário

1. Faça login no console do Amazon Cognito.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Fazer login. Localize Recuperação de contas de usuários e clique em Editar
5. Para permitir que os usuários redefinam suas próprias senhas, selecione Habilitar recuperação de contas de autoatendimento.
6. Configure o método de entrega dos códigos de recuperação de senha que seu grupo de usuários envia aos usuários. Em Método de entrega para mensagens de recuperação de conta de usuário, selecione uma opção disponível. Como prática recomendada, escolha uma opção que tenha um método secundário para enviar mensagens, por exemplo, E-mail se disponível, ou SMS. Com um método de entrega secundário, o Amazon Cognito pode enviar códigos aos usuários de forma que eles precisem usar um meio diferente para redefinição de senha e para MFA.
7. Selecione Salvar alterações.

Comportamento para esquecimento da senha

Em uma determinada hora, permitimos entre 5 e 20 tentativas para que um usuário solicite ou insira um código de redefinição de senha como parte das ações e ações de esquecimento da senha. confirm-forgot-password O valor exato depende dos parâmetros de risco associados às solicitações. Observe que esse comportamento está sujeito a alterações.

Como adicionar requisitos de senha do grupo de usuários

Senhas fortes e complexas são a prática recomendada de segurança para seu grupo de usuários. Especialmente em aplicações abertas à Internet, senhas fracas podem expor as credenciais dos usuários a sistemas que adivinham senhas e tentam acessar seus dados. Quanto mais complexa for uma senha, mais difícil será adivinhá-la. O Amazon Cognito tem ferramentas adicionais para administradores preocupados com a segurança, como proteção contra ameaças e AWS WAF web ACLs, mas sua política de senha é um elemento central da segurança do seu diretório de usuários.

As senhas para usuários locais nos grupos de usuários do Amazon Cognito não expiram automaticamente. Como prática recomendada, registre a hora, a data e os metadados das redefinições de senha do usuário em um sistema externo. Com um log externo de tempo da senha,

sua aplicação ou um acionador do Lambda pode pesquisar o tempo da senha de um usuário e exigir uma redefinição após determinado período.

Você pode configurar o grupo de usuários para exigir uma complexidade mínima de senha que esteja de acordo com seus padrões de segurança. Senhas complexas têm no mínimo oito caracteres. Eles também incluem uma combinação de caracteres maiúsculos, numéricos e especiais.

Com os níveis de recursos Essentials ou Plus, é possível definir uma política de reutilização de senha. Você pode impedir que um usuário redefina sua senha para uma nova senha que corresponda à senha atual ou a qualquer uma das 23 senhas anteriores adicionais, totalizando no máximo 24.

Como definir uma política do grupo de usuários

1. Crie um grupo de usuários e navegue até a etapa Configurar requisitos de segurança ou acesse um grupo de usuários existente e navegue até o menu Métodos de autenticação.
2. Navegue até Política de senha.
3. Escolha um Modo de política de senha. Os Padrões do Cognito configuram o grupo de usuários com as configurações mínimas recomendadas. Também é possível escolher uma política de senha Personalizada.
4. Defina um Tamanho mínimo de senha. Todos os usuários devem se cadastrar ou serem criados com uma senha com tamanho maior ou igual a esse valor. É possível definir esse valor mínimo de até 99, mas os usuários podem definir senhas com até 256 caracteres.
5. Configure as regras de complexidade de senhas em Requisitos de senha. Escolha os tipos de caracteres: números, caracteres especiais, letras maiúsculas e minúsculas, dos quais você deseja exigir pelo menos um na senha de cada usuário.

   Você pode exigir que as senhas contenham pelo menos um dos seguintes caracteres: Depois que o Amazon Cognito verificar se as senhas contêm os caracteres mínimos necessários, as senhas de seus usuários podem conter caracteres adicionais de qualquer tipo até atingir o tamanho máximo.

   - Letras maiúsculas e minúsculas do latim básico
   - Números
   - Os caracteres especiais a seguir.

     ```
     ^ $ * . [ ] { } ( ) ? " ! @ # % & / \ , > < ' : ; | _ ~ ` = + -
     ```

- Caracteres de espaço não iniciais e não finais.

6. Defina um valor para Senhas temporárias definidas por administradores expiram em. Após esse período, um novo usuário criado no console do Amazon Cognito com `AdminCreateUser` não poderá fazer login e definir uma nova senha. Depois de fazerem login com a senha temporária, as contas de usuário nunca expirarão. Para atualizar a duração da senha na API de grupos de usuários do Amazon Cognito, defina um valor para TemporaryPasswordValidityDays a sua solicitação CreateUserPoolou para a UpdateUserPoolAPI.
7. Defina um valor para Impedir o uso de senhas anteriores, se disponível. Para usar esse recurso, selecione o nível de recursos Essentials ou Plus em seu grupo de usuários. O valor desse parâmetro é o número de senhas anteriores que uma nova senha é impedida de corresponder quando o usuário redefine a senha.

Para redefinir o acesso de uma conta de usuário expirada, siga um destes procedimentos:

- Envie uma nova senha temporária e redefina o período de expiração com uma solicitação de AdminCreateUserAPI `MessageAction` definida como`RESEND`.
- Exclua o perfil de usuário e crie outro.
- Gere um novo código de confirmação em uma solicitação de AdminResetUserPasswordAPI.

Como importar usuários para um grupo de usuários

Existem duas maneiras de importar ou migrar usuários do seu diretório de usuários ou de um banco de dados de usuários existente para grupos de usuários do Amazon Cognito. Você pode migrar os usuários quando eles fizerem login usando o Amazon Cognito pela primeira vez com um acionador do Lambda de migração de usuários. Com essa abordagem, os usuários podem continuar usando suas senhas existentes e não terão que redefini-las após a migração para o grupo de usuários. Como alternativa, você pode migrar usuários em lote carregando um arquivo CSV que contenha os atributos de perfil do usuário para todos os usuários. As seções a seguir descrevem essas duas abordagens.

Mais atributos

- Abordagens para migrar usuários para grupos de usuários do Amazon Cognito
- AWS re:inforce 2023 — Migração para o Amazon Cognito

Tópicos

- Como importar usuários com um acionador do Lambda de migração de usuários
- Como importar usuários para grupos de usuários com base em um arquivo CSV

Como importar usuários com um acionador do Lambda de migração de usuários

Com essa abordagem, você pode migrar perfeitamente os usuários do diretório existente para grupos de usuários quando um usuário fizer login pela primeira vez com sua aplicação ou solicitar uma redefinição de senha. Adicione uma função Migrar o acionador do Lambda do usuário ao grupo de usuários para que ele receba metadados sobre os usuários que tentam fazer login e retorne informações de perfil de usuário de uma fonte de identidade externa. Para obter detalhes e um código de exemplo para esse acionador do Lambda, bem como parâmetros de solicitação e resposta, consulte Parâmetros do acionador do Lambda de migrar usuário.

Antes de iniciar a migração de usuários, crie uma função Lambda de migração de usuários em sua Conta da AWS e, em seu grupo de usuários, configure a função do Lambda como acionador de migração de usuários. Adicione uma política de autorização à sua função do Lambda que permita que somente a entidade principal da conta de serviço do Amazon Cognito, `cognito-idp.amazonaws.com`, invoque a função do Lambda, e apenas no contexto de seu próprio grupo de usuários. Para obter mais informações, consulte Uso de políticas baseadas em recursos para o AWS Lambda (políticas de função do Lambda).

Processo de login

1. O usuário abre sua aplicação e faz login com a API de grupos de usuários do Amazon Cognito ou por meio do login gerenciado. Para obter mais informações sobre como facilitar o login com o Amazon APIs Cognito, consulte. Integração da autenticação e autorização do Amazon Cognito com aplicações móveis e da web
2. Sua aplicação envia o nome de usuário e a senha ao Amazon Cognito. Se seu aplicativo tiver uma interface de usuário de login personalizada que você criou com um AWS SDK, seu aplicativo deverá usar InitiateAuthou AdminInitiateAuthcom o `USER_PASSWORD_AUTH` fluxo ou. `ADMIN_USER_PASSWORD_AUTH` Quando a aplicação usa um desses fluxos, o SDK envia a senha ao servidor.

> **Note**
>
> Antes de adicionar um acionador de migração de usuários, ative o fluxo `USER_PASSWORD_AUTH` ou `ADMIN_USER_PASSWORD_AUTH` nas configurações do cliente de aplicação. Você deve usar esses fluxos em vez do fluxo `USER_SRP_AUTH` padrão. O

Amazon Cognito deve enviar uma senha à sua função do Lambda para que ele possa verificar a autenticação do usuário no outro diretório. Uma SRP obscurece a senha do usuário de sua função do Lambda.

3. O Amazon Cognito verifica se o nome de usuário enviado corresponde a um nome de usuário ou alias no grupo de usuários. Você pode definir o nome de usuário preferido, o endereço de e-mail ou o número de telefone do usuário como um alias no grupo de usuários. Se o usuário não existir, o Amazon Cognito enviará parâmetros, incluindo o nome de usuário e a senha, à função Migrar o acionador do Lambda do usuário.
4. Sua função Migrar o acionador do Lambda do usuário verifica ou autentica o usuário com seu diretório de usuários existente ou com o banco de dados de usuários. A função retorna atributos do usuário que o Amazon Cognito armazena no perfil do usuário no grupo de usuários. Você pode retornar um parâmetro `username` somente se o nome de usuário enviado corresponder a um atributo de alias. Se você quiser que os usuários continuem usando a senha que eles já têm, sua função definirá o atributo `finalUserStatus` como `CONFIRMED` na resposta do Lambda. Sua aplicação deve retornar todos os parâmetros `"response"` mostrados em Parâmetros do acionador do Lambda de migrar usuário.

   Important

   Não registre todo o objeto de evento de solicitação em seu código Lambda de migração de usuários. Esse objeto de evento de solicitação inclui a senha do usuário. Se você não limpar os registros, as senhas aparecerão nos Registros. CloudWatch
5. O Amazon Cognito cria o perfil de usuário no grupo de usuários e retorna tokens para o aplicativo cliente.
6. Sua aplicação executa a entrada de token, aceita a autenticação do usuário e prossegue para o conteúdo solicitado.

Depois de migrar seus usuários, use `USER_SRP_AUTH` para fazer login. O protocolo Secure Remote Password (SRP) não envia a senha pela rede e oferece benefícios de segurança em relação ao fluxo `USER_PASSWORD_AUTH` usado durante a migração.

Em caso de erros durante a migração, incluindo problemas com o dispositivo cliente ou a rede, a aplicação receberá respostas de erro da API de grupos de usuários do Amazon Cognito. Quando isso acontecer, o Amazon Cognito poderá ou não criar a conta de usuário em seu grupo de usuários.

Em seguida, o usuário deverá tentar entrar novamente. Se o login falhar repetidamente, tente redefinir a senha do usuário com o fluxo de esquecimento de senha em sua aplicação.

O fluxo de esquecimento de senha também chama sua função Migrar o acionador do Lambda do usuário com uma fonte de eventos `UserMigration_ForgotPassword`. Como o usuário não envia uma senha quando solicita uma redefinição de senha, o Amazon Cognito não inclui uma senha no evento que ele envia à sua função do Lambda. Sua função só pode pesquisar o usuário em seu diretório de usuários existente e retornar atributos para adicionar ao perfil do usuário em seu grupo de usuários. Depois que a função conclui a invocação e retorna a resposta ao Amazon Cognito, o grupo de usuários envia um código de redefinição de senha por e-mail ou SMS. Em seu aplicativo, solicite ao usuário o código de confirmação e uma nova senha e, em seguida, envie essas informações para o Amazon Cognito em uma solicitação de ConfirmForgotPasswordAPI. Também é possível usar as páginas integradas para o fluxo de esquecimento da senha no login gerenciado.

Recursos adicionais do

- Abordagens para migrar usuários para grupos de usuários do Amazon Cognito

Como importar usuários para grupos de usuários com base em um arquivo CSV

Quando você tem um repositório de identidade externo e tem tempo para preparar seu grupo de usuários para novos usuários locais, a importação em massa de usuários de um arquivo de valores separados por vírgula (CSV) pode ser uma opção simplificada e econômica para a migração para um grupo de usuários do Amazon Cognito. A importação de um arquivo CSV é um processo de baixar e preencher um arquivo de modelo e, em seguida, entregar o arquivo ao seu grupo de usuários em um trabalho de importação. Você pode usar uma importação de CSV para criar rapidamente usuários de teste. Você também pode preencher programaticamente o arquivo com solicitações da API de leitura para seu repositório de identidade externo e, em seguida, analisar seus detalhes e atributos em operações de gravação no arquivo.

O processo de importação define valores para todos os atributos de usuário, exceto password. Não há suporte para a importação de senha, pois as melhores práticas de segurança exigem que as senhas não estejam disponíveis como texto sem formatação, e não oferecemos suporte à importação de hashes. Isso significa que os usuários devem alterar suas senhas na primeira vez em que fizerem login. Seus usuários estão em estado `RESET_REQUIRED` quando são importados por esse método.

A maneira mais simples de importar usuários de um CSV é ativar o login sem senha no grupo de usuários. Com os atributos de endereço de e-mail e número de telefone e a configuração correta do

grupo de usuários, os usuários podem entrar com senhas de uso único por e-mail ou SMS (OTPs) imediatamente após a conclusão do trabalho de importação. Para obter mais informações, consulte Solicitação de redefinição de senha aos usuários importados.

Também é possível definir as senhas dos usuários com uma solicitação de API AdminSetUserPassword que define o parâmetro `Permanent` como `true`. A importação de CSV não contribui para a cobrança mensal de usuários ativos (MAUs) em seu grupo de usuários. No entanto, as operações de redefinição de senha são geradas. MAUs Para gerenciar os custos ao importar um grande número de usuários com senhas que podem não estar imediatamente ativos, configure a aplicação para solicitar aos usuários uma nova senha quando eles fizerem login e receberem o desafio `RESET_REQUIRED`.

> **Note**
>
> A data de criação de cada usuário é a hora em que o usuário foi importado para o grupo de usuários. A data de criação não é um dos atributos importados.

Etapas para criar um trabalho de importação de usuário

1. Crie uma função do Amazon CloudWatch Logs no console AWS Identity and Access Management (IAM).
2. Crie o arquivo .csv de importação do usuário.
3. Crie e execute o trabalho de importação do usuário.
4. Carregue o arquivo.csv de importação do usuário.
5. Inicie e execute o trabalho de importação do usuário.
6. Use CloudWatch para verificar o registro de eventos.
7. Solicite que os usuários importados redefinam suas senhas.

Mais atributos

- Arquitetura de referência de exportação de perfis de usuário do Cognito para exportar contas de usuário entre grupos de usuários

Tópicos

- Criação da função do CloudWatch Logs IAM

- Criar o arquivo CSV de importação do usuário
- Como criar e executar o trabalho de importação do grupo de usuários do Amazon Cognito
- Visualizando os resultados da importação do grupo de usuários no CloudWatch console
- Solicitação de redefinição de senha aos usuários importados

Criação da função do CloudWatch Logs IAM

Se você estiver usando a CLI ou a API do Amazon Cognito, precisará criar uma função do IAM. CloudWatch O procedimento a seguir descreve como criar uma função do IAM que o Amazon Cognito pode usar para gravar os resultados do seu trabalho de importação no Logs. CloudWatch

> **Note**
>
> Ao criar um trabalho de importação no console do Amazon Cognito, você pode criar o perfil do IAM ao mesmo tempo. Quando você seleciona Create a new IAM role (Criar um perfil do IAM), o Amazon Cognito aplica automaticamente a política de confiança e a política do IAM apropriadas ao perfil.

Para criar a função do CloudWatch Logs IAM para importação de grupos de usuários (AWS CLI, API)

1. Faça login no Console de gerenciamento da AWS e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Crie uma nova função do IAM para um AWS service (Serviço da AWS). Para obter instruções detalhadas, consulte Criar um perfil para um AWS service (Serviço da AWS) no Guia do usuário do AWS Identity and Access Management .
   a. Ao selecionar um Use case (Caso de uso) para o Trusted entity type (Tipo de entidade confiável), escolha qualquer serviço. Atualmente, o Amazon Cognito não está listado nos casos de uso de serviço.
   b. Na tela Add permissions (Adicionar permissões), escolha Create policy (Criar política) e insira a instrução de política a seguir. *REGION*Substitua pelo Região da AWS do seu grupo de usuários, por exemplo`us-east-1`. *ACCOUNT*Substitua pelo seu Conta da AWS ID, por exemplo`111122223333`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-1:111122223333:log-group:/aws/
cognito/*"
            ]
        }
    ]
}
```

3. Como você não escolheu o Amazon Cognito como entidade confiável ao criar o perfil, agora é necessário editar manualmente a relação de confiança do perfil. No painel de navegação do console do IAM, selecione Roles (Perfis) e escolha o perfil criado.
4. Selecione a guia Trust relationships (Relações de confiança).
5. Selecione Edit trust policy (Editar política de confiança).
6. Cole a seguinte instrução de política em Edit trust policy (Editar política de confiança), substituindo qualquer texto existente:

   JSON

```
{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "cognito-idp.amazonaws.com"
                },
```

```
                "Action": "sts:AssumeRole"
            }
        ]
    }
```

7. Escolha Atualizar política.
8. Anote o ARN do perfil do . Forneça o ARN ao criar o trabalho de importação.

Criar o arquivo CSV de importação do usuário

Antes de poder importar os usuários existentes para o grupo de usuários, é necessário criar um arquivo de valores separados por vírgula (CSV) que contenha os usuários que você deseja importar e os atributos deles. No grupo de usuários, é possível recuperar um arquivo de importação de usuários com cabeçalhos que refletem o esquema de atributos do grupo de usuários. Depois, você pode inserir informações do usuário que correspondam aos requisitos de formatação em Formatar o arquivo CSV.

Baixar o cabeçalho do arquivo CSV (console)

Use o procedimento a seguir para baixar o arquivo de cabeçalho CSV.

Como baixar o cabeçalho do arquivo CSV

1. Acesse o console do Amazon Cognito. Você pode ser solicitado a fornecer suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Usuários.
5. Na seção Import users (Importar usuários), selecione Create an import job (Criar um trabalho de importação).
6. Em Upload CSV (Fazer upload do CSV, selecione o link template.csv e baixe o arquivo CSV.

Baixar o cabeçalho do arquivo CSV (AWS CLI)

Para obter uma lista dos cabeçalhos corretos, no menu Usuários, em Importar usuários, selecione Criar trabalho de importação. Na caixa de diálogo a seguir, selecione o link `template.csv` para baixar um arquivo de modelo com os atributos do grupo de usuários.

Você também pode executar o seguinte comando da CLI, onde *USER_POOL_ID* está o identificador do grupo de usuários para o qual você importará usuários:

```
aws cognito-idp get-csv-header --user-pool-id "USER_POOL_ID"
```

Resposta de exemplo:

```
{
    "CSVHeader": [
        "name",
        "given_name",
        "family_name",
        "middle_name",
        "nickname",
        "preferred_username",
        "profile",
        "picture",
        "website",
        "email",
        "email_verified",
        "gender",
        "birthdate",
        "zoneinfo",
        "locale",
        "phone_number",
        "phone_number_verified",
        "address",
        "updated_at",
        "cognito:mfa_enabled",
        "cognito:username"
    ],
    "UserPoolId": "USER_POOL_ID"
}
```

Formatar o arquivo CSV

O arquivo de cabeçalho CSV de importação de usuários baixado se parece com a string a seguir. Ele também inclui os atributos personalizados que você adicionou ao grupo de usuários.

```
cognito:username,name,given_name,family_name,middle_name,nickname,preferred_username,profile,pi
```

Edite o arquivo CSV para que ele inclua esse cabeçalho e os valores de atributo dos usuários e seja formatado de acordo com as seguintes regras:

> **Note**
>
> Para obter mais informações sobre valores de atributo, como o formato apropriado de números de telefone, consulte Trabalhar com atributos do usuário.

- A primeira linha no arquivo é a linha de cabeçalho baixada que contém os nomes de atributo de usuário.
- A ordem das colunas no arquivo CSV não importa.
- Cada linha após a primeira linha contém os valores de atributo para um usuário.
- Todas as colunas do cabeçalho devem estar presente, mas você não precisa fornecer valores em cada coluna.
- Os seguintes atributos são necessários:
  - cognito:username
  - email_verified ou phone_number_verified
    - Pelo menos um dos atributos verificados automaticamente devem ser `true` para todos os usuários. Um atributo verificado automaticamente é um endereço de e-mail ou número de telefone para o qual o Amazon Cognito envia automaticamente um código quando um novo usuário se junta ao grupo de usuários.
    - O grupo de usuários deve ter, pelo menos, um atributo verificado automaticamente, email_verified ou phone_number_verified. Se o grupo de usuários não tiver atributos verificados automaticamente, o trabalho de importação não será iniciado.
    - Se o grupo de usuários tiver apenas um atributo verificado automaticamente, esse atributo deverá ser verificado para todos os usuários. Por exemplo, se o grupo de usuários tiver apenas phone_number como atributo verificado automaticamente, o valor de phone_number_verified deverá ser `true` para todos os usuários.

    > **Note**
    >
    > Para que os usuários redefinam suas senhas, eles deverão ter um e-mail ou número de telefone verificado. O Amazon Cognito envia uma mensagem contendo um código de redefinição de senha para o e-mail ou ao número de telefone especificado no arquivo CSV. Se a mensagem for enviada ao número de telefone, ela será enviada por

mensagem de SMS. Para obter mais informações, consulte Como verificar informações de contato no cadastro.

  - email (se email_verified for `true`)
  - phone_number (se phone_number_verified for `true`)
  - Todos os atributos marcados como necessários quando você criou o grupo de usuários
- Os valores de atributo que são strings não devem ser aspas.
- Se um valor de atributo contiver uma vírgula, você deverá colocar uma barra invertida (\) antes da vírgula. Isso acontece porque os campos em um arquivo CSV são separados por vírgulas.
- O conteúdo do arquivo CSV deve estar no formato UTF-8 sem a marca de ordem de byte.
- O campo cognito:username é obrigatório e deve ser exclusivo no grupo de usuários. Ele pode ser qualquer string Unicode. No entanto, ele não pode conter espaços ou guias.
- Os valores da data de nascimento, se presentes, devem estar no formato *`mm/dd/yyyy`*. Isso significa, por exemplo, que a data de nascimento 1°. de fevereiro de 1985 deve ser codificada como **`02/01/1985`**.
- O campo cognito:mfa_enabled deve corresponder aos requisitos de MFA do seu grupo de usuários. Se você tiver definido a autenticação multifator (MFA) para ser obrigatória no grupo de usuários, esse campo deverá ser `true` ou estar em branco para todos os usuários. Se você tiver definido a MFA para ser desativada, esse campo deverá ser `false` ou ficar em branco para todos os usuários. Um valor em branco define o status habilitado para MFA dos usuários importados para o estado exigido pelo grupo de usuários. Você pode importar usuários em um grupo de usuários exigido pela MFA sem um fator de MFA válido, independentemente de ter definido um valor `cognito:mfa_enabled`. Os usuários nesse estado têm a MFA ativa, mas não podem fazer login até configurarem um atributo de e-mail, um atributo de número de telefone ou uma TOTP, e essa configuração é um fator de MFA válido em seu grupo de usuários.
- O comprimento máximo da linha é de 16.000 caracteres.
- O tamanho máximo do arquivo CSV é 100 MB.
- O número máximo de linhas (usuários) no arquivo é de 500.000. Esse máximo não inclui a linha de cabeçalho.
- Espera-se que o valor do campo updated_at esteja no formato de época em segundos, por exemplo: **`1471453471`**.
- Qualquer espaço em branco à esquerda ou à direita em um valor de atributo será aparado.

A lista a seguir é um exemplo de arquivo de importação CSV para um grupo de usuários sem atributos personalizados. Seu esquema do grupo de usuários pode ser diferente deste exemplo. Nesse caso, você deve fornecer valores de teste no modelo CSV baixado do seu grupo de usuários.

```
cognito:username,name,given_name,family_name,middle_name,nickname,preferred_username,profile,pi
John,,John,Doe,,,,,,,johndoe@example.com,TRUE,,02/01/1985,,,+12345550100,TRUE,123 Any
 Street,,FALSE
Jane,,Jane,Roe,,,,,,,janeroe@example.com,TRUE,,01/01/1985,,,+12345550199,TRUE,100 Main
 Street,,FALSE
```

Como criar e executar o trabalho de importação do grupo de usuários do Amazon Cognito

Esta seção descreve como criar e executar o trabalho de importação do grupo de usuários usando o console do Amazon Cognito e o AWS Command Line Interface ()AWS CLI.

Tópicos

- Importar usuários de um arquivo CSV (console)
- Como importar usuários (AWS CLI)

Importar usuários de um arquivo CSV (console)

O procedimento a seguir descreve como importar os usuários do arquivo CSV.

Como importar usuários do arquivo CSV (console)

1. Acesse o console do Amazon Cognito. Você pode ser solicitado a fornecer suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Usuários.
5. Na seção Import users (Importar usuários), selecione Create an import job (Criar um trabalho de importação).
6. Na página Create import job (Criar um trabalho de importação), insira um Job name (Nome do trabalho).
7. Escolha entre Create a new IAM role (Criar um perfil do IAM) ou Use an existing IAM role (Usar um perfil do IAM existente).

a. Se você optou por Create a new IAM role (Criar um perfil do IAM), insira um nome para o novo perfil. O Amazon Cognito criará automaticamente uma função com as permissões e a relação de confiança corretas. A entidade principal do IAM que cria o trabalho de importação deve ter permissões para criar perfis do IAM.

b. Se você optou por Use an existing IAM role (Usar um perfil do IAM existente), escolha um perfil na lista em IAM role selection (Seleção de perfil do IAM). Esse perfil deve ter as permissões e a política de confiança descritas em Criação da função do CloudWatch Logs IAM.

8. Em Fazer upload do CSV, selecione Escolher arquivo e anexe o arquivo CSV que você preparou.
9. Selecione Create job (Criar trabalho) para enviar seu trabalho, mas iniciá-lo mais tarde. Selecione Create and start job (Criar e iniciar trabalho) para enviar seu trabalho e iniciá-lo imediatamente.
10. Se você criou o trabalho, mas não o iniciou, poderá iniciá-lo mais tarde. No menu Usuários, em Importar usuários, escolha o trabalho de importação e clique em Iniciar. Você também pode enviar uma solicitação de StartUserImportJobAPI a partir de um AWS SDK.
11. Monitore o andamento do trabalho de importação de usuários no menu Usuários em Importar usuários. Se o trabalho não for bem-sucedido, você poderá selecionar o valor do Status. Para obter mais detalhes, selecione Visualizar os CloudWatch registros para obter mais detalhes e analisar quaisquer problemas no console de CloudWatch registros.

Como importar usuários (AWS CLI)

Os comandos da CLI a seguir estão disponíveis para importar usuários para um grupo de usuários:

- `create-user-import-job`
- `get-csv-header`
- `describe-user-import-job`
- `list-user-import-jobs`
- `start-user-import-job`
- `stop-user-import-job`

Para obter a lista de opções de linha de comando desses comandos, use a opção de linha de comando `help`. Por exemplo:

```
aws cognito-idp get-csv-header help
```

Como criar um trabalho de importação de usuário

Depois de criar seu arquivo CSV, crie um trabalho de importação de usuários executando o seguinte comando da CLI, *JOB_NAME* onde está o nome que você está escolhendo para o trabalho*USER_POOL_ID*, o ID do grupo de usuários ao qual os novos usuários serão adicionados *ROLE_ARN* e o ARN da função que você recebeu em: Criação da função do CloudWatch Logs IAM

```
aws cognito-idp create-user-import-job --job-name "JOB_NAME" --user-pool-id
 "USER_POOL_ID" --cloud-watch-logs-role-arn "ROLE_ARN"
```

O *PRE_SIGNED_URL* retorno na resposta é válido por 15 minutos. Após esse tempo, ele expirará e você deverá criar um novo trabalho de importação de usuário para obter um novo URL.

Example Resposta:

```
{
    "UserImportJob": {
        "Status": "Created",
        "SkippedUsers": 0,
        "UserPoolId": "USER_POOL_ID",
        "ImportedUsers": 0,
        "JobName": "JOB_NAME",
        "JobId": "JOB_ID",
        "PreSignedUrl": "PRE_SIGNED_URL",
        "CloudWatchLogsRoleArn": "ROLE_ARN",
        "FailedUsers": 0,
        "CreationDate": 1470957431.965
    }
}
```

Valores de status de um trabalho de importação de usuário

Nas respostas aos comandos de importação de usuário, você verá um dos seguintes valores de `Status`:

- `Created`: o trabalho foi criado, mas não foi iniciado.
- `Pending`: um estado de transição. Você iniciou o trabalho, mas não começou a importação de usuários ainda.

- `InProgress`: o trabalho foi iniciado e os usuários estão sendo importados.
- `Stopping`: você interrompeu o trabalho, mas o trabalho ainda não parou de importar usuários.
- `Stopped`: você interrompeu o trabalho e o trabalho interrompeu a importação de usuários.
- `Succeeded`: o trabalho foi concluído com êxito.
- `Failed`: o trabalho foi interrompido devido a um erro.
- `Expired`: você criou um trabalho, mas não iniciou o trabalho no intervalo de 24 a 48 horas. Todos os dados associados ao trabalho foram excluídos e o trabalho não pode ser iniciado.

Fazer upload do arquivo CSV

Use o comando `curl` a seguir para fazer upload do arquivo CSV que contém os dados do usuário no URL pré-assinado que você obteve da resposta do comando `create-user-import-job`.

```
curl -v -T "PATH_TO_CSV_FILE" -H "x-amz-server-side-encryption:aws:kms"
 "PRE_SIGNED_URL"
```

Na saída deste comando, procure a frase `"We are completely uploaded and fine"`. Essa frase indica que o upload do arquivo foi realizado com êxito. Os grupos de usuários não mantêm as informações nos arquivos de importação depois que você executa os trabalhos de importação. Depois que eles forem concluídos ou expirarem, o Amazon Cognito excluirá seu arquivo CSV carregado.

Como descrever um trabalho de importação de usuário

Para obter uma descrição do seu trabalho de importação de usuários, use o comando a seguir, onde *`USER_POOL_ID`* está o ID do grupo de usuários e *`JOB_ID`* o ID do trabalho que foi retornado quando você criou o trabalho de importação de usuários.

```
aws cognito-idp describe-user-import-job --user-pool-id "USER_POOL_ID" --job-id
 "JOB_ID"
```

Example Resposta de exemplo:

```
{
    "UserImportJob": {
        "Status": "Created",
        "SkippedUsers": 0,
```

```
        "UserPoolId": "USER_POOL_ID",
        "ImportedUsers": 0,
        "JobName": "JOB_NAME",
        "JobId": "JOB_ID",
        "PreSignedUrl": "PRE_SIGNED_URL",
        "CloudWatchLogsRoleArn":"ROLE_ARN",
        "FailedUsers": 0,
        "CreationDate": 1470957431.965
    }
}
```

No exemplo de saída anterior, *PRE_SIGNED_URL* é o URL para o qual você carregou o arquivo CSV. Esse *ROLE_ARN* é o ARN da função de CloudWatch registros que você recebeu ao criar a função.

Como listar os trabalhos de importação de usuário

Para listar os trabalhos de importação de usuário, use o comando a seguir:

```
aws cognito-idp list-user-import-jobs --user-pool-id "USER_POOL_ID" --max-results 2
```

Example Resposta de exemplo:

```
{
    "UserImportJobs": [
        {
            "Status": "Created",
            "SkippedUsers": 0,
            "UserPoolId": "USER_POOL_ID",
            "ImportedUsers": 0,
            "JobName": "JOB_NAME",
            "JobId": "JOB_ID",
            "PreSignedUrl":"PRE_SIGNED_URL",
            "CloudWatchLogsRoleArn":"ROLE_ARN",
            "FailedUsers": 0,
            "CreationDate": 1470957431.965
        },
        {
            "CompletionDate": 1470954227.701,
            "StartDate": 1470954226.086,
            "Status": "Failed",
            "UserPoolId": "USER_POOL_ID",
            "ImportedUsers": 0,
```

```
            "SkippedUsers": 0,
            "JobName": "JOB_NAME",
            "CompletionMessage": "Too many users have failed or been skipped during the
 import.",
            "JobId": "JOB_ID",
            "PreSignedUrl":"PRE_SIGNED_URL",
            "CloudWatchLogsRoleArn":"ROLE_ARN",
            "FailedUsers": 5,
            "CreationDate": 1470953929.313
        }
    ],
    "PaginationToken": "PAGINATION_TOKEN"
}
```

Os trabalhos são listados em ordem cronológica, do último criado ao primeiro. A *PAGINATION_TOKEN* string após o segundo trabalho indica que há resultados adicionais para esse comando de lista. Para listar os resultados adicionais, use a opção `--pagination-token` opção da seguinte forma:

```
aws cognito-idp list-user-import-jobs --user-pool-id "USER_POOL_ID" --max-results 10 --
pagination-token "PAGINATION_TOKEN"
```

Como iniciar um trabalho de importação de usuário

Para iniciar um trabalho de importação de usuário, use o seguinte comando:

```
aws cognito-idp start-user-import-job --user-pool-id "USER_POOL_ID" --job-id "JOB_ID"
```

Somente um trabalho de importação pode ser ativado por vez por conta.

Example Resposta de exemplo:

```
{
    "UserImportJob": {
        "Status": "Pending",
        "StartDate": 1470957851.483,
        "UserPoolId": "USER_POOL_ID",
        "ImportedUsers": 0,
        "SkippedUsers": 0,
        "JobName": "JOB_NAME",
        "JobId": "JOB_ID",
```

```
        "PreSignedUrl":"PRE_SIGNED_URL",
        "CloudWatchLogsRoleArn": "ROLE_ARN",
        "FailedUsers": 0,
        "CreationDate": 1470957431.965
    }
}
```

Como interromper um trabalho de importação de usuário

Para interromper um trabalho de importação de usuário em andamento, use o comando a seguir. Após interromper o trabalho, ele não poderá ser reiniciado.

```
aws cognito-idp stop-user-import-job --user-pool-id "USER_POOL_ID" --job-id "JOB_ID"
```

Example Resposta de exemplo:

```
{
    "UserImportJob": {
        "CompletionDate": 1470958050.571,
        "StartDate": 1470958047.797,
        "Status": "Stopped",
        "UserPoolId": "USER_POOL_ID",
        "ImportedUsers": 0,
        "SkippedUsers": 0,
        "JobName": "JOB_NAME",
        "CompletionMessage": "The Import Job was stopped by the developer.",
        "JobId": "JOB_ID",
        "PreSignedUrl":"PRE_SIGNED_URL",
        "CloudWatchLogsRoleArn": "ROLE_ARN",
        "FailedUsers": 0,
        "CreationDate": 1470957972.387
    }
}
```

Visualizando os resultados da importação do grupo de usuários no CloudWatch console

Você pode ver os resultados do seu trabalho de importação no CloudWatch console da Amazon.

Tópicos

- Como visualizar os resultados
- Como interpretar os resultados

Como visualizar os resultados

As etapas a seguir descrevem como exibir os resultados de importação de grupo de usuários.

Para exibir os resultados da importação de grupos de usuários

1. Faça login no Console de gerenciamento da AWS e abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Escolha Logs.
3. Escolha o grupo de logs dos trabalhos de importação de grupo de usuários. O nome do grupo de logs está no formato `/aws/cognito/userpools/`*`USER_POOL_ID`*`/`*`USER_POOL_NAME`*.
4. Escolha o log do trabalho de importação de usuário que você acabou de executar. O nome do registro está no formato*`JOB_ID`*/*`JOB_NAME`*. Os resultados do registro referem-se aos usuários por número de linha. Nenhum dado de usuário é gravado no log. Para cada usuário, uma linha semelhante à seguinte é exibida:

   - `[SUCCEEDED] Line Number 5956 - The import succeeded.`
   - `[SKIPPED] Line Number 5956 - The user already exists.`
   - `[FAILED] Line Number 5956 - The User Record does not set any of the auto verified attributes to true. (Example: email_verified to true).`

Como interpretar os resultados

Usuários importados com sucesso têm seu status definido como "PasswordReset”.

Nos casos a seguir, o usuário não será importado, mas o trabalho de importação continuará:

- Nenhum atributo verificado automaticamente é definido como `true`.
- Os dados do usuário não correspondem ao esquema.
- Não é possível importar o usuário devido a um erro interno.

Nos casos a seguir, o trabalho de importação apresentará falha:

- A função Amazon CloudWatch Logs não pode ser assumida, não tem a política de acesso correta ou foi excluída.
- O grupo de usuários foi excluído.
- O Amazon Cognito não pode analisar o arquivo .csv.

Solicitação de redefinição de senha aos usuários importados

Se o grupo de usuários oferecer somente login baseado em senha, os usuários deverão redefinir as senhas após a importação. Na primeira vez que fizerem login, poderão inserir qualquer senha. O Amazon Cognito solicita que eles insiram uma nova senha na resposta da API à solicitação de login da sua aplicação.

Se o grupo de usuários tiver fatores de autenticação sem senha, o Amazon Cognito os usará como padrão para usuários importados. Eles não serão solicitados a criar uma nova senha e poderão fazer login imediatamente com uma OTP enviada por e-mail ou SMS, sem senha. Você também poderá solicitar que os usuários definam uma senha para poderem usar outros métodos de login, como nome de usuário e senha ou chave de acesso. As condições a seguir se aplicam ao login sem senha após a importação do usuário.

1. Você deve importar usuários com um atributo que corresponda a um fator de login sem senha disponível. Se os usuários puderem fazer login com um endereço de e-mail, você deverá importar um atributo `email`. Se puderem fazer login com um número de telefone, você deverá importar um atributo `phone_number`. Se puderem fazer login com ambos, importe um valor para qualquer um dos atributos.
2. Normalmente, os usuários são importados em um estado `RESET_REQUIRED` em que precisam redefinir a senha. Se eles forem importados com a capacidade de fazer login com um fator sem senha, o Amazon Cognito definirá seu estado como `CONFIRMED`.

Para saber mais sobre a autenticação sem senha, incluindo como configurá-la e como criar o fluxo de autenticação em sua aplicação, consulte Autenticação com grupos de usuários do Amazon Cognito.

O procedimento a seguir descreve a experiência do usuário em um mecanismo de login personalizado com usuários locais em um `RESET_REQUIRED` após a importação de um arquivo CSV. Se seus usuários fizerem login com login gerenciado, instrua-os a selecionar a opção Esqueceu a senha?, informe o código deles por e-mail ou mensagem de texto e defina uma senha.

Solicitação de redefinição de senha aos usuários importados

1. Na aplicação, tente fazer login silenciosamente para o usuário atual com `InitiateAuth` usando uma senha aleatória.

2. O Amazon Cognito retorna uma `NotAuthorizedException` quando `PreventUserExistenceErrors` está habilitado. Caso contrário, retornará `PasswordResetRequiredException`.
3. A aplicação faz uma solicitação da API `ForgotPassword` e redefine a senha do usuário.
    a. A aplicação envia o nome de usuário em uma solicitação de API `ForgotPassword`.
    b. O Amazon Cognito envia um código para o e-mail ou telefone verificado. O destino depende dos valores que você forneceu para `email_verified` e `phone_number_verified` no arquivo CSV. A resposta à solicitação `ForgotPassword` indica o destino do código.

        > **Note**
        >
        > O grupo de usuários deve estar configurado para verificar e-mails ou números de telefone. Para obter mais informações, consulte Como cadastrar e confirmar contas de usuários.

    c. A aplicação exibe uma mensagem para o usuário verificar o local para onde o código foi enviado e solicita que o usuário insira o código e uma nova senha.
    d. O usuário informa o código e a nova senha no aplicativo.
    e. A aplicação envia o código e a nova senha em uma solicitação da API `ConfirmForgotPassword`.
    f. A aplicação redireciona o usuário para fazer login.

Trabalhar com atributos do usuário

Os atributos são informações que ajudam a identificar usuários específicos, como nome, endereço de e-mail e número de telefone. Um novo grupo de usuários tem um conjunto padrão de atributos. Você também pode adicionar atributos personalizados à sua definição de grupo de usuários no Console de gerenciamento da AWS. Este tópico descreve esses atributos detalhadamente e oferece dicas sobre como configurar seu grupo de usuários.

Não armazene todas as informações sobre seus usuários nos atributos. Por exemplo, mantenha os dados do usuário que são alterados com frequência, como estatísticas de uso ou pontuações de jogos, em um repositório de dados separado, como o Amazon Cognito Sync ou o Amazon DynamoDB.

Limpe as entradas dos valores de string de atributos do usuário antes de enviá-los ao grupo de usuários. Um método para analisar os valores de atributos do usuário propostos é utilizando um acionador do Lambda, como o de pré-cadastro.

> **Note**
>
> Alguns documentos e padrões se referem a atributos como membros.

Tópicos

Atributos padrão

O Amazon Cognito atribui a todos os usuários um conjunto de atributos padrão com base na Especificação do OpenID Connect. Por padrão, os valores de atributo padrão e personalizados podem ser qualquer string de até 2.048 caracteres, mas alguns valores têm restrições de formato.

Os atributos padrão são:

- `name`
- `family_name`
- `given_name`
- `middle_name`
- `nickname`
- `preferred_username`
- `profile`
- `picture`
- `website`
- `gender`
- `birthdate`

- `zoneinfo`
- `locale`
- `updated_at`
- `address`
- `email`
- `phone_number`
- `sub`

Exceto `sub`, os atributos padrão são opcionais por padrão para todos os usuários. Para tornar um atributo obrigatório, durante o processo de criação do grupo de usuários, marque a caixa de seleção Required (Obrigatório) ao lado do atributo. O Amazon Cognito atribui um valor de identificador de usuário exclusivo ao atributo `sub` de cada usuário. Somente os atributos email e phone_number podem ser verificados.

Os atributos padrão têm propriedades predefinidas que você pode visualizar no `SchemaAttributes` parâmetro de uma resposta da DescribeUserPool API. Você pode definir valores personalizados para essas propriedades de atributos, como restrições de tipo de dados, mutabilidade e comprimento. Para modificar as propriedades do atributo padrão, defina seus valores personalizados no parâmetro CreateUserPool Esquema. O esquema também representa onde você define os atributos necessários. Você não pode modificar propriedades de atributos padrão ao criar grupos de usuários no console do Amazon Cognito.

> **Note**
>
> Quando você marcar um atributo padrão como Required (Obrigatório), o usuário não poderá se inscrever, a menos que forneça um valor para o atributo. Para criar usuários e não fornecer valores para os atributos necessários, os administradores podem usar a AdminCreateUserAPI. Após a criação de um grupo de usuários, não é possível alternar um atributo entre obrigatório e não obrigatório.

Detalhes do atributo padrão e restrições de formato

birthdate

O valor deve ser uma data válida de 10 caracteres no formato YYYY-MM-DD.

email

Os usuários e administradores podem verificar valores de endereço de e-mail.

Um administrador com Conta da AWS as permissões adequadas pode alterar o endereço de e-mail do usuário e também marcá-lo como verificado. Marque um endereço de e-mail como verificado com a AdminUpdateUserAttributesAPI ou o comando admin-update-user-attributes AWS Command Line Interface (AWS CLI). Com esse comando, o administrador pode alterar o atributo `email_verified` para `true`. Você também pode editar um usuário no menu Usuários do console do Amazon Cognito para marcar um endereço de e-mail como verificado.

O valor deve ser uma string de endereço de e-mail válida seguindo o formato de e-mail padrão com o símbolo @ e o domínio, com até 2.048 caracteres.

phone_number

O usuário deverá fornecer um número de telefone se a autenticação multifator (MFA) de SMS estiver ativa. Para obter mais informações, consulte Adicionar MFA a um grupo de usuários.

Os usuários e administradores podem verificar valores de número de telefone.

Um administrador com Conta da AWS as permissões adequadas pode alterar o número de telefone do usuário e também marcá-lo como verificado. Marque um número de telefone como verificado com a AdminUpdateUserAttributesAPI ou o admin-update-user-attributes AWS CLI comando. Com esse comando, o administrador pode alterar o atributo `phone_number_verified` para `true`. Você também pode editar um usuário no menu Usuários do console do Amazon Cognito para marcar um número de telefone como verificado.

> **Important**
>
> Os números de telefone devem seguir estas regras de formatação: devem começar com um sinal de mais (**+**), seguido imediatamente do código do país. Um número de telefone pode conter apenas o sinal **+** e os dígitos. Remova quaisquer outros caracteres de um número de telefone, como parênteses, espaços ou traços (**-**) antes de enviar o valor ao serviço. Por exemplo, um número de telefone dos Estados Unidos deve seguir este formato: **+14325551212**.

preferred_username

Você pode selecionar `preferred_username` conforme necessário ou como um alias, mas não ambos. Se `preferred_username` for um alias, você pode fazer uma solicitação para a operação da UpdateUserAttributesAPI e adicionar o valor do atributo depois de confirmar o usuário.

sub

Indexe e pesquise seus usuários com base no atributo `sub`. O atributo `sub` é um identificador de usuário exclusivo em cada grupo de usuários. Os usuários podem alterar atributos como `phone_number` e `email`. O atributo `sub` tem um valor fixo. Para ter mais informações sobre como descobrir usuários, consulte Como gerenciar e pesquisar contas de usuários.

Exibir atributos obrigatórios

Siga o procedimento abaixo a fim de exibir atributos obrigatórios para um determinado grupo de usuários.

> **Note**
>
> Não é possível alterar atributos obrigatórios após a criação de um grupo de usuários.

Para exibir atributos obrigatórios

1. Acesse o Amazon Cognito no. Console de gerenciamento da AWS Se o console solicitar, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Cadastrar-se.
5. Na seção Required attributes (Atributos obrigatórios), veja os atributos obrigatórios de seu grupo de usuários.

Nome de usuário e nome de usuário preferencial

O valor de `username` é um atributo separado; não equivale ao atributo `name`. Cada usuário tem um atributo `username`. O Amazon Cognito gera automaticamente um nome de usuário para usuários

federados. Você deve fornecer um atributo `username` para criar um usuário local no diretório do Amazon Cognito. Após a criação de um usuário, não é possível alterar o valor do atributo `username`.

Os desenvolvedores podem usar o atributo `preferred_username` para atribuir aos usuários nomes de usuário que eles podem alterar. Para obter mais informações, consulte Personalização dos atributos de login.

Se sua aplicação não exigir um nome de usuário, não será necessário solicitar que os usuários o forneçam. O aplicativo pode criar um nome de usuário exclusivo para os usuários no plano de fundo. Isso pode ser útil se você desejar que os usuários se inscrevam e façam login com um endereço de e-mail e senha. Para obter mais informações, consulte Personalização dos atributos de login.

`username` deve ser exclusivo em um grupo de usuários. `username` pode ser reutilizado, mas somente depois que você o excluir e ele não estiver mais em uso. Para obter informações sobre as restrições de string aos `username` atributos, consulte a propriedade username de uma solicitação de SignUpAPI.

Personalização dos atributos de login

Ao criar um grupo de usuários, você pode configurar atributos de nome de usuário caso queira que os usuários possam se inscrever e fazer login com um endereço de e-mail ou um número de telefone como nome de usuário. Como alternativa, você pode configurar atributos de alias para dar aos usuários a seguinte opção: incluir vários atributos ao se inscreverem e, então, fazer login com um nome de usuário, nome de usuário preferencial, endereço de e-mail ou número de telefone.

> **Important**
>
> Após a criação de um grupo de usuários, não é possível alterar essa configuração.

Como escolher entre atributos de alias e atributos de nome de usuário

Seu requisito	Atributos de alias	Atributos do nome de usuário
Os usuários têm vários atributos de login	Sim[1]	Não[2]
Os usuários devem verificar o endereço de e-mail ou o	Sim	Não

Seu requisito	Atributos de alias	Atributos do nome de usuário
número de telefone antes de poderem fazer login com ele		
Inscreva usuários com endereços de e-mail ou números de telefone duplicados e evite erros[3] `UsernameExistsException`	Sim	Não
Pode atribuir o mesmo valor de atributo de endereço de e-mail ou número de telefone a mais de um usuário	Sim[4]	Não

[1] Os atributos de login disponíveis são nome de usuário, endereço de e-mail, número de telefone e nome de usuário de preferência.

[2] É possível fazer login com o endereço de e-mail ou o número de telefone.

[3] O grupo de usuários não gera erros `UsernameExistsException` quando os usuários se registram com endereços de e-mail ou números de telefone possivelmente duplicados, mas sem nome de usuário. Esse comportamento é independente de Evitar erros de existência de nome de usuário, que se aplica às operações de login, mas não às operações de inscrição.

[4] Somente o último usuário que verificou o atributo pode fazer login com ele.

Opção 1: vários atributos de login (atributos de alias)

Um atributo é um alias quando os usuários têm um nome de usuário, mas também podem fazer login com esse atributo. Configure aliases quando quiser permitir que os usuários optem entre o nome de usuário e outros valores de atributo no campo de nome de usuário do formulário de login. O atributo `username` é um valor fixo que os usuários não podem mudar. Se você marcar um atributo como alias, os usuários poderão fazer login usando esse atributo em vez de o nome de usuário. Você pode marcar os atributos de endereço de e-mail, número de telefone e nome de usuário preferido como aliases. Por exemplo, se você selecionar um endereço de e-mail e número de telefone como aliases

para um grupo de usuários, os usuários desse grupo poderão fazer login usando o respectivo nome de usuário, endereço de e-mail ou número de telefone com a senha.

Para escolher atributos de alias, selecione User name (Nome de usuário) e pelo menos uma opção de login adicional ao criar o grupo de usuários.

> **Note**
>
> Ao configurar seu grupo de usuários para indistinção de maiúsculas e minúsculas, o usuário poderá usar letras minúsculas ou maiúsculas para se cadastrar ou fazer login com o alias. Para obter mais informações, consulte a Referência CreateUserPoolda API de grupos de usuários do Amazon Cognito.

Se você selecionar o endereço de e-mail como um alias, o Amazon Cognito não aceitará um nome de usuário que corresponda a um formato de endereço de e-mail válido. Da mesma forma, se você selecionar o número de telefone como alias, o Amazon Cognito não aceitará um nome de usuário para esse grupo de usuários que corresponda a um formato de número de telefone válido.

> **Note**
>
> Os valores de alias devem ser exclusivos em um grupo de usuários. Se configurar um alias para um endereço de e-mail ou número de telefone, o valor que você fornecer poderá apresentar o estado verificado em apenas uma conta. Durante o cadastro, se o usuário fornecer um endereço de e-mail ou número de telefone como um valor de alias e outro usuário já tiver usado esse valor, o registro será bem-sucedido. No entanto, quando um usuário tentar confirmar a conta com esse e-mail (ou número de telefone) e inserir o código válido, o Amazon Cognito retornará um erro `AliasExistsException`. Esse erro indica ao usuário que já existe uma conta com esse endereço de e-mail (ou número de telefone). Nesse ponto, o usuário pode abandonar a tentativa de criar a nova conta e, em vez disso, tentar redefinir a senha para a conta antiga. Se o usuário continuar criando a nova conta, sua aplicação deverá chamar a API `ConfirmSignUp` com a opção `forceAliasCreation`. `ConfirmSignUp` com `forceAliasCreation` move o alias da conta anterior para a conta recém-criada e marca o atributo não verificado na conta anterior.

Os números de telefone e os endereços de e-mail só se tornarão aliases ativos para um usuário depois que você os verificar. Recomendamos escolher a verificação automática dos endereços de e-mail e números de telefone se você os usar como aliases.

Escolha atributos de alias para evitar erros `UsernameExistsException` nos atributos de endereço de e-mail e número de telefone quando os usuários se inscreverem.

Ative o atributo `preferred_username` para que o usuário possa alterar o nome que ele usa para fazer login enquanto o valor de atributo `username` não mudar. Se desejar configurar essa experiência de usuário, envie o novo valor de `username` como `preferred_username` e escolha `preferred_username` como alias. Assim, os usuários poderão fazer login com o novo valor que eles inseriram. Se você selecionar `preferred_username` como alias, o usuário só poderá fornecer o valor quando confirmar uma conta. Ele não poderá fornecer o valor durante o registro.

Quando o usuário se inscreve com um nome de usuário, é possível escolher se ele pode fazer login com um ou mais dos aliases a seguir.

- Um endereço de e-mail verificado
- Um número de telefone verificado
- Nome de usuário preferido

Depois que o usuário se cadastra, ele não pode alterar esses aliases.

> **Important**
>
> Quando seu grupo de usuários é compatível com login com aliases e você deseja autorizar ou pesquisar um usuário, não identifique seu usuário por nenhum de seus atributos de login. O identificador de usuário de valor fixo `sub` é o único indicador consistente da identidade do seu usuário.

Inclua as etapas a seguir quando criar o grupo de usuários para que os usuários possam fazer login com um alias.

Phone number or email address (console)

Você deve definir o endereço de e-mail e o número de telefone como atributos de alias ao criar um grupo de usuários.

Como criar um grupo de usuários com aliases de nome de usuário no console do Amazon Cognito

1. Acesse o Amazon Cognito no Console de gerenciamento da AWS. Se o console solicitar, insira suas AWS credenciais.
2. Crie um novo grupo de usuários com o botão Comece a usar ou Criar grupo de usuários.
3. Escolha as configurações da aplicação em Definir a aplicação.
4. Em Configurar opções, em Opções para identificadores de login, marque a caixa de seleção ao lado de Nome de usuário e pelo menos uma das outras opções, E-mail e Número de telefone.
5. Escolha os atributos de alias como Atributos obrigatórios para a inscrição. No formulário de cadastro de login gerenciado, o Amazon Cognito solicita que novos usuários forneçam valores para os atributos obrigatórios.
6. Em Adicionar um URL de retorno, configure um URL de retorno de chamada da aplicação para redirecionamento após o login de login gerenciado.
7. Escolha Criar.

Phone number or email address (API/SDK)

Crie um novo grupo de usuários com a operação CreateUserPoolda API. Configure o parâmetro `AliasAttributes` conforme mostrado. Você pode remover a entrada `email` se quiser somente aliases de números de telefone ou remover a entrada `phone_number` se quiser somente aliases de endereço de e-mail.

```
"AliasAttributes": [
   "email",
   "phone_number"
],
```

Preferred username (API/SDK)

O console do Amazon Cognito cria grupos de usuários sem `preferred_username` como um alias. Para criar grupos de usuários com um `preferred_username` alias, configure grupos de usuários com solicitações de CreateUserPoolAPI em um AWS SDK. Para oferecer suporte à criação de atributos de nome de usuário preferenciais no cadastro, defina `preferred_username` como um atributo obrigatório. No formulário de cadastro de login gerenciado, o Amazon Cognito solicita que novos usuários forneçam valores para os atributos

obrigatórios. Você pode definir `preferred_username` como um atributo obrigatório no console do Amazon Cognito, mas isso não o torna disponível como um alias.

Configurar como um alias

Configure `preferred_username` como um alias no parâmetro `AliasAttributes` de uma solicitação `CreateUserPool`, conforme mostrado. Remova todos os valores que você não deseja como atributos de alias da lista.

```
"AliasAttributes": [
   "email",
   "phone_number",
   "preferred_username"
],
```

Configurar como obrigatório

No formulário de cadastro de login gerenciado, o Amazon Cognito solicita que novos usuários forneçam valores para os atributos obrigatórios. Configure `preferred_username` conforme necessário no `SchemaAttributes` parâmetro de uma CreateUserPoolsolicitação.

Para definir o nome de usuário preferencial como um atributo obrigatório, configure-o conforme mostrado. O exemplo a seguir modifica o esquema padrão de `preferred_username` para torná-lo obrigatório. Outros parâmetros do esquema, como `AttributeDataType` (usa `string` como padrão) e `StringAttributeConstraints` (usa de 1 a 99 caracteres como padrão), assumem valores padrão.

```
"Schema": [
   {
      "Name": "preferred_username",
      "Required": true
   }
]
```

Opção 2: endereço de e-mail ou número de telefone como atributo de login (atributos de nome de usuário)

Quando o usuário se inscreve com um endereço de e-mail ou número de telefone como o respectivo nome de usuário, é possível escolher se ele pode se inscrever apenas com endereços de e-mail, apenas números de telefone ou qualquer um dos dois.

Para escolher atributos de nome de usuário, não selecione Nome de usuário como opção de login ao criar o grupo de usuários.

O endereço de e-mail ou o número de telefone deve ser exclusivo e não deve estar em uso por outro usuário. Ele não precisa ser verificado. Depois que o usuário se cadastra com um endereço de e-mail ou número de telefone, ele não pode criar uma nova conta com o mesmo endereço de e-mail ou número de telefone. O usuário só poderá reutilizar a conta existente e redefinir a respectiva senha, se necessário. No entanto, ele pode alterar o endereço de e-mail ou o número de telefone para um novo endereço de e-mail ou número de telefone. Se o endereço de e-mail ou o número de telefone ainda não estiver em uso, ele se tornará o novo nome de usuário.

Ao selecionar endereço de e-mail e número de telefone como atributos de nome de usuário, os usuários podem fazer login com um ou outro, mesmo que forneçam valores para ambos os atributos. O nome de usuário de login é baseado no valor que você passa no `Username` parâmetro de. SignUp

> **Note**
>
> Se um usuário se inscrever com um endereço de e-mail como nome de usuário, ele poderá alterar o nome de usuário para outro endereço de e-mail, mas não poderá alterá-lo para um número de telefone. Se os usuários se inscreverem com um número de telefone, eles poderão alterar o nome de usuário para outro número de telefone, mas não poderão alterá-lo para um endereço de e-mail.

Siga as etapas abaixo durante o processo de criação do grupo de usuários para configurar o cadastro e o login com um endereço de e-mail ou número de telefone.

Username attributes (console)

O procedimento a seguir cria um grupo de usuários com atributos de nome de usuário de endereço de e-mail ou número de telefone. A diferença no processo de atributos de nome de usuário no console do Amazon Cognito é que você também não define o Nome de usuário como um atributo de login.

Como criar um grupo de usuários com atributos de nome de usuário no console do Amazon Cognito

1. Acesse o Amazon Cognito no Console de gerenciamento da AWS. Se o console solicitar, insira suas AWS credenciais.

2. Crie um novo grupo de usuários com o botão Comece a usar ou Criar grupo de usuários.
3. Escolha as configurações da aplicação em Definir a aplicação.
4. Em Configurar opções, em Opções para identificadores de login, selecione os atributos de nome de usuário: E-mail, Número de telefone ou ambos. Deixe Nome de usuário desmarcado.
5. Como prática recomendada, selecione os atributos de nome de usuário como Atributos obrigatórios para inscrição. No formulário de cadastro de login gerenciado, o Amazon Cognito solicita que novos usuários forneçam valores para os atributos obrigatórios. Se você não definir os atributos de nome de usuário como obrigatórios, o Amazon Cognito não solicitará que os novos usuários os forneçam. Nesse cenário, você deve configurar sua aplicação para coletar e enviar endereços de e-mail ou números de telefone de cada usuário antes que ele possa fazer login.
6. Em Adicionar um URL de retorno, configure um URL de retorno de chamada da aplicação para redirecionamento após o login de login gerenciado.
7. Escolha Criar.

Username attributes (API/SDK)

Em uma CreateUserPoolsolicitação, configure o `UsernameAttributes` parâmetro conforme mostrado. Para permitir o login somente com nomes de usuário de endereço de e-mail, especifique `email` sozinho nessa lista. Para permitir o login somente com nomes de usuário de números de telefone, especifique `phone_number` sozinho. Esse parâmetro substitui o nome de usuário como opção de login.

```
"UsernameAttributes": [
   "email",
   "phone_number"
],
```

Ao configurar atributos de nome de usuário, você pode fazer solicitações de SignUpAPI que transmitem um endereço de e-mail ou número de telefone no `username` parâmetro. Veja a seguir o comportamento da operação de API `SignUp` com atributos de nome de usuário.

- Se a string `username` estiver no formato válido de endereço de e-mail, por exemplo, `user@example.com`, o grupo de usuários preencherá automaticamente o atributo `email` do usuário com o valor `username`.

- Se a string `username` estiver no formato válido de número de telefone, por exemplo, `+12065551212`, o grupo de usuários ocupa automaticamente o atributo `phone_number` do usuário com o valor `username`.
- Se a string `username` não estiver no formato de endereço de e-mail ou número de telefone, a API `SignUp` retornará uma exceção.
- Se a string `username` contiver um endereço de e-mail ou número de telefone já em uso, a API `SignUp` retornará uma exceção.
- A API `SignUp` preenche o atributo `username` com um UUID para seu usuário. Este UUID possui o mesmo valor reivindicado pelo `sub` no token de identidade do usuário.

Você pode usar um endereço de e-mail ou número de telefone no lugar do nome de usuário em tudo APIs, exceto na ListUsersoperação. Nas solicitações de API `ListUsers`, é possível especificar um `Filter` de `email` ou `phone_number`. Se você filtrar por `username`, deverá fornecer o nome de usuário do UUID, não o endereço de e-mail ou número de telefone.

Atributos personalizados

Você pode adicionar até 50 atributos personalizados ao grupo de usuários. Você pode especificar um comprimento mínimo e/ou máximo para os atributos personalizados. No entanto, o comprimento máximo para qualquer atributo personalizado não pode ultrapassar 2.048 caracteres. O nome de um atributo personalizado deve corresponder ao padrão de expressão regular descrito no `Name` parâmetro de SchemaAttributeType.

Todo atributo personalizado tem as seguintes características:

- Você pode defini-lo como uma string, número, booleano ou objeto `DateTime`. O Amazon Cognito grava valores de atributo personalizados no token de ID somente como strings.

> **Note**
>
> No console do Amazon Cognito, é possível adicionar atributos personalizados somente dos tipos de dados de string e número. Opções adicionais, como tipos de dados booleanos e de `DateTime` atributos, só estão disponíveis na `SchemaAttributes` propriedade CreateUserPoole nas solicitações da UpdateUserPoolAPI.

- Não é possível exigir que os usuários forneçam um valor para o atributo.
- Você não poderá removê-lo ou alterá-lo depois de adicioná-lo ao grupo de usuários.

- A extensão de caracteres do nome do atributo está dentro do limite que o Amazon Cognito aceita. Para obter mais informações, consulte Cotas no Amazon Cognito.
- Ele pode ser mutável ou imutável. É possível gravar um valor em um atributo imutável ao criar um usuário. Você pode alterar o valor de um atributo mutável se o cliente de aplicação tiver permissão de gravação para o atributo. Consulte Permissões e escopos do atributo para obter mais informações.

> **Note**
>
> No código e nas configurações de regra de Controle de acesso com base em perfil, os atributos personalizados requerem o prefixo `custom:` para que sejam diferenciados dos atributos padrão.

Você também pode adicionar atributos de desenvolvedor ao criar grupos de usuários, na `SchemaAttributes` propriedade de CreateUserPool. Os atributos de desenvolvedor têm um prefixo `dev:`. Você só pode modificar os atributos de desenvolvedor de um usuário com AWS credenciais. Os atributos de desenvolvedor são um recurso herdado que o Amazon Cognito substituiu pelas permissões de leitura e gravação do cliente da aplicação.

Siga o procedimento abaixo para criar um novo atributo personalizado.

Para adicionar um atributo personalizado usando o console

1. Acesse o Amazon Cognito no. Console de gerenciamento da AWS Se o console solicitar, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Cadastrar-se e, na seção Atributos personalizados, selecione Adicionar atributos personalizados.
5. Na página Add custom attributes (Adicionar atributos personalizados), forneça os seguintes detalhes sobre o novo atributo:
   - Insira um Name (Nome).
   - Selecione um Type (Tipo), que pode ser String ou Number (Número).
   - Insira um tamanho de string Min. (Mínimo) ou um valor numérico.

- Insira um tamanho de string Max. (Máximo) ou um valor numérico.
- Selecione Mutable (Mutável) se quiser conceder permissão aos usuários para alterar o valor de um atributo personalizado depois que eles definirem o valor inicial.

6. Escolha Salvar alterações.

Permissões e escopos do atributo

Para cada aplicação cliente, é possível configurar permissões de leitura e gravação para cada atributo de usuário. Dessa forma, você pode controlar o acesso de leitura que qualquer aplicação tiver e modificar cada atributo armazenado para seus usuários. Por exemplo, você pode ter um atributo personalizado que indique se um usuário é ou não um cliente pagante. Suas aplicações podem ver esse atributo, mas não o alterar diretamente. Em vez disso, você atualizará o atributo usando uma ferramenta administrativa ou um processo em segundo plano. Você pode definir permissões para atributos de usuário no console do Amazon Cognito, na API do Amazon Cognito ou na AWS CLI. Por padrão, todos os novos atributos personalizados só estarão disponíveis depois que você definir permissões de leitura e gravação para eles. Por padrão, quando você cria um cliente de aplicação, concede à aplicação permissões de leitura e gravação para todos os atributos padrão e personalizados. Para limitar a aplicação somente à quantidade de informações necessárias, atribua permissões específicas aos atributos na configuração do cliente da aplicação.

Como prática recomendada, especifique as permissões de leitura e gravação dos atributos ao criar um cliente de aplicação. Conceda ao cliente de aplicação acesso ao conjunto mínimo de atributos de usuário necessários para a operação da aplicação.

> **Note**
>
> DescribeUserPoolClientretorna somente valores para `ReadAttributes` e `WriteAttributes` quando você configura permissões do cliente do aplicativo que não sejam as padrão.

Como atualizar as permissões de atributo (Console de gerenciamento da AWS)

1. Acesse o Amazon Cognito no. Console de gerenciamento da AWS Se o console solicitar, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.

4. Clique no menu Clientes da aplicação e selecione um cliente da aplicação na lista.
5. Na guia Permissões de atributos, selecione Editar.
6. Na página Edit attribute read and write permissions (Editar permissões de leitura e gravação de atributos), configure suas permissões de leitura e gravação e, em seguida, escolha Save changes (Salvar alterações).

Repita essas etapas para cada cliente de aplicativo usando o atributo personalizado.

Para cada cliente de aplicação, é possível marcar atributos como legíveis ou graváveis. Isso se aplica aos atributos padrão e aos atributos personalizados. A aplicação pode recuperar o valor dos atributos que você marca como legíveis, bem como definir ou modificar o valor dos atributos marcados como graváveis. Se seu aplicativo tentar definir um valor para um atributo que não está autorizado a escrever, o Amazon Cognito retornará. `NotAuthorizedException` GetUseras solicitações incluem um token de acesso com uma reivindicação do cliente do aplicativo; o Amazon Cognito retorna somente valores para atributos que o cliente do aplicativo pode ler. O token de ID do usuário de uma aplicação contém apenas declarações que correspondem aos atributos legíveis. Todos os clientes da aplicação podem gravar os atributos necessários do grupo de usuários. Você só pode definir o valor de um atributo em uma solicitação de API de grupos de usuários do Amazon Cognito quando também fornece um valor para quaisquer atributos obrigatórios que ainda não tenham um valor.

Os atributos personalizados têm recursos distintos para permissões de leitura e gravação. É possível criá-los como mutáveis ou imutáveis para o grupo de usuários e defini-los como atributos de leitura ou gravação para qualquer cliente da aplicação.

Um atributo personalizado imutável pode ser atualizado uma vez durante a criação do usuário. É possível preencher um atributo imutável com os métodos a seguir.

- `SignUp`: um usuário se inscreve em um cliente da aplicação que tenha acesso de gravação a um atributo personalizado imutável. Ele fornece um valor para esse atributo.
- Fazer login com um IdP de terceiros: um usuário faz login em um cliente da aplicação que tem acesso de gravação a um atributo personalizado imutável. A configuração do grupo de usuários para o IdP tem uma regra para associar uma declaração fornecida a um atributo imutável. Isso é possível, mas não prático, porque o usuário só poderá fazer login uma vez. O Amazon Cognito rejeita todas as tentativas de login após a primeira, pois existe uma regra de mapeamento para um atributo now-unwriteable.
- `AdminCreateUser`: você fornece um valor para um atributo imutável.

Permissões de atributos com escopos

Nos grupos de usuários que você configura com um AWS SDK ou CDK, a API REST ou o AWS CLI, você pode configurar o acesso de leitura ou gravação do cliente do aplicativo com o escopo do OIDC. `oidc:profile` O `oidc:profile` concede acesso de leitura ou gravação aos seguintes atributos padrão:

- `name`
- `family_name`
- `given_name`
- `middle_name`
- `nickname`
- `preferred_username`
- `profile`
- `picture`
- `website`
- `gender`
- `birthdate`
- `zoneinfo`
- `locale`

Essa lista contém os atributos padrão do OIDC menos `email`, `phone_number`, `sub` e `address`, conforme definido na seção 2.4 da especificação do OIDC. Para receber informações sobre os escopos que você pode atribuir aos clientes da aplicação, consulte Escopos, M2M e servidores de recursos.

Para configurar seu cliente de aplicativo para gravar nos atributos sob o `oidc:profile` escopo, defina o valor de WriteAttributestoo`idc:profile`, além de quaisquer outros atributos que você queira permitir que seu aplicativo modifique, em uma CreateUserPoolClientsolicitação de UpdateUserPoolClientAPI. Da mesma forma, para conceder acesso de leitura a esses atributos, adicione `oidc:profile` ao valor de ReadAttributes.

Você pode alterar os escopos e as permissões de atributo após ter criado o grupo de usuários.

Compreendendo os tokens web JSON do grupo de usuários () JWTs

Os tokens são artefatos de autenticação que as aplicações podem usar como prova de autenticação OIDC e para solicitar acesso aos recursos. As reivindicações em tokens são informações sobre o usuário. O token de ID contém declarações sobre a identidade dele, como nome de usuário, nome de família e endereço de e-mail. O token de acesso contém afirmações como as `scope` que o usuário autenticado pode usar para acessar terceiros APIs, operações de API de autoatendimento para usuários do Amazon Cognito e o. endpoint userinfo Tanto o token de acesso quanto o de ID incluem uma declaração `cognito:groups` que contém a associação do usuário ao grupo de usuários. Para obter mais informações sobre grupos de usuários, consulte Como adicionar grupos a um grupo de usuários.

O Amazon Cognito também tem tokens de atualização que você pode usar para obter novos tokens ou revogar tokens existentes. Refresh a token (Atualizar um token) para recuperar novos tokens de ID e de acesso. Revogar um token para revogar o acesso de usuário permitido por tokens de atualização.

O Amazon Cognito emite tokens como strings codificadas em base64url. Você pode decodificar qualquer token de ID ou acesso do Amazon Cognito de `base64url` para JSON em texto sem formatação. Os tokens de atualização do Amazon Cognito são criptografados, opacos para usuários e administradores de grupos de usuários e só podem ser lidos pelo seu grupo de usuários.

Autenticação com tokens

Quando um usuário faz login na sua aplicação, o Amazon Cognito verifica as informações de login. Se o login for bem-sucedido, o Amazon Cognito criará uma sessão e retornará um token de ID, um de acesso e um de atualização para o usuário autenticado. Você pode usar os tokens para conceder aos seus usuários acesso a recursos downstream, APIs como o Amazon API Gateway. Outra opção é trocá-los por credenciais da AWS temporárias para acessar outros Serviços da AWS.

Armazenar tokens

Sua aplicação deve ser capaz de armazenar tokens de tamanhos variados. O tamanho do token pode mudar por vários motivos, entre eles, declarações adicionais, alterações nos algoritmos de codificação e alterações nos algoritmos de criptografia. Quando você habilita a revogação de token no grupo de usuários, o Amazon Cognito adiciona declarações de token web JSON, o que aumenta o tamanho deles. As novas declarações `origin_jti` e `jti` são adicionadas aos tokens de acesso e ID. Para obter mais informações sobre revogação de tokens, consulte Como revogar tokens.

> **Important**
>
> Como prática recomendada, proteja todos os tokens em trânsito e no armazenamento no contexto da aplicação. Os tokens podem conter informações de identificação pessoal sobre seus usuários e informações sobre o modelo de segurança que você usa para o grupo de usuários.

Personalização de tokens

É possível personalizar os tokens de acesso e ID transmitidos pelo Amazon Cognito à aplicação. Em um Acionador do Lambda antes da geração do token, é possível adicionar, modificar e suprimir declarações de token. O gatilho de pré-geração de tokens é uma função do Lambda para a qual o Amazon Cognito envia um conjunto padrão de declarações. As reivindicações incluem escopos OAuth 2.0, associação a grupos de grupos de usuários, atributos do usuário e outros. A função pode então aproveitar a oportunidade para fazer alterações em runtime e retornar declarações de token atualizadas para o Amazon Cognito.

Custos adicionais se aplicam à personalização do token de acesso com eventos da versão 2. Para mais informações, consulte Preço do Amazon Cognito.

Tópicos

- Como entender o token de identidade (ID)
- Como entender o token de acesso
- Tokens de atualização
- Encerrar sessões de usuário com revogação de token
- Como verificar tokens web JSON
- Gerenciar a expiração e o armazenamento em cache do token do grupo de usuários

Como entender o token de identidade (ID)

O token de ID é um token web JSON (JWT) que contém declarações sobre a identidade do usuário autenticado, como `name`, `email` e `phone_number`. Você pode usar essas informações de identidade dentro da aplicação. O token de ID também pode ser usado para autenticar usuários nos seus servidores de recursos ou aplicações de servidor. Você também pode usar um token de ID fora da aplicação com suas operações de API da Web. Nesses casos, é preciso verificar a assinatura do token de ID antes de confiar em qualquer solicitação dentro do token de ID. Consulte Como verificar tokens web JSON.

Você pode definir a validade do token de ID para qualquer valor entre cinco minutos e um dia. Esse valor pode ser definido para cada cliente da aplicação.

> **Important**
>
> Quando o usuário faz login com login gerenciado, o Amazon Cognito define cookies de sessão válidos por 1 hora. Se você usar login gerenciado para autenticação em sua aplicação e especificar uma duração mínima inferior a 1 hora para os tokens de acesso e ID, os usuários ainda terão uma sessão válida até que o cookie expire. Se o usuário tiver tokens que expiram durante a sessão de 1 hora, o usuário poderá atualizar os respectivos tokens sem precisar se autenticar novamente.

Cabeçalho do token de ID

O cabeçalho contém duas informações: o ID de chave (`kid`) e o algoritmo (`alg`).

```
{
"kid" : "1234example=",
"alg" : "RS256"
}
```

`kid`

O ID da chave. Seu valor indica a chave usada para proteger a JSON web signature (JWS) do token. Você pode ver a chave de assinatura do grupo de usuários IDs no `jwks_uri` endpoint.

Para mais informações sobre o parâmetro `kid`, consulte Key identifier (kid) header parameter [Parâmetro de cabeçalho do identificador de chave (kid)].

`alg`

O algoritmo criptográfico que o Amazon Cognito usou para proteger o token de acesso. Os grupos de usuários usam um algoritmo RS256 criptográfico, que é uma assinatura RSA com SHA-256.

Para obter informações sobre o parâmetro `alg`, consulte Algorithm (alg) header parameter (Parâmetro de cabeçalho algoritmo [alg]).

Carga útil padrão do token de ID

Esta é uma carga útil de exemplo de um token de ID. Ela contém alegações sobre o usuário autenticado. Para obter mais informações sobre solicitações padrão OpenID Connect (OIDC), consulte a lista de declarações OpenID Connect. Você pode adicionar declarações de seu próprio padrão com um Acionador do Lambda antes da geração do token.

```
<header>.{
    "sub": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
    "cognito:groups": [
        "test-group-a",
        "test-group-b",
        "test-group-c"
    ],
    "email_verified": true,
    "cognito:preferred_role": "arn:aws:iam::111122223333:role/my-test-role",
    "iss": "https://cognito-idp.us-west-2.amazonaws.com/us-west-2_example",
    "cognito:username": "my-test-user",
    "middle_name": "Jane",
    "nonce": "abcdefg",
    "origin_jti": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
    "cognito:roles": [
        "arn:aws:iam::111122223333:role/my-test-role"
    ],
    "aud": "xxxxxxxxxxxxexample",
    "identities": [
        {
            "userId": "amzn1.account.EXAMPLE",
            "providerName": "LoginWithAmazon",
            "providerType": "LoginWithAmazon",
            "issuer": null,
            "primary": "true",
            "dateCreated": "1642699117273"
```

```
        }
    ],
    "event_id": "64f513be-32db-42b0-b78e-b02127b4f463",
    "token_use": "id",
    "auth_time": 1676312777,
    "exp": 1676316377,
    "iat": 1676312777,
    "jti": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
    "email": "my-test-user@example.com"
}
.<token signature>
```

sub

Um identificador exclusivo (UUID), ou assunto, do usuário autenticado. O nome de usuário pode não ser exclusivo em seu grupo de usuários. A reivindicação `sub` é a melhor maneira de identificar determinado usuário.

cognito:groups

Uma matriz dos nomes dos grupos de usuários que têm o usuário como membro. Os grupos podem ser um identificador que você apresenta à aplicação ou podem gerar uma solicitação para um perfil preferencial do IAM a partir de um banco de identidades.

cognito:preferred_role

O ARN do perfil do IAM que você associou ao grupo de grupos de usuários de maior prioridade do usuário. Para obter mais informações sobre como o grupo de usuários seleciona essa declaração de perfil, consulte Como atribuir valores de precedência a grupos.

iss

O provedor de identidades que emitiu o token. A reivindicação tem o formato a seguir.

`https://cognito-idp.<Region>.amazonaws.com/<your user pool ID>`

cognito:username

O nome do usuário no grupo de usuários.

nonce

A `nonce` declaração vem de um parâmetro com o mesmo nome que você pode adicionar às solicitações ao seu `authorize` endpoint OAuth 2.0. Quando você adiciona o parâmetro, `nonce` está incluído no token de ID que o Amazon Cognito emite e você pode usá-lo para se proteger

contra ataques de repetição. Se você não fornecer um valor para `nonce` em sua solicitação, o Amazon Cognito gera e valida automaticamente um nonce quando você se autentica por meio de um provedor de identidade de terceiros e, em seguida, adiciona-o como uma declaração `nonce` ao token de ID. A implementação da declaração `nonce` no Amazon Cognito é baseada nos padrões OIDC.

`origin_jti`

Um identificador de revogação de token associado ao token de atualização do seu usuário. O Amazon Cognito faz referência à `origin_jti` reivindicação quando verifica se você revogou o token do seu usuário com a operação da API Revogar endpoint ou da RevokeTokenAPI. Quando você revoga um token, o Amazon Cognito invalida todos os tokens de acesso e ID com o mesmo valor `origin_jti`.

`cognito:roles`

Uma matriz dos nomes dos perfis do IAM associados aos grupos do usuário. Cada grupo de usuários pode ter um perfil do IAM associado a ele. Essa matriz representa todos os perfis do IAM para os grupos do usuário, independentemente da precedência. Para obter mais informações, consulte Como adicionar grupos a um grupo de usuários.

`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.

`identities`

O conteúdo do atributo `identities` do usuário. O atributo contém informações sobre cada perfil de provedor de identidades de terceiros vinculado a um usuário, seja por login federado ou vinculando um usuário federado a um perfil local. Essas informações contêm o nome do provedor, o ID exclusivo dele e outros metadados.

`token_use`

A finalidade do token. Em um token de ID, seu valor é `id`.

`auth_time`

A hora de autenticação, no formato de hora Unix, em que o usuário concluiu a autenticação.

`exp`

O tempo de validade, no formato de horário Unix, em que o token do usuário expira.

`iat`

A emissão no momento, no formato de horário Unix, em que o Amazon Cognito emitiu o token do usuário.

`jti`

O identificador exclusivo do JWT.

O token de ID pode conter as declarações do padrão OIDC que estão definidas em OIDC standard claims (Declarações do padrão OIDC). Ele também pode conter atributos personalizados definidos por você no grupo de usuários. O Amazon Cognito grava valores de atributo personalizados no token de ID como strings, independentemente do tipo de atributo.

> **Note**
>
> Os atributos personalizados do grupo de usuários sempre são acompanhados de um prefixo `custom:`.

Assinatura do token de ID

A assinatura do token de ID é calculada com base no cabeçalho e na carga útil do token JWT. Antes de aceitar as reivindicações em qualquer token de ID recebido pela aplicação, verifique a assinatura do token. Para ter mais informações, consulte Como verificar um token Web JSON. Como verificar tokens web JSON

Como entender o token de acesso

O token de acesso ao grupo de usuários contém alegações sobre o usuário autenticado, uma lista dos grupos do usuário e uma lista de escopos. O objetivo do token de acesso é autorizar as operações de API. Seu grupo de usuários aceita tokens de acesso para autorizar as operações de autoatendimento do usuário. Por exemplo, é possível usar o token de acesso para conceder ao usuário acesso para adicionar, alterar ou excluir atributos de usuário.

Com escopos OAuth 2.0 em um token de acesso, derivados dos escopos personalizados que você adiciona ao seu grupo de usuários, você pode autorizar seu usuário a recuperar informações de uma API. Por exemplo, o Amazon API Gateway é compatível com a autorização com tokens de acesso do Amazon Cognito. Você pode preencher um autorizador de API REST com informações do grupo

de usuários ou usar o Amazon Cognito como um autorizador do token web JSON (JWT) para uma API HTTP. Para gerar um token de acesso com escopos personalizados, é necessário solicitá-lo por meio dos endpoints públicos do grupo de usuários.

Com o plano de recursos Essentials ou Plus, você também pode implementar um acionador do Lambda de pré-geração de tokens que adiciona escopos aos tokens de acesso em runtime. Para obter mais informações, consulte Acionador do Lambda antes da geração do token.

O token de acesso de um usuário com o escopo `openid` é a permissão para solicitar mais informações sobre os atributos do usuário no endpoint userinfo. A quantidade de informações do endpoint `userInfo` deriva dos escopos adicionais no token de acesso: por exemplo, `profile` para todos os dados do usuário, `email` para seu endereço de e-mail.

O token de acesso de um usuário com o escopo `aws.cognito.signin.user.admin` é a permissão para ler e gravar atributos do usuário, listar fatores de autenticação, configurar preferências de autenticação multifator (MFA) e gerenciar dispositivos memorizados. O nível de acesso aos atributos que seu token de acesso concede a esse escopo corresponde às read/write permissões de atributo que você atribui ao seu cliente do aplicativo.

O token de acesso é um JSON Web Token (JWT). O cabeçalho do token de acesso tem a mesma estrutura que o token de ID. O Amazon Cognito assina tokens de acesso com uma chave diferente da chave que assina os tokens de ID. O valor de uma reivindicação de ID de chave de acesso (`kid`) não corresponderá ao valor da reivindicação `kid` em um token de ID da mesma sessão do usuário. No código da aplicação, verifique os tokens de ID e os tokens de acesso de forma independente. Não confie nas reivindicações em um token de acesso até verificar a assinatura. Para obter mais informações, consulte Como verificar tokens web JSON. Você pode definir a validade do token de acesso para qualquer valor entre cinco minutos e um dia. Esse valor pode ser definido para cada cliente da aplicação.

> **Important**
>
> Para tokens de acesso e de ID, não especifique um mínimo inferior a 1 hora se você usar o login gerenciado. O login gerenciado define cookies de navegadores que são válidos por 1 hora. Se você configurar uma duração de token de acesso de menos de 1 hora, isso não afetará a validade do cookie de login gerenciado e a capacidade dos usuários de se autenticarem novamente sem credenciais adicionais por 1 hora após o login inicial.

Cabeçalho do token de acesso

O cabeçalho contém duas informações: o ID de chave (`kid`) e o algoritmo (`alg`).

```
{
"kid" : "1234example="
"alg" : "RS256",
}
```

`kid`

O ID da chave. Seu valor indica a chave usada para proteger a JSON web signature (JWS) do token. Você pode ver a chave de assinatura do grupo de usuários IDs no `jwks_uri` endpoint.

Para mais informações sobre o parâmetro `kid`, consulte Key identifier (kid) header parameter [Parâmetro de cabeçalho do identificador de chave (kid)].

`alg`

O algoritmo criptográfico que o Amazon Cognito usou para proteger o token de acesso. Os grupos de usuários usam um algoritmo RS256 criptográfico, que é uma assinatura RSA com SHA-256.

Para obter informações sobre o parâmetro `alg`, consulte Algorithm (alg) header parameter (Parâmetro de cabeçalho algoritmo [alg]).

Carga útil padrão do token de acesso

Esta é uma carga útil de exemplo de um token de acesso. Para mais informações, consulte JWT claims (Declarações JWT). Você pode adicionar declarações de seu próprio padrão com um Acionador do Lambda antes da geração do token.

```
<header>.
{
   "sub":"aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
   "device_key": "aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
   "cognito:groups":[
      "testgroup"
   ],
   "iss":"https://cognito-idp.us-west-2.amazonaws.com/us-west-2_example",
   "version":2,
   "client_id":"xxxxxxxxxxxxexample",
```

```
   "aud": "https://api.example.com",
   "origin_jti":"aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
   "event_id":"aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
   "token_use":"access",
   "scope":"phone openid profile resourceserver.1/appclient2 email",
   "auth_time":1676313851,
   "exp":1676317451,
   "iat":1676313851,
   "jti":"aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee",
   "username":"my-test-user"
}
.<token signature>
```

`sub`

Um identificador exclusivo (UUID), ou assunto, do usuário autenticado. O nome de usuário pode não ser exclusivo em seu grupo de usuários. A reivindicação `sub` é a melhor maneira de identificar determinado usuário.

`cognito:groups`

Uma matriz dos nomes dos grupos de usuários que têm o usuário como membro.

`iss`

O provedor de identidade que emitiu o token. A reivindicação tem o formato a seguir.

`https://cognito-idp.us-east-1.amazonaws.com/us-east-1_EXAMPLE`

`client_id`

O cliente de aplicação do grupo de usuários que autenticou o usuário. O Amazon Cognito renderiza o mesmo valor na reivindicação `aud` do token de ID.

aud

O URL da API para a qual o token de acesso se destina a autorizar. Presente somente se a aplicação solicitou uma vinculação de recursos do servidor de autorização.

`origin_jti`

Um identificador de revogação de token associado ao token de atualização do seu usuário. O Amazon Cognito faz referência à `origin_jti` reivindicação quando verifica se você revogou o token do seu usuário com a operação da API Revogar endpoint ou da RevokeTokenAPI. Quando você revoga um token, o Amazon Cognito não valida mais tokens de acesso e ID com o mesmo valor `origin_jti`.

`token_use`

A finalidade do token. Em um token de acesso, seu valor é `access`.

`scope`

Uma lista de escopos OAuth 2.0 emitida para o usuário conectado. Os escopos definem o acesso que o token fornece às operações externas APIs de autoatendimento do usuário e aos dados do usuário no `userInfo` endpoint. Um token do Endpoint de token pode conter qualquer escopo compatível com seu cliente de aplicação. Um token do login da API do Amazon Cognito contém somente o escopo `aws.cognito.signin.user.admin`.

`auth_time`

A hora de autenticação, no formato de hora Unix, em que o usuário concluiu a autenticação.

`exp`

O tempo de validade, no formato de horário Unix, em que o token do usuário expira.

`iat`

A emissão no momento, no formato de horário Unix, em que o Amazon Cognito emitiu o token do usuário.

`jti`

O identificador exclusivo do JWT.

`username`

O nome de usuário do usuário no grupo de usuários.

Mais atributos

- Como personalizar tokens de acesso nos grupos de usuários do Amazon Cognito

Assinatura do token de acesso

A assinatura do token de acesso, assinada com a chave anunciada no endpoint `.well-known/jwks.json`, valida a integridade do cabeçalho e da carga útil do token. Ao usar tokens de acesso para autorizar o acesso externo APIs, sempre configure seu autorizador de API para verificar essa assinatura em relação à chave que a assinou. Para obter mais informações, consulte Como verificar tokens web JSON.

Tokens de atualização

Você pode usar o token de atualização para recuperar novos tokens de ID e acesso. Por padrão, o token de atualização expira 30 dias depois que o usuário da aplicação fizer login no seu grupo de usuários. Ao criar uma aplicação para seu grupo de usuários, você pode definir a validade do token de atualização da aplicação em qualquer valor entre 60 minutos e 10 anos.

Obter novos tokens de acesso e identidade com um token de atualização

O Amazon Cognito emite tokens de atualização em resposta à autenticação bem-sucedida com o fluxo do código de autorização de login gerenciado e com operações de API ou métodos de SDK. O token de atualização retorna novos tokens de ID e acesso e, opcionalmente, um novo token de atualização. Você pode usar tokens de atualização das maneiras a seguir.

GetTokensFromRefreshToken

A operação GetTokensFromRefreshTokenda API emite novos tokens de ID e acesso a partir de um token de atualização válido. Você também receberá um novo token de atualização se tiver habilitado a alternância de tokens de atualização.

InitiateAuth and AdminitiateAuth

As operações InitiateAuthda API AdminInitiateAuthou incluem o fluxo de `REFRESH_TOKEN_AUTH` autenticação. Nesse fluxo, você transmite um token de atualização e recebe novos tokens de ID e acesso. Não é possível autenticar com `REFRESH_TOKEN_AUTH` em clientes da aplicação com a alternância de tokens de atualização habilitada.

OAuth ponto final do token

O endpoint de token em grupos de usuários com um domínio tem um tipo de concessão `refresh_token` que emite novos tokens de ID, tokens de acesso e, opcionalmente (com alternância de tokens de atualização), tokens de atualização com base em um token de atualização válido.

Alternância de tokens de atualização

Opcionalmente, é possível configurar a alternância de tokens de atualização no cliente da aplicação. Com a alternância de tokens de atualização, o cliente pode invalidar o token de atualização original e emitir um novo token de atualização a cada atualização de token. Quando essa configuração está habilitada, cada solicitação bem-sucedida em todas as formas de atualização de token retorna um novo token de ID, token de acesso e token de atualização. Quando essa configuração está

desabilitada, as solicitações de atualização de token retornam somente novos tokens de acesso e ID, e o token de atualização original permanece válido. O novo token de atualização é válido pela duração restante do token de atualização original. É possível configurar clientes da aplicação para alternar os tokens de atualização ou manter o token de atualização original. Para permitir novas tentativas por um breve período, também é possível configurar um período de carência para o token de atualização original de até 60 segundos.

O que é importante saber sobre a alternância de tokens de atualização

- Após habilitar a alternância de tokens de atualização, novas declarações são adicionadas aos tokens web JSON do grupo de usuários. As solicitações `origin_jti` e `jti` são adicionadas aos tokens de acesso e de ID. Essas reivindicações aumentam o tamanho do JWTs.
- A alternância de tokens de atualização não é compatível com o fluxo de autenticação `REFRESH_TOKEN_AUTH`. Para implementar a rotação do token de atualização, você deve desativar esse fluxo de autenticação no seu cliente de aplicativo e projetar seu aplicativo para enviar solicitações de atualização de token com a operação de GetTokensFromRefreshTokenAPI ou o método SDK equivalente.
- Com a alternância de tokens de atualização inativa, você pode concluir solicitações de atualização do token com `GetTokensFromRefreshToken` ou `REFRESH_TOKEN_AUTH`.
- Quando a memorização de dispositivos está ativa no grupo de usuários, é necessário fornecer a chave do dispositivo nas solicitações `GetTokensFromRefreshToken`. Se o usuário não tiver uma chave de dispositivo confirmada que sua aplicação envia na solicitação de autenticação inicial, o Amazon Cognito emitirá uma nova. Para atualizar os tokens nessa configuração, é necessário fornecer uma chave de dispositivo, independentemente de ter especificado uma em `AuthParameters` ou recebido uma nova na resposta de autenticação.
- Você pode transmitir `ClientMetadata` para o acionador do Lambda de pré-geração de tokens na solicitação `GetTokensFromRefreshToken`. Esses dados, que são transmitidos para o evento de entrada do acionador, fornecem contexto adicional que você pode usar na lógica personalizada da função do Lambda.

Como prática recomendada de segurança, habilite a alternância de tokens de atualização nos clientes da aplicação.

Enable refresh token rotation (console)

O procedimento a seguir ativa ou desativa a alternância de tokens de atualização para o cliente da aplicação. Esse procedimento requer um cliente da aplicação existente. Para saber mais sobre

como criar um cliente da aplicação, consulte Configurações específicas da aplicação com clientes de aplicação.

Como habilitar a alternância de token de atualização

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Navegue até o menu Clientes da aplicação e selecione um cliente da aplicação existente.
5. Clique em Editar na seção Informações do cliente de aplicação da página.
6. Em Configurações avançadas de segurança, localize a opção Habilitar alternância de tokens de atualização.
7. Para habilitar a alternância, marque a caixa de seleção. Para desabilitar a alternância, desmarque a caixa de seleção.
8. Em Período de carência de alternância de tokens de atualização, insira um número de segundos, até 60, que você deseja definir como o atraso antes que o token de atualização alternado seja revogado.

Enable refresh token rotation (API)

Configure a rotação do token de atualização em uma solicitação de UpdateUserPoolClientAPI CreateUserPoolClientou de uma solicitação. O corpo parcial da solicitação a seguir ativa a alternância de tokens de atualização e define o período de carência para 10 segundos.

```
"RefreshTokenRotation" : {
   "Feature" : "ENABLED,
   "RetryGracePeriodSeconds" : 10
}
```

Atualização de tokens de API e SDK

Há duas maneiras de usar o token de atualização para obter novos tokens de ID e acesso com a API de grupos de usuários, dependendo se a alternância de tokens de atualização está ativa. Em clientes de aplicativos com a rotação do token de atualização ativa, use a operação de GetTokensFromRefreshTokenAPI. Em clientes de aplicativos sem rotação de token de atualização, use o `REFRESH_TOKEN_AUTH` fluxo das operações AdminInitiateAuthou da InitiateAuthAPI.

> **Note**
>
> Os usuários podem se autenticar com grupos de usuários no login gerenciado ou em aplicativos personalizados que você cria com AWS SDKs as operações da API do Amazon Cognito. O fluxo `REFRESH_TOKEN_AUTH` e `GetTokensFromRefreshToken` podem concluir a atualização de token para usuários de login gerenciado. A atualização de token em aplicações personalizadas não afeta as sessões de login gerenciado. Essas sessões são definidas em um cookie do navegador e são válidas por 1 hora. A resposta `GetTokensFromRefreshToken` emite novos tokens de ID, de acesso e, opcionalmente, de atualização, mas não renova o cookie da sessão de login gerenciado. `REFRESH_TOKEN_AUTH` não está disponível em clientes da aplicação com a alternância de tokens de atualização habilitada.

GetTokensFromRefreshToken

GetTokensFromRefreshTokenretorna novos tokens de ID, acesso e atualização de uma solicitação que você autoriza com um token de atualização. Veja a seguir um exemplo de corpo da solicitação para `GetTokensFromRefreshToken`. Você pode enviar metadados do cliente para acionadores do Lambda em solicitações para essa operação.

```
{
    "RefreshToken": "eyJjd123abcEXAMPLE",
    "ClientId": "1example23456789",
    "ClientSecret": "myappclientsecret123abc",
    "ClientMetadata": {
      "MyMetadataKey" : "MyMetadataValue"
   },
}
```

AdminInitiateAuth/InitiateAuth

Para usar o token de atualização quando a rotação do token de atualização estiver inativa, use as operações de API AdminInitiateAuthou InitiateAuth. Transmita `REFRESH_TOKEN_AUTH` para o parâmetro `AuthFlow`. Na propriedade `AuthParameters` de `AuthFlow`, transmita o token de atualização do usuário como o valor de `"REFRESH_TOKEN"`. O Amazon Cognito retorna novos tokens de ID e acesso depois que sua solicitação à API passar por todos os desafios.

Veja a seguir um exemplo de corpo de solicitação para uma atualização de token com a API `InitiateAuth` ou `AdminInitiateAuth`.

```
{
    "AuthFlow": "REFRESH_TOKEN_AUTH",
    "ClientId": "1example23456789",
    "UserPoolId": "us-west-2_EXAMPLE",
    "AuthParameters": {
        "REFRESH_TOKEN": "eyJjd123abcEXAMPLE",
        "SECRET_HASH": "kT5acwCVrbD6JexhW3EQwnRSe6fLuPTRkEQ50athqv8="
    }
}
```

OAuth atualização de token

Você também pode enviar tokens de atualização para o Endpoint de token em um grupo de usuários em que configurou um domínio. No corpo da solicitação, inclua um valor `grant_type` de `refresh_token` e um valor `refresh_token` do token de atualização do usuário.

As solicitações para o endpoint do token estão disponíveis em clientes da aplicação com a alternância de tokens de atualização ativa e naqueles em que ela está inativa. Quando a alternância de tokens de atualização está ativa, o endpoint do token retorna um novo token de atualização.

Veja a seguir um exemplo de solicitação com um token de atualização.

```
POST /oauth2/token HTTP/1.1
Host: auth.example.com
Content-Type: application/x-www-form-urlencoded
Authorization: Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw
Content-Length: **

client_id=1example23456789&grant_type=refresh_token&refresh_token=eyJjd123abcEXAMPLE
```

Como revogar tokens de atualização

Você pode revogar tokens de atualização que pertencem a um usuário. Para obter mais informações sobre revogação de tokens, consulte Encerrar sessões de usuário com revogação de token.

> **Note**
>
> A revogação do token de atualização revogará todos os tokens de ID e acesso que o Amazon Cognito emitiu de solicitações de atualização com esse token.

Para desconectar os usuários de toda a sessão atual conectada, revogue todos os seus tokens ou solicitações de API. GlobalSignOutAdminUserGlobalSignOut Depois que o usuário é desconectado, os seguintes efeitos acontecem:

- O token de atualização do usuário não pode obter novos tokens para ele.
- O token de acesso do usuário não pode fazer solicitações de API autorizadas por token.
- O usuário precisa se autenticar novamente para obter novos tokens. Como os cookies de sessão de login gerenciado não expiram automaticamente, o usuário pode se autenticar novamente com um cookie de sessão, sem nenhuma solicitação adicional de credenciais. Após desconectar os usuários de login gerenciado, redirecione-os para o Endpoint de logout, onde o Amazon Cognito limpará o cookie da sessão.

Com os tokens de atualização, você pode manter as sessões dos usuários na aplicação por um longo tempo. Com o tempo, os usuários podem querer desautorizar algumas aplicações nas quais permaneceram conectados com seus tokens de atualização. Para desconectar o usuário de uma única sessão, revogue o token de atualização. Quando seu usuário quiser sair de todas as sessões autenticadas, gere uma solicitação de GlobalSignOutAPI. A aplicação pode oferecer ao usuário uma opção como Sair de todos os dispositivos. `GlobalSignOut` aceita o token de acesso válido/inalterado, não expirado e não revogado de um usuário. Como essa API é autorizada por token, um usuário não pode usá-la para iniciar a saída de outro usuário.

No entanto, você pode gerar uma solicitação de AdminUserGlobalSignOutAPI autorizada com suas AWS credenciais para desconectar qualquer usuário de todos os seus dispositivos. O aplicativo administrador deve chamar essa operação de API com credenciais de AWS desenvolvedor e passar o ID do grupo de usuários e o nome de usuário do usuário como parâmetros. A API `AdminUserGlobalSignOut` pode retirar qualquer usuário no grupo de usuários.

Para obter mais informações sobre solicitações que você pode autorizar com AWS credenciais ou com o token de acesso de um usuário, consulte. Lista de operações de API agrupadas por modelo de autorização

Encerrar sessões de usuário com revogação de token

É possível revogar tokens de atualização e encerrar sessões de usuário com os métodos a seguir. Quando você revoga um token de atualização, todos os tokens de acesso que foram emitidos anteriormente por esse token de atualização se tornam inválidos. Os outros tokens de atualização emitidos para o usuário não são afetados.

RevokeToken operação

RevokeTokenrevoga todos os tokens de acesso de um determinado token de atualização, incluindo o token de acesso inicial do login interativo. Essa operação não afeta nenhum dos outros tokens de atualização do usuário nem os filhos de token de ID e acesso desses outros tokens de atualização.

Endpoint de revogação

O endpoint de revogação revoga um determinado token de atualização e todos os tokens de ID e acesso que o token de atualização gerou. Esse endpoint também revoga o token de acesso inicial do login interativo. As solicitações para esse endpoint não afetam nenhum dos outros tokens de atualização do usuário nem os filhos de token de ID e acesso desses outros tokens de atualização.

GlobalSignOut operação

GlobalSignOuté uma operação de autoatendimento que um usuário autoriza com seu token de acesso. Essa operação revoga todos os tokens de atualização, ID e acesso do usuário solicitante.

AdminUserGlobalSignOut operação

AdminUserGlobalSignOuté uma operação do lado do servidor que um administrador autoriza com credenciais do IAM. Essa operação revoga todos os tokens de atualização, ID e acesso do usuário de destino.

O que é importante saber sobre a revogação de tokens

- A solicitação para revogar um token de atualização deve incluir ID do cliente que foi usado para obter o token.
- O grupo de usuários JWTs é independente, com uma assinatura e um prazo de expiração que foram atribuídos quando o token foi criado. Tokens revogados não podem ser usados com chamadas de API do Amazon Cognito que exijam um token. No entanto, os tokens revogados

ainda serão válidos se forem verificados usando qualquer biblioteca JWT que verifique a assinatura e a validade do token.

- Quando você cria um novo cliente do grupo de usuários, a revogação de token é habilitada por padrão.
- Você só pode revogar tokens de atualização em clientes da aplicação com a revogação de tokens habilitada.
- Após a habilitação da revogação de tokens, novas solicitações são adicionadas aos tokens web JSON do Amazon Cognito. As solicitações `origin_jti` e `jti` são adicionadas aos tokens de acesso e de ID. Essas solicitações aumentam o tamanho do acesso do cliente de aplicação e de tokens de ID.
- Quando você desabilita a revogação de tokens em um cliente da aplicação onde ela estava habilitada anteriormente, os tokens revogados não se tornam ativos novamente.
- Quando você desabilita uma conta de usuário (que revoga tokens de atualização e acesso), os tokens revogados não se tornam ativos se você habilitar a conta de usuário novamente.
- Quando você cria um novo cliente de grupo de usuários usando a Console de gerenciamento da AWS, a ou a AWS API AWS CLI, a revogação de token é ativada por padrão.

Habilitar revogação de token

Antes de poder revogar um token para um cliente de grupo de usuários existente, você deve habilitar a revogação de token. Você pode ativar a revogação de token para clientes de grupos de usuários existentes usando a AWS CLI ou a AWS API. Para isso, chame o comando de CLI `aws cognito-idp describe-user-pool-client` ou a operação de API `DescribeUserPoolClient` para recuperar as configurações atuais do cliente de aplicação. Depois, chame o comando de CLI `aws cognito-idp update-user-pool-client` ou a operação de API `UpdateUserPoolClient`. Inclua as configurações atuais do cliente de aplicação e defina o parâmetro `EnableTokenRevocation` como `true`.

Para criar ou modificar um cliente de aplicativo com a revogação de token habilitada com a API do Amazon Cognito ou com AWS um SDK, inclua o seguinte parâmetro na sua solicitação ou na CreateUserPoolClientsua solicitação de API. UpdateUserPoolClient

```
"EnableTokenRevocation": true
```

Para configurar a revogação do token no console do Amazon Cognito, selecione um cliente da aplicação no menu Clientes da aplicação no grupo de usuários. Clique no botão Editar em

Informações do cliente de aplicação e habilite ou desabilite a revogação do token em Configuração avançada.

Revogar um token

Você pode revogar um token de atualização usando uma solicitação de RevokeTokenAPI, por exemplo, com o comando CLI`aws cognito-idp revoke-token`. Você também pode revogar tokens usando o Revogar endpoint. Esse endpoint fica disponível depois que você adiciona um domínio ao seu grupo de usuários. Você pode usar o endpoint de revogação em um domínio hospedado do Amazon Cognito ou no seu próprio domínio personalizado.

Veja a seguir o corpo de um exemplo de uma solicitação de API `RevokeToken`.

```
{
   "ClientId": "1example23456789",
   "ClientSecret": "abcdef123456789ghijklexample",
   "Token": "eyJjdHkiOiJKV1QiEXAMPLE"
}
```

Veja a seguir um exemplo de solicitação cURL para o endpoint `/oauth2/revoke` de um grupo de usuários com um domínio personalizado.

```
curl --location 'auth.mydomain.com/oauth2/revoke' \
--header 'Content-Type: application/x-www-form-urlencoded' \
--header 'Authorization: Basic Base64Encode(client_id:client_secret)' \
--data-urlencode 'token=abcdef123456789ghijklexample' \
--data-urlencode 'client_id=1example23456789'
```

A operação `RevokeToken` e o endpoint `/oauth2/revoke` não precisam de autorização adicional, a menos que o cliente de aplicação tenha um segredo de cliente.

Como verificar tokens web JSON

Os tokens web JSON (JWTs) podem ser decodificados, lidos e modificados facilmente. Um token de acesso modificado cria um risco de escalonamento de privilégios. Um token de ID modificado cria um risco de falsificação de identidade. Sua aplicação confia no grupo de usuários como emissor de token, mas e se um usuário interceptar o token em trânsito? Você deve garantir que a aplicação receba o mesmo token emitido pelo Amazon Cognito.

O Amazon Cognito emite tokens que usam alguns dos recursos de integridade e confidencialidade da especificação do OpenID Connect (OIDC). Os tokens do grupo de usuários indicam a validade

com objetos como prazo de validade, emissor e assinatura digital. A assinatura, o terceiro e o último segmento do JWT delimitado por `.`, é o principal componente da validação do token. Um usuário mal-intencionado pode modificar um token, mas se sua aplicação recuperar a chave pública e comparar a assinatura, não haverá correspondência. Qualquer aplicativo processado JWTs a partir da autenticação OIDC deve realizar essa operação de verificação a cada login.

Nesta página, fazemos algumas recomendações gerais e específicas para verificação de JWTs. O desenvolvimento de aplicações abrange uma variedade de linguagens de programação e plataformas. Como o Amazon Cognito implementa o OIDC suficientemente próximo da especificação pública, qualquer biblioteca JWT confiável em seu ambiente de desenvolvedor preferencial pode lidar com seus requisitos de verificação.

Essas etapas descrevem como verificar um JSON web token (JWT) do grupo de usuários.

Tópicos

Pré-requisitos

Talvez sua biblioteca, SDK ou estrutura de software já realize as tarefas desta seção. AWS SDKs forneça ferramentas para manipulação e gerenciamento de tokens do grupo de usuários do Amazon Cognito em seu aplicativo. AWS Amplify inclui funções para recuperar e atualizar tokens do Amazon Cognito.

Para obter mais informações, consulte as páginas a seguir.

- Integração da autenticação e autorização do Amazon Cognito com aplicações móveis e da web
- Exemplos de código para o Amazon Cognito Identity Provider usando AWS SDKs
- Fluxos de trabalho avançados no Amplify Dev Center

Muitas bibliotecas estão disponíveis para decodificação e verificação de um JSON Web Token (JWT). Se você precisar processar tokens manualmente para o processamento da API no lado do servidor ou se estiver usando outras linguagens de programação, essas bibliotecas poderão ajudar. Consulte a OpenID foundation list of libraries for working with JWT tokens (Lista básica de bibliotecas da OpenID para trabalhar com tokens JWT).

Valide tokens com aws-jwt-verify

Em um aplicativo Node.js, AWS recomenda que a aws-jwt-verifybiblioteca valide os parâmetros no token que o usuário passa para o seu aplicativo. Com `aws-jwt-verify`, é possível preencher um `CognitoJwtVerifier` com os valores de reivindicação que você deseja verificar para um ou mais grupos de usuários. Alguns dos valores que ele pode verificar incluem o seguinte.

- Que os tokens de acesso ou ID não estão malformados nem expirados e têm uma assinatura válida.
- Que os tokens de acesso vieram dos grupos de usuários e clientes de aplicações corretos.
- Essas declarações de token de acesso contêm os escopos OAuth 2.0 corretos.
- Que as chaves que assinaram os tokens de acesso e ID correspondem a uma chave de assinatura `kid` do URI JWKS dos grupos de usuários.

  O URI do JWKS contém informações públicas sobre a chave privada que assinou o token do usuário. Você pode encontrar o URI do JWKS para seu grupo de usuários em `https://cognito-idp.<Region>.amazonaws.com/<userPoolId>/.well-known/jwks.json`.

Para obter mais informações e exemplos de códigos que você pode usar em um aplicativo Node.js ou em um AWS Lambda autorizador, consulte aws-jwt-verifyem GitHub.

Noções básicas e inspeções de tokens

Antes de integrar a inspeção de tokens ao seu aplicativo, considere como o Amazon Cognito é montado. JWTs Recupere exemplos de token do grupo de usuários. Decodifique-os e examine-os detalhadamente para entender suas características e determinar o que você deseja verificar e quando. Por exemplo, talvez você queira examinar a associação de grupo em um cenário e os escopos em outro.

As seções a seguir descrevem um processo para inspecionar manualmente o Amazon JWTs Cognito enquanto você prepara seu aplicativo.

Confirmar a estrutura do JWT

Um JSON Web Token (JWT) inclui três seções com um delimitador `.` (ponto) entre elas.

Cabeçalho

O ID da chave, o `kid` e o algoritmo RSA, o `alg`, que o Amazon Cognito usou para assinar o token. O Amazon Cognito assina tokens com um `alg` de `RS256`. `kid` é uma referência truncada a uma chave de assinatura privada RSA de 2.048 bits mantida pelo seu grupo de usuários.

Carga útil

Reivindicações de tokens. Em um token de ID, as reivindicações incluem atributos do usuário e informações sobre o grupo de usuários, o `iss` e o cliente da aplicação, o `aud`. Em um token de acesso, a carga útil inclui escopos, associação ao grupo, o grupo de usuários como `iss` e o cliente de aplicação como `client_id`.

Signature

A assinatura não é decodificável em base64url, como o cabeçalho e a carga útil. É um RSA256 identificador derivado de uma chave de assinatura e de parâmetros que você pode observar no URI do JWKS.

O cabeçalho e a carga útil são JSON codificados em base64url. É possível identificá-los pelos caracteres de abertura `eyJ` que são decodificados para o caractere inicial `{`. Se o usuário apresentar um JWT codificado em base64url para a aplicação e ele não estiver no formato `[JSON Header].[JSON Payload].[Signature]`, ele não é um token válido do Amazon Cognito e você poderá descartá-lo.

A aplicação de exemplo a seguir verifica os tokens do grupo de usuários com `aws-jwt-verify`.

```
// cognito-verify.js
// Usage example: node cognito-verify.js eyJra789ghiEXAMPLE

const { CognitoJwtVerifier } = require('aws-jwt-verify');

// Replace with your Amazon Cognito user pool ID
const userPoolId = 'us-west-2_EXAMPLE';

async function verifyJWT(token) {
  try {
    const verifier = CognitoJwtVerifier.create({
      userPoolId,
      tokenUse: 'access', // or 'id' for ID tokens
      clientId: '1example23456789', // Optional, only if you need to verify the token
 audience
```

```
    });

    const payload = await verifier.verify(token);
    console.log('Decoded JWT:', payload);
  } catch (err) {
    console.error('Error verifying JWT:', err);
  }
}

// Example usage
if (process.argv.length < 3) {
  console.error('Please provide a JWT token as an argument.');
  process.exit(1);
}

const MyToken = process.argv[2];
verifyJWT(MyToken);
```

Validar o JWT

A assinatura JWT é uma combinação com hash do cabeçalho e da carga útil. O Amazon Cognito gera dois pares de chaves criptográficas RSA para cada grupo de usuários. Uma chave privada assina tokens de acesso e a outra assina tokens de ID.

Para verificar a assinatura de um token JWT

1. Decodifique o token de ID.

   A OpenID Foundation também mantém uma lista de bibliotecas para trabalhar com tokens JWT.

   Você também pode usar AWS Lambda para decodificar o grupo JWTs de usuários. Para obter mais informações, consulte Decodificar e verificar os tokens JWT do Amazon Cognito usando AWS Lambda.
2. Compare o ID de chave local (`kid`) com o `kid` público.

   a. Faça download e armazene o JSON Web Key (JWK) público correspondente para seu grupo de usuários. Ele está disponível como parte de um JSON Web Key Set (JWKS). Você pode localizá-lo construindo o seguinte URI `jwks_uri` para seu ambiente:

      ```
      https://cognito-idp.<Region>.amazonaws.com/<userPoolId>/.well-known/jwks.json
      ```

Para mais informações sobre JWK e conjuntos JWK , consulte JSON Web Key (JWK).

> **Note**
>
> O Amazon Cognito pode alternar a chave de assinatura no grupo de usuários. Como prática recomendada, armazene as chaves públicas na aplicação usando o `kid` como chave de cache, e atualize o cache periodicamente. Compare o `kid` nos tokens que a aplicação recebe com o cache.
> Se você receber um token com o emissor correto, mas um `kid` diferente, o Amazon Cognito pode ter alternado a chave de assinatura. Atualize o cache do endpoint `jwks_uri` do grupo de usuários.

Este é um exemplo de arquivo `jwks.json`:

```
{
 "keys": [{
  "kid": "1234example=",
  "alg": "RS256",
  "kty": "RSA",
  "e": "AQAB",
  "n": "1234567890",
  "use": "sig"
 }, {
  "kid": "5678example=",
  "alg": "RS256",
  "kty": "RSA",
  "e": "AQAB",
  "n": "987654321",
  "use": "sig"
 }]
}
```

ID da chave (**`kid`**)

O `kid` é uma dica que indica qual foi a chave usada para proteger a assinatura web JSON (JWS) do token.

Algoritmo (**`alg`**)

O parâmetro de cabeçalho `alg` representa o algoritmo criptográfico usado para proteger o token de ID. Os grupos de usuários usam um algoritmo RS256 criptográfico, que é uma assinatura RSA com SHA-256. Para mais informações sobre RSA, consulte Criptografia RSA.

Tipo de chave (**`kty`**)

O parâmetro `kty` identifica o algoritmo criptográfico usado pela família com a chave, como “RSA”, neste exemplo.

Expoente RSA (**`e`**)

O parâmetro `e` contém o valor de expoente da chave pública RSA. Ele é representado como um valor codificado em Base64URLUInt.

Modulus (**`n`**) RSA

O parâmetro `n` contém o valor de modulus da chave pública RSA. Ele é representado como um valor codificado em Base64URLUInt.

Usar o **`use`**

O parâmetro `use` descreve o uso pretendido da chave pública. Neste exemplo, o `use` valor `sig` representa assinatura.

b. Pesquise a chave web JSON pública de um `kid` que corresponda ao `kid` do JWT.

Verificar as declarações

Para verificar alegações JWT

1. Usando um dos métodos a seguir, verifique se o token não expirou.

   a. Decodifique o token e compare a reivindicação `exp` com a hora atual.

   b. Se seu token de acesso incluir uma `aws.cognito.signin.user.admin` reivindicação, envie uma solicitação para uma API como GetUser. As solicitações de API que você autoriza com um token de acesso retornarão um erro se o token tiver expirado.

   c. Apresente seu token de acesso em uma solicitação ao endpoint userinfo. A solicitação retornará um erro se o token tiver expirado.

2. A declaração `aud` em um token de ID e a declaração `client_id` em um token de acesso devem corresponder ao ID do cliente de aplicação criado no grupo de usuários do Amazon Cognito.
3. A solicitação de emissor (`iss`) deve corresponder ao seu grupo de usuários. Por exemplo, um grupo de usuários criado na região `us-east-1` terá o seguinte valor de `iss`:

   `https://cognito-idp.us-east-1.amazonaws.com/<userpoolID>`.
4. Verifique a alegação `token_use`.

   - Se você estiver aceitando apenas o token de acesso nas APIs da Web, o valor do token terá de ser `access`.
   - Se estiver usando apenas o token de ID, o valor dele precisa ser `id`.
   - Se estiver usando os tokens de ID e acesso, a alegação `token_use` deverá ser `id` ou `access`.

Você já pode aceitar as alegações dentro do token.

Gerenciar a expiração e o armazenamento em cache do token do grupo de usuários

Sua aplicação deve concluir com êxito uma das solicitações a seguir sempre que você quiser obter um novo token web JSON (JWT).

- Solicite as credenciais do cliente ou a concessão do código de autorização do Endpoint de token.
- Solicite uma concessão implícita de suas páginas de login gerenciado.
- Autentique um usuário local em uma solicitação da API do Amazon Cognito, como. InitiateAuth

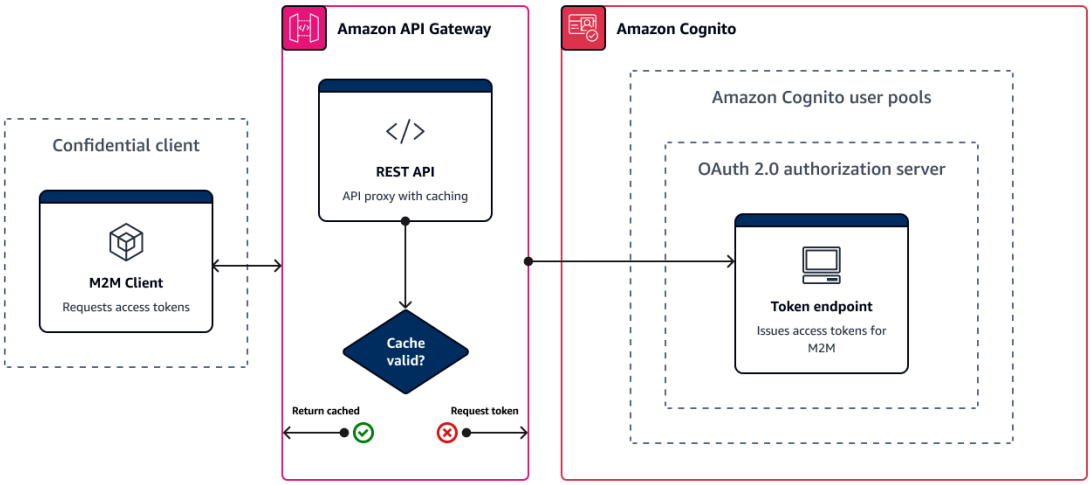
Você pode configurar o grupo de usuários para definir que os tokens expirem em minutos, horas ou dias. Para garantir a performance e a disponibilidade da aplicação, use os tokens do Amazon Cognito por cerca 75% da vida útil do token, e só então recupere novos tokens. Uma solução de cache que você cria para a aplicação mantém os tokens disponíveis e evita a rejeição de solicitações do Amazon Cognito quando a taxa de solicitação é muito alta. Uma aplicação do lado do cliente deve armazenar tokens em um cache de memória. Uma aplicação do lado do servidor pode adicionar um mecanismo de cache criptografado para armazenar tokens.

Quando seu grupo de usuários gera um grande volume de usuários ou machine-to-machine atividades, você pode encontrar os limites que o Amazon Cognito define para o número de solicitações de tokens que você pode fazer. Para reduzir o número de solicitações realizadas aos endpoints do Amazon Cognito, você pode armazenar e reutilizar dados de autenticação com segurança ou implementar recuos exponenciais e novas tentativas.

Os dados de autenticação originam-se de duas classes de endpoints. Os endpoints do Amazon Cognito OAuth 2.0 incluem o endpoint de token, que atende às credenciais do cliente e às solicitações gerenciadas de código de autorização de login. Os endpoints de serviço respondem a solicitações da API de grupos de usuários, como `InitiateAuth` e `RespondToAuthChallenge`. Cada tipo de solicitação tem seu próprio limite. Para obter mais informações sobre limites, consulte Cotas no Amazon Cognito.

Armazenamento em cache de tokens de machine-to-machine acesso com o Amazon API Gateway

Com o armazenamento em cache de tokens do API Gateway, seu aplicativo pode ser escalado em resposta a eventos maiores do que a cota padrão da taxa de solicitação dos endpoints do Amazon OAuth Cognito.



É possível armazenar os tokens de acesso em cache para que a aplicação solicite apenas um novo token de acesso se o token armazenado em cache expirar. Do contrário, o armazenamento do endpoint em cache retornará um token do cache. Isso evita uma chamada adicional para um endpoint da API do Amazon Cognito. Quando você usa o Amazon API Gateway como um proxy para o Endpoint de token, a API responde à maioria das solicitações que, de outra forma, contribuiriam

para sua cota de solicitações, evitando solicitações malsucedidas em decorrência da limitação da taxa.

A solução baseada no API Gateway a seguir oferece uma implementação de cache de tokens de baixa latência e pouco uso de código/nenhum código. O API Gateway APIs é criptografado em trânsito e, opcionalmente, em repouso. Um cache do API Gateway é ideal para a concessão de credenciais de cliente OAuth 2.0, um tipo de concessão frequentemente de alto volume que produz tokens de acesso para autorizar machine-to-machine e sessões de microsserviço. Em um evento como um aumento de tráfego que faz com que seus microsserviços sejam escalados horizontalmente, você pode acabar com muitos sistemas usando as mesmas credenciais de cliente em um volume que excede o limite de AWS taxa de solicitação do seu grupo de usuários ou cliente de aplicativo. Para preservar a disponibilidade e a baixa latência da aplicação, uma solução de armazenamento em cache é a prática recomendada nesses cenários.

Nessa solução, você define um cache na sua API para armazenar um token de acesso separado para cada combinação de OAuth escopos e cliente de aplicativo que você deseja solicitar em seu aplicativo. Quando a aplicação faz uma solicitação correspondente à chave de cache, a API responde com um token de acesso que o Amazon Cognito emitiu para a primeira solicitação correspondente à chave de cache. Quando a duração da chave de cache expira, a API encaminha a solicitação ao endpoint do token e armazena em cache um novo token de acesso.

> **Note**
>
> A duração da chave de cache deve ser menor do que a duração do token de acesso do cliente da aplicação.

A chave de cache é uma combinação dos OAuth escopos que você solicita no `scope` parâmetro no corpo da solicitação e no `Authorization` cabeçalho da solicitação. O cabeçalho `Authorization` contém o ID do cliente da aplicação e o respectivo segredo. Você não precisa implementar lógica adicional na aplicação para implementar essa solução. Você só deve atualizar sua configuração para alterar o caminho para o endpoint do token do grupo de usuários.

Você também pode implementar o armazenamento em cache de tokens com ElastiCache (Redis OSS). Para um controle detalhado com políticas do AWS Identity and Access Management (IAM), considere um cache do Amazon DynamoDB.

> **Note**
>
> O armazenamento em cache no API Gateway está sujeito a um custo adicional. Para obter mais detalhes, consulte a definição de preço.

Como configurar um proxy de armazenamento em cache com o API Gateway

1. Abra o console do API Gateway e crie uma API REST.
2. Em Resources (Recursos), crie um método POST.
   a. Selecione o integration type (tipo de integração) HTTP.
   b. Selecione Use HTTP proxy integration (Usar integração de proxy HTTP).
   c. Digite um Endpoint URL (URL de endpoint) do `https://<your user pool domain>/oauth2/token`.
3. Em Resources (Recursos), configure a chave de cache.
   a. Edite a Method request (Solicitação de método) do método POST.

      > **Note**
      >
      > Essa validação de solicitação de método é para uso com a autorização `client_secret_basic` em solicitações de token, onde o segredo do cliente é codificado no cabeçalho da solicitação `Authorization`. Para validar o corpo da solicitação JSON na autorização `client_secret_post`, crie um modelo de dados que exija que client_secret esteja presente. Nesse modelo, seu Validador de solicitação deve validar o corpo, os parâmetros da string de consulta e os cabeçalhos.

   b. Configure o método Validador de solicitação para Validar parâmetros de string de consulta e cabeçalhos. Para obter mais informações sobre validação de solicitações, consulte Solicitar validação no Guia do desenvolvedor do Amazon API Gateway.
   c. Defina o parâmetro `scope` e o cabeçalho `Authorization` como sua chave de armazenamento em cache.

i. Adicione uma string de consulta a Parâmetros de string de consulta de URL. Insira `scope` no Nome da string de consulta e selecione Obrigatório e Armazenamento em cache.

ii. Adicione um cabeçalho a Cabeçalhos de solicitação HTTP. Insira `Authorization` no Nome do cabeçalho de solicitação e selecione Obrigatório e Armazenamento em cache.

4. Em Stages (Estágios), configure o armazenamento em cache.

   a. Escolha o estágio que deseja modificar e clique em Editar em Detalhes do estágio.

   b. Em Configurações adicionais, Configurações de cache, ative a opção Provisionar cache de APIs.

   c. Selecione uma Cache capacity (Capacidade de cache). Uma maior capacidade de cache melhora o desempenho, mas tem um custo adicional.

   d. Desmarque a caixa de seleção Exigir autorização. Selecione Continuar.

   e. O API Gateway aplica políticas de cache somente aos métodos GET do nível do estágio. É necessário aplicar uma substituição de política de cache ao método POST.

      Expanda o estágio configurado e selecione o método `POST`. Para criar configurações de cache para o método, selecione Criar substituição.

   f. Ative a opção Ativar cache de método.

   g. Insira um cache time-to-live (TTL) de 3600 segundos. Escolha Salvar.

5. Em Stages (Estágios), anote o Invoke URL (URL de invocação).

6. Atualize a aplicação para solicitações de token POST para o Invoke URL (URL de invocação) de sua API em vez do endpoint `/oauth2/token` do grupo de usuários.

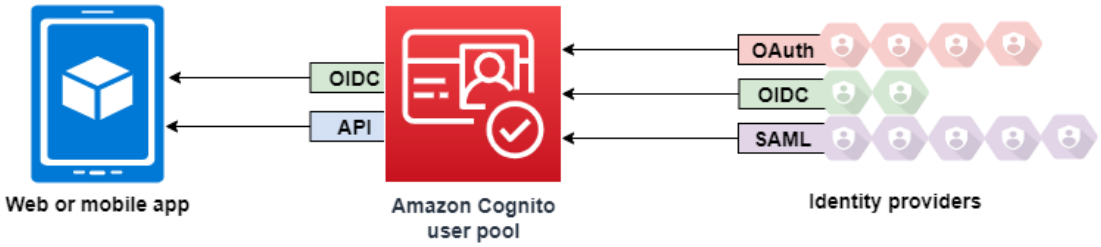
Como acessar recursos após o login bem-sucedido

Os usuários da aplicação podem fazer login diretamente por meio de um grupo de usuários ou federar por meio de um provedor de identidades (IdP) de terceiros. O grupo de usuários gerencia a sobrecarga de lidar com os tokens que são retornados do login social por meio do Facebook, Google, Amazon e Apple, e do OpenID Connect (OIDC) e SAML. IdPs Para obter mais informações, consulte Compreendendo os tokens web JSON do grupo de usuários () JWTs.

Depois de uma autenticação bem-sucedida no grupo de usuários, sua aplicação receberá tokens do grupo de usuários do Amazon Cognito. Você pode usar tokens do grupo de usuários para:

- Recupere AWS credenciais que autorizam solicitações de recursos de aplicativos, como Amazon Serviços da AWS DynamoDB e Amazon S3.
- Fornecer um comprovante de autenticação temporário e revogável.
- Preencher dados de identidade em um perfil de usuário na aplicação.
- Autorizar alterações no perfil do usuário conectado no diretório do grupo de usuários.
- Autorizar solicitações de informações do usuário com um token de acesso.
- Autorize solicitações de dados que estão por trás do acesso externo protegido APIs com tokens de acesso.
- Autorizar o acesso aos ativos da aplicação que estão armazenados no cliente ou no servidor com o Amazon Verified Permissions.

Para obter mais informações, consulte Um exemplo de sessão de autenticação e Compreendendo os tokens web JSON do grupo de usuários () JWTs.



Tópicos

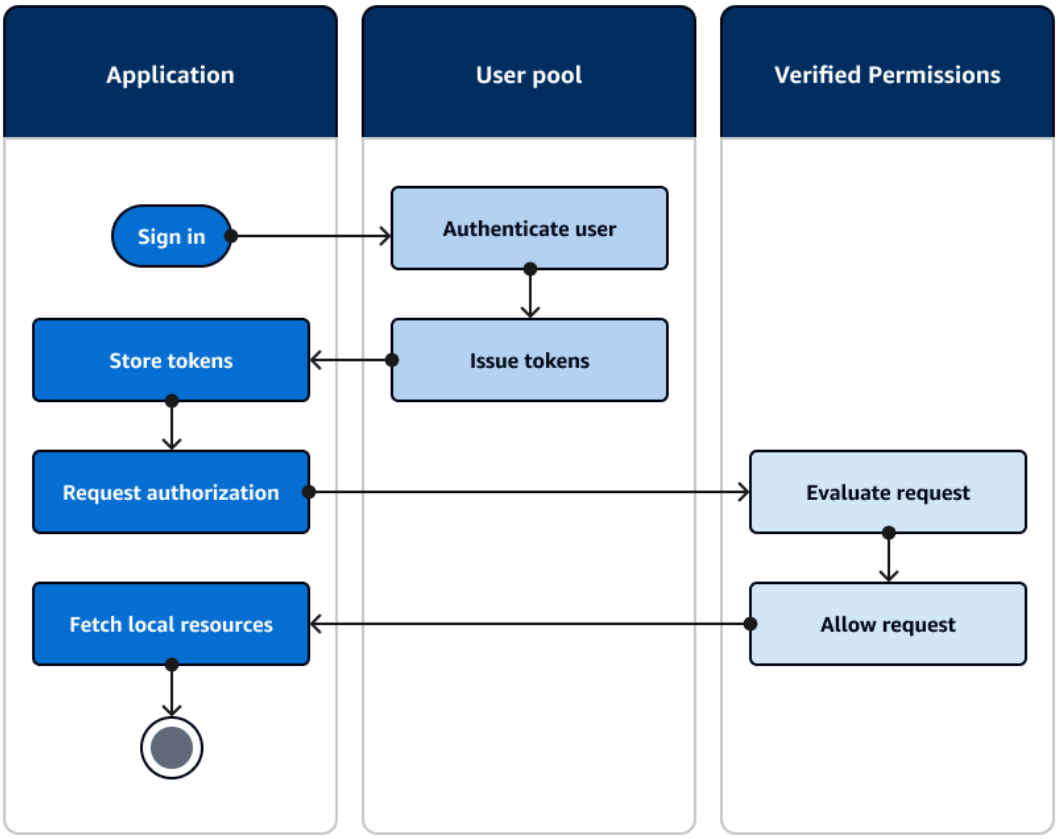
- Autorizar o acesso aos recursos do cliente ou do servidor com o Amazon Verified Permissions
- Como acessar recursos com o API Gateway após o acesso
- Acessando Serviços da AWS usando um pool de identidades após o login

Autorizar o acesso aos recursos do cliente ou do servidor com o Amazon Verified Permissions

Sua aplicação pode passar os tokens de um usuário conectado para o Amazon Verified Permissions. O Verified Permissions é um serviço de autorização e gerenciamento de permissões escaláveis e refinadas para aplicações criadas por você. Um grupo de usuários do Amazon Cognito pode ser uma fonte de identidade para um armazenamento de políticas do Verified Permissions. O Verified

Permissions toma decisões de autorização para ações e recursos solicitados, como `GetPhoto` para `premium_badge.png`, da entidade principal e seus atributos nos tokens do grupo de usuários.

O diagrama a seguir mostra como sua aplicação pode passar o token de um usuário para o Verified Permissions em uma solicitação de autorização.



Comece a usar o Amazon Verified Permissions

Depois de integrar seu grupo de usuários com Verified Permissions, você obtém uma fonte central de autorização granular para todas as aplicações do Amazon Cognito. Isso elimina a necessidade de uma lógica de segurança refinada que, de outra forma, você teria que codificar e replicar entre todas as aplicações. Para obter mais informações sobre autorização com o Verified Permissions, consulte Autorização com o Amazon Verified Permissions.

As solicitações de autorização de permissões verificadas exigem AWS credenciais. Você pode implementar algumas das técnicas a seguir para aplicar com segurança as credenciais às solicitações de autorização.

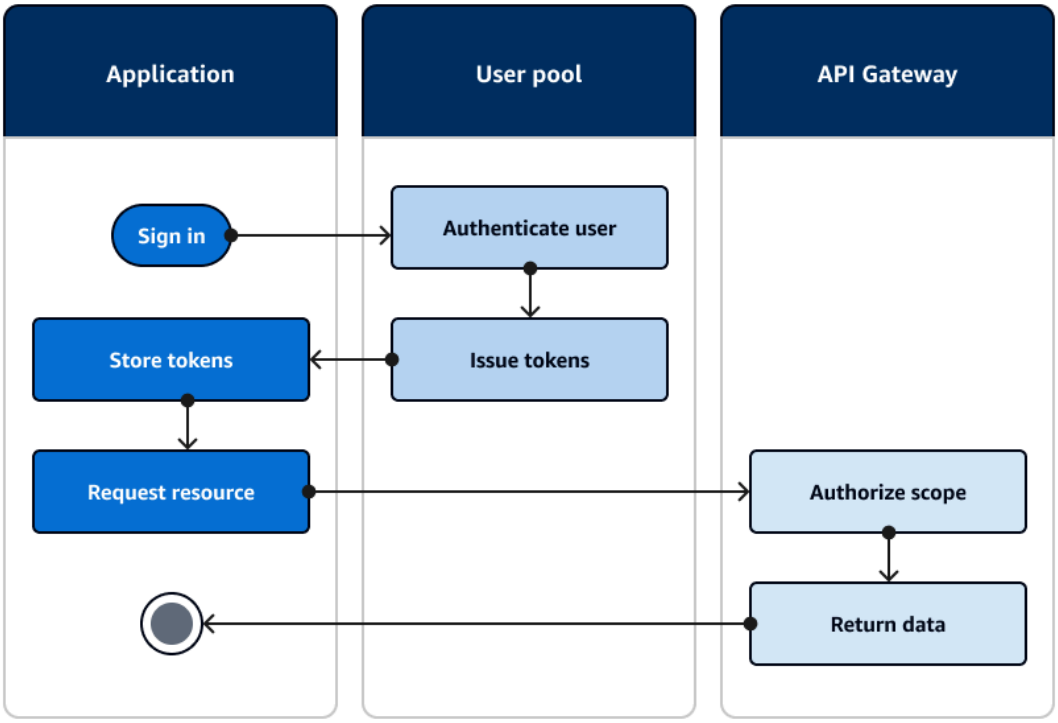
- Opere uma aplicação web que possa armazenar segredos no backend do servidor.

- Adquira credenciais autenticadas do banco de identidades.
- Proxy as solicitações do usuário por meio de uma access-token-authorized API e anexe AWS as credenciais à solicitação.

Como acessar recursos com o API Gateway após o acesso

Um uso comum dos tokens de grupos de usuários do Amazon Cognito é autorizar solicitações para uma API REST do API Gateway. Os escopos OAuth 2.0 nos tokens de acesso podem autorizar um método e um caminho, como `HTTP GET` for. `/app_assets` Os tokens de ID podem servir como autenticação genérica para uma API e transmitir atributos do usuário para o serviço de backend. O API Gateway tem opções de autorização personalizadas adicionais, como autorizadores JWT para autorizadores HTTP e APIs Lambda, que podem aplicar uma lógica mais refinada.

O diagrama a seguir ilustra um aplicativo que está obtendo acesso a uma API REST com os escopos OAuth 2.0 em um token de acesso.



Sua aplicação deve coletar os tokens das sessões autenticadas e adicioná-los como tokens portadores a um cabeçalho `Authorization` na solicitação. Configure o autorizador que você configurou para a API, o caminho e o método para avaliar o conteúdo do token. O API Gateway

retorna dados somente se a solicitação corresponder às condições que você configurou para seu autorizador.

Algumas maneiras possíveis pelas quais a API do API Gateway pode aprovar o acesso a partir de uma aplicação são:

- O token de acesso é válido, não expirou e contém o escopo OAuth 2.0 correto. O autorizador de grupos de usuários do Amazon Cognito para uma API REST é uma implementação comum com pouca barreira de entrada. Você também pode avaliar o corpo, os parâmetros da string de consulta e os cabeçalhos de uma solicitação para esse tipo de autorizador.
- O token de ID é válido e não expirou. Ao passar um token de ID para um autorizador do Amazon Cognito, você pode fazer uma validação adicional do conteúdo do token de ID no seu servidor de aplicações.
- Um grupo, reivindicação, atributo ou função em um token de acesso ou ID atende aos requisitos que você define em uma função do Lambda. Um autorizador do Lambda analisa o token no cabeçalho da solicitação e o avalia para uma decisão de autorização. Você pode criar uma lógica personalizada em sua função ou fazer uma solicitação de API para o Amazon Verified Permissions.

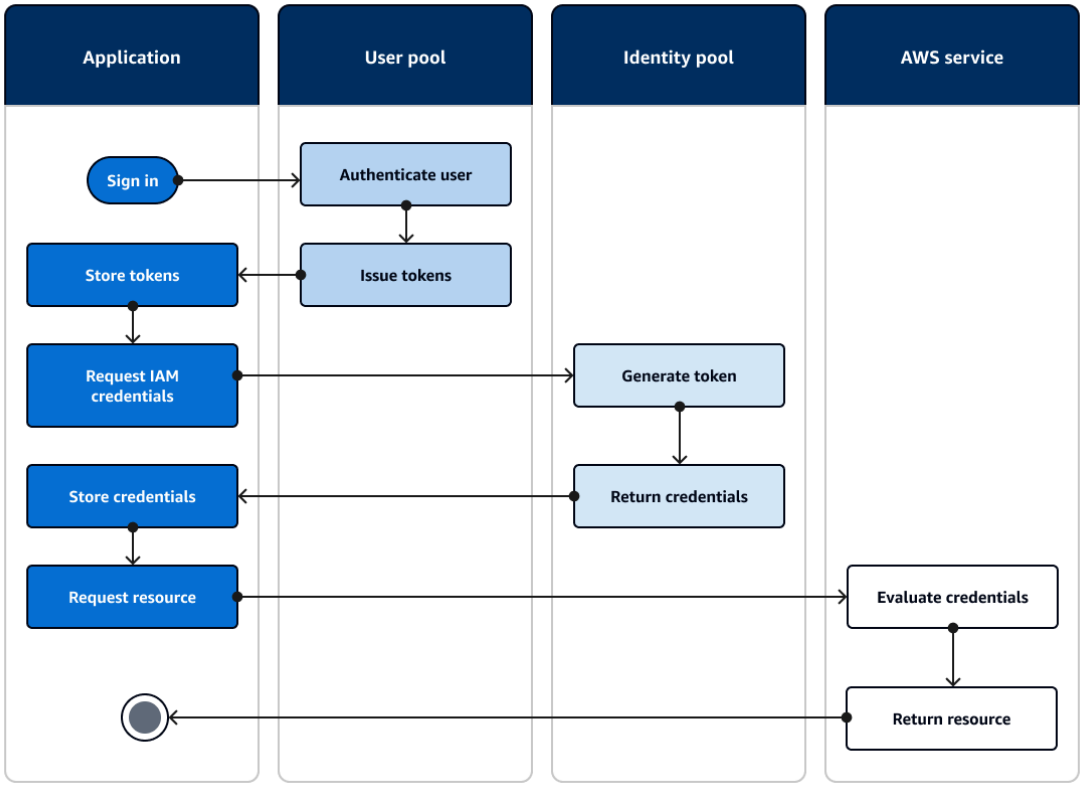
Você também pode autorizar solicitações para uma API GraphQL do AWS AppSync com tokens de um grupo de usuários.

Acessando Serviços da AWS usando um pool de identidades após o login

Depois que seus usuários entrarem com um grupo de usuários, eles poderão acessar Serviços da AWS com credenciais de API temporárias emitidas de um grupo de identidades.

Sua aplicação web ou móvel recebe tokens de um grupo de usuários. Quando você configura seu grupo de usuários como um provedor de identidade para seu grupo de identidades, o grupo de identidades troca tokens por AWS credenciais temporárias. Essas credenciais podem ser definidas de acordo com as funções do IAM e suas políticas, que dão aos usuários acesso a um conjunto limitado de AWS recursos. Para obter mais informações, consulte Fluxo de autenticação dos bancos de identidades.

O diagrama a seguir mostra como uma aplicação faz login com um grupo de usuários, recupera as credenciais do banco de identidades e solicita um ativo de um AWS service (Serviço da AWS).



Você pode usar as credenciais do banco de identidades para:

- Fazer solicitações de autorização detalhadas para o Amazon Verified Permissions com as próprias credenciais do seu usuário.
- Conecte-se a uma API REST do Amazon API Gateway ou a uma API AWS AppSync GraphQL que autorize conexões com o IAM.
- Conectar-se a um backend de banco de dados, como Amazon DynamoDB ou Amazon RDS, que autoriza conexões com o IAM.
- Recuperar ativos da aplicação de um bucket do Amazon S3.
- Inicie uma sessão com um desktop WorkSpaces virtual da Amazon.

Os bancos de identidades não operam exclusivamente em uma sessão autenticada com um grupo de usuários. Eles também aceitam autenticação diretamente de provedores de identidades de terceiros e podem gerar credenciais para usuários convidados não autenticados.

Para obter mais informações sobre o uso de grupos de identidades junto com grupos de grupos de usuários para controlar o acesso aos seus AWS recursos, consulte Como adicionar grupos a um grupo de usuários Controle de acesso com base em perfil e. Além disso, para obter mais

informações sobre grupos de identidades e AWS Identity and Access Management, consulteFluxo de autenticação dos bancos de identidades.

Configurando um grupo de usuários com o Console de gerenciamento da AWS

Crie um grupo de usuários do Amazon Cognito e anote o ID de grupos de usuários e o ID do cliente da aplicação de cada uma das suas aplicações clientes. Para obter mais informações sobre como criar grupos de usuários, consulte Conceitos básicos dos grupos de usuários.

Configurando um pool de identidades com o Console de gerenciamento da AWS

O procedimento a seguir descreve como usar o Console de gerenciamento da AWS para integrar um grupo de identidades a um ou mais grupos de usuários e aplicativos clientes.

Como adicionar um provedor de identidades (IdP) de grupos de usuários do Amazon Cognito

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Selecione Adicionar provedor de identidade.
4. Selecione Grupo de usuários do Amazon Cognito.
5. Insira um ID de grupo de usuários e um ID de cliente de aplicativo.
6. Para alterar o perfil que o Amazon Cognito solicita ao emitir credenciais para usuários que se autenticaram com esse provedor, defina Configurações de perfil.

    a. Você pode conceder aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras. Com um IdP de grupo de usuários do Amazon Cognito, você também pode Escolher perfil com a reivindicação preferred_role em tokens. Para ter mais informações sobre a declaração `cognito:preferred_role`, consulte Como atribuir valores de precedência a grupos.

        i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual você deseja comparar a reivindicação à regra, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que você deseja atribuir quando houver correspondência com a Atribuição de perfil. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

ii. Se você selecionar Escolher perfil com a reivindicação preferred_role em tokens, o Amazon Cognito emitirá credenciais para o perfil na reivindicação do usuário `cognito:preferred_role`. Se nenhuma reivindicação de perfil preferencial estiver presente, o Amazon Cognito emitirá credenciais com base na Resolução de função.

b. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.

7. Para alterar as tags de identidade principal que o Amazon Cognito atribui ao emitir credenciais para usuários que se autenticaram com esse provedor, configure Atributos para controle de acesso.

   - Para não aplicar nenhuma tag de entidade principal, selecione Inativo.
   - Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.
   - Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.

8. Selecione Salvar alterações.

Como integrar um grupo de usuários com um grupo de identidades

Depois que o usuário do aplicativo for autenticado, adicione o token de identidade desse usuário ao mapa de logins no provedor de credenciais. O nome do provedor dependerá do ID do grupo de usuários do Amazon Cognito. Ele terá a seguinte estrutura:

```
cognito-idp.<region>.amazonaws.com/<YOUR_USER_POOL_ID>
```

Você pode derivar o valor de a *<region>* partir da ID do grupo de usuários. Por exemplo, se o ID do grupo de usuários for`us-east-1_EXAMPLE1`, então *<region>* é`us-east-1`. Se o ID do grupo de usuários for`us-west-2_EXAMPLE2`, então *<region>* é`us-west-2`.

JavaScript

```
var cognitoUser = userPool.getCurrentUser();

if (cognitoUser != null) {
 cognitoUser.getSession(function(err, result) {
  if (result) {
```

```
   console.log('You are now logged in.');

   // Add the User's Id Token to the Cognito credentials login map.
   AWS.config.credentials = new AWS.CognitoIdentityCredentials({
    IdentityPoolId: 'YOUR_IDENTITY_POOL_ID',
    Logins: {
     'cognito-idp.<region>.amazonaws.com/<YOUR_USER_POOL_ID>':
 result.getIdToken().getJwtToken()
    }
   });
  }
 });
}
```

Android

```
cognitoUser.getSessionInBackground(new AuthenticationHandler() {
 @Override
 public void onSuccess(CognitoUserSession session) {
  String idToken = session.getIdToken().getJWTToken();

  Map<String, String> logins = new HashMap<String, String>();
  logins.put("cognito-idp.<region>.amazonaws.com/<YOUR_USER_POOL_ID>",
 session.getIdToken().getJWTToken());
  credentialsProvider.setLogins(logins);
 }

});
```

iOS - objective-C

```
AWSServiceConfiguration *serviceConfiguration = [[AWSServiceConfiguration alloc]
 initWithRegion:AWSRegionUSEast1 credentialsProvider:nil];
AWSCognitoIdentityUserPoolConfiguration *userPoolConfiguration =
 [[AWSCognitoIdentityUserPoolConfiguration alloc] initWithClientId:@"YOUR_CLIENT_ID"
  clientSecret:@"YOUR_CLIENT_SECRET" poolId:@"YOUR_USER_POOL_ID"];
[AWSCognitoIdentityUserPool
 registerCognitoIdentityUserPoolWithConfiguration:serviceConfiguration
 userPoolConfiguration:userPoolConfiguration forKey:@"UserPool"];
AWSCognitoIdentityUserPool *pool = [AWSCognitoIdentityUserPool
 CognitoIdentityUserPoolForKey:@"UserPool"];
```

```
AWSCognitoCredentialsProvider *credentialsProvider = [[AWSCognitoCredentialsProvider
 alloc] initWithRegionType:AWSRegionUSEast1 identityPoolId:@"YOUR_IDENTITY_POOL_ID"
 identityProviderManager:pool];
```

iOS - swift

```
let serviceConfiguration = AWSServiceConfiguration(region: .USEast1,
 credentialsProvider: nil)
let userPoolConfiguration = AWSCognitoIdentityUserPoolConfiguration(clientId:
 "YOUR_CLIENT_ID", clientSecret: "YOUR_CLIENT_SECRET", poolId: "YOUR_USER_POOL_ID")
AWSCognitoIdentityUserPool.registerCognitoIdentityUserPoolWithConfiguration(serviceConfigura
 userPoolConfiguration: userPoolConfiguration, forKey: "UserPool")
let pool = AWSCognitoIdentityUserPool(forKey: "UserPool")
let credentialsProvider = AWSCognitoCredentialsProvider(regionType: .USEast1,
 identityPoolId: "YOUR_IDENTITY_POOL_ID", identityProviderManager:pool)
```

Escopos, M2M e servidores de recursos

Depois de configurar um domínio para seu grupo de usuários, o Amazon Cognito provisiona automaticamente um servidor de autorização OAuth 2.0 e uma interface de usuário web hospedada com páginas de cadastro e login que seu aplicativo pode apresentar aos seus usuários. Para obter mais informações, consulte Login gerenciado do grupo de usuários. Você pode escolher os escopos que deseja que o servidor de autorização adicione aos tokens de acesso. Os escopos autorizam o acesso aos servidores de recursos e aos dados de usuário.

Um servidor de recursos é um servidor de API OAuth 2.0. Para proteger recursos protegidos por acesso, ele valida se os tokens de acesso do grupo de usuários contêm os escopos que autorizam o método e o caminho solicitados na API que ele protege. Ele confirma o emissor, com base na assinatura do token, a validade, com base no tempo de expiração do token, e o nível de acesso, com base nos escopos das solicitações de token. Os escopos do grupo de usuários estão na reivindicação `scope` do token de acesso. Para obter mais informações sobre as reivindicações nos tokens de acesso do Amazon Cognito, consulte Como entender o token de acesso.

Com o Amazon Cognito, os escopos nos tokens de acesso podem autorizar o acesso a atributos externos APIs ou de usuário. Você pode emitir tokens de acesso para usuários locais, usuários federados ou identidades de máquinas.

Tópicos

- Autorização da API

- Machine-to-machine Autorização (M2M)
- Sobre escopos
- Sobre servidores de recursos
- Vinculação de recursos

Autorização da API

A seguir estão algumas das maneiras pelas quais você pode autorizar solicitações APIs com tokens do Amazon Cognito:

Token de acesso

Ao adicionar um autorizador do Amazon Cognito a uma configuração de solicitação de método da API REST, adicione escopos de autorização à configuração do autorizador. Com essa configuração, sua API aceita tokens de acesso no cabeçalho `Authorization` e analisa os escopos aceitos neles.

Token de ID

Quando você passa um token de ID válido para um autorizador do Amazon Cognito em sua API REST, o API Gateway aceita a solicitação e passa o conteúdo do token de ID para o backend da API.

Amazon Verified Permissions

No Verified Permissions, você tem a opção de criar um repositório de políticas vinculado à API. O Verified Permissions cria e atribui um autorizador Lambda que processa tokens de ID ou de acesso do cabeçalho `Authorization` da sua solicitação. Esse autorizador Lambda passa seu token para o repositório de políticas, onde o Verified Permissions o compara com as políticas e retorna uma decisão de permissão ou negação ao autorizador.

Mais atributos

- Controlar e gerenciar acesso a uma API REST no API Gateway
- Autorização com o Amazon Verified Permissions

Machine-to-machine Autorização (M2M)

O Amazon Cognito aceita aplicações que acessam dados de API com identidades de máquinas. As identidades de máquinas em grupos de usuários são clientes confidenciais que são executados em servidores de aplicativos e se conectam remotamente APIs. Sua operação acontece sem a interação do usuário: tarefas agendadas, fluxos de dados ou atualizações de ativos. Quando esses clientes autorizam suas solicitações com um token de acesso, eles realizam a autorização máquina a máquina, ou M2M. Na autorização M2M, um segredo compartilhado substitui as credenciais do usuário no controle de acesso.

Uma aplicação que acessa uma API com autorização M2M deve ter um ID do cliente e uma chave secreta do cliente. Em seu grupo de usuários, você deve criar um cliente de aplicação que permita a concessão de credenciais de clientes. Para permitir credenciais de cliente, o cliente de aplicação deve ter um segredo, enquanto você deve ter um domínio de grupo de usuários. Nesse fluxo, a identidade da sua máquina solicita um token de acesso diretamente do Endpoint de token. Você pode autorizar somente escopos personalizados de servidores de recursos em tokens de acesso para concessões de credenciais de clientes. Para obter mais informações sobre a configuração de clientes de aplicação, consulte Configurações específicas da aplicação com clientes de aplicação.

O token de acesso de uma concessão de credenciais do cliente é uma declaração verificável das operações que você deseja permitir que a identidade da sua máquina solicite de uma API. Para saber mais sobre como os tokens de acesso autorizam solicitações de API, leia a seguir. Para ver um exemplo de aplicação, consulte Autorização máquina a máquina baseada no Amazon Cognito e no API Gateway usando o CDK da AWS.

A autorização M2M tem um modelo de cobrança que difere da forma como os usuários ativos mensais (MAUs) são cobrados. Quando a autenticação do usuário tem um custo por usuário ativo, a cobrança de M2M reflete as credenciais ativas do cliente, os clientes da aplicação e o volume total de solicitações de tokens. Para mais informações, consulte Preço do Amazon Cognito. Para controlar os custos da autorização M2M, otimize a duração dos tokens de acesso e o número de solicitações de token que as aplicações fazem. Consulte Gerenciar a expiração e o armazenamento em cache do token do grupo de usuários para saber como usar o cache do API Gateway para reduzir as solicitações de novos tokens na autorização M2M.

Para obter informações sobre como otimizar as operações do Amazon Cognito que adicionam custos à AWS sua fatura, consulte. Gerenciar custos

Metadados do cliente para credenciais do cliente machine-to-machine (M2M)

Você pode transmitir metadados do cliente em solicitações de M2M. Os metadados do cliente são informações adicionais de um usuário ou ambiente de aplicação que podem contribuir para os resultados de um Acionador do Lambda antes da geração do token. Nas operações de autenticação com um usuário principal, você pode passar os metadados do cliente para o gatilho de pré-geração do token no corpo das solicitações AdminRespondToAuthChallengede RespondToAuthChallengeAPI. Como as aplicações conduzem o fluxo de geração de tokens de acesso para M2M com solicitações diretas ao Endpoint de token, elas têm um modelo diferente. No corpo POST das solicitações de token para credenciais do cliente, transmita um parâmetro `aws_client_metadata` com o objeto de metadados do cliente codificado em URL (`x-www-form-urlencoded`) para string. Para ver um exemplo de solicitação, consulte Credenciais do cliente com autorização básica. Veja a seguir um exemplo de parâmetro que transmite os pares de chave-valor `{"environment": "dev", "language": "en-US"}`.

```
aws_client_metadata=%7B%22environment%22%3A%20%22dev%22,%20%22language%22%3A%20%22en-US%22%7D
```

Sobre escopos

Um escopo é um nível de acesso que um aplicativo pode solicitar para um recurso. Em um token de acesso do Amazon Cognito, o escopo é respaldado pela confiança que você configura com o grupo de usuários: um emissor confiável de tokens de acesso com uma assinatura digital conhecida. Os grupos de usuários podem gerar tokens de acesso com escopos que provam que o cliente tem permissão para gerenciar parte ou a totalidade de seu próprio perfil de usuário ou recuperar dados de uma API de back-end. Os grupos de usuários do Amazon Cognito emitem tokens de acesso com o escopo reservado da API dos grupos de usuários, escopos personalizados e escopos do OpenID Connect (OIDC).

O escopo reservado da API do grupo de usuários

O escopo do `aws.cognito.signin.user.admin` autoriza operações de autoatendimento para o usuário atual na API de grupos de usuários do Amazon Cognito. Ele autoriza o portador de um token de acesso a consultar e atualizar todas as informações sobre o portador com, por exemplo, as operações da API GetUser. UpdateUserAttributes Quando você autentica o usuário com a API de grupos de usuários do Amazon Cognito, esse é o único escopo que você recebe no token de acesso. Também é o único escopo necessário para ler e gravar atributos de usuário que você autorizou o cliente da aplicação a ler e gravar. Também é possível solicitar esse escopo em solicitações ao Autorizar endpoint. Esse escopo por si só não é suficiente para solicitar atributos de usuário do endpoint userinfo. Para tokens de acesso que autorizam a API de grupos de usuários

e solicitações `userInfo` para os usuários, é necessário solicitar os dois escopos `openid` e `aws.cognito.signin.user.admin` em uma solicitação `/oauth2/authorize`.

Escopos personalizados

Os escopos personalizados autorizam solicitações externas APIs que os servidores de recursos protegem. Você pode solicitar escopos personalizados com outros tipos de escopos. É possível encontrar mais informações sobre escopos personalizados em toda esta página.

Escopos do OpenID Connect (OIDC)

Ao autenticar usuários com o servidor de autorização do grupo de usuários, inclusive com login gerenciado, você deve solicitar escopos. É possível autenticar usuários locais do grupo de usuários e usuários federados de terceiros no servidor de autorização do Amazon Cognito. Os escopos do OIDC autorizam a aplicação a ler as informações de usuário do endpoint userinfo do grupo de usuários. O OAuth modelo, em que você consulta os atributos do usuário a partir do `userInfo` endpoint, pode otimizar seu aplicativo para um grande volume de solicitações de atributos do usuário. O endpoint `userInfo` retorna atributos em um nível de permissão que é determinado pelos escopos no token de acesso. Você pode autorizar seu cliente de aplicação a emitir tokens de acesso com os seguintes escopos padrão do OIDC:

OpenID

Um escopo mínimo para consultas do OpenID Connect (OIDC). Autoriza o token de ID, a reivindicação de identificador exclusivo `sub` e a capacidade de solicitar outros escopos.

> **Note**
>
> Quando você solicita o escopo `openid` e nenhum outro, o token de ID do grupo de usuários e a resposta `userInfo` incluem declarações para todos os atributos do usuário que o cliente da aplicação pode ler. Quando você solicita `openid` e outros escopos OIDC, como `profile`, `email` e `phone`, o conteúdo do token de ID e a resposta userInfo são limitados às restrições dos escopos adicionais.
> Por exemplo, uma solicitação ao Autorizar endpoint com o parâmetro `scope=openid+email` retorna um token de ID com `sub`, `email` e `email_verified`. O token de acesso dessa solicitação exibe os mesmos atributos de endpoint userinfo. Uma solicitação com o parâmetro `scope=openid` exibe todos os atributos legíveis pelo cliente no token de ID e de `userInfo`.

perfil

Autoriza todos os atributos de usuário que o cliente da aplicação pode ler.

email

Autoriza os atributos do usuário `email` e `email_verified`. O Amazon Cognito vai gerar `email_verified` se tiver um valor definido explicitamente.

phone

Autoriza os atributos do usuário `phone_number` e `phone_number_verified`.

Sobre servidores de recursos

Uma API do servidor de recursos pode conceder acesso às informações em um banco de dados ou controlar seus recursos de TI. Um token de acesso do Amazon Cognito pode autorizar o acesso a APIs esse suporte 2.0. OAuth O Amazon API Gateway REST APIs tem suporte integrado para autorização com tokens de acesso do Amazon Cognito. A aplicação transmite o token de acesso na chamada de API para o servidor de recursos. O servidor de recursos inspeciona o token de acesso para determinar se o acesso deve ser concedido.

O Amazon Cognito pode fazer futuras atualizações no esquema dos tokens de acesso do grupo de usuários. Se a aplicação analisar o conteúdo do token de acesso antes de passá-lo para uma API, você deverá criar seu código para aceitar atualizações no esquema.

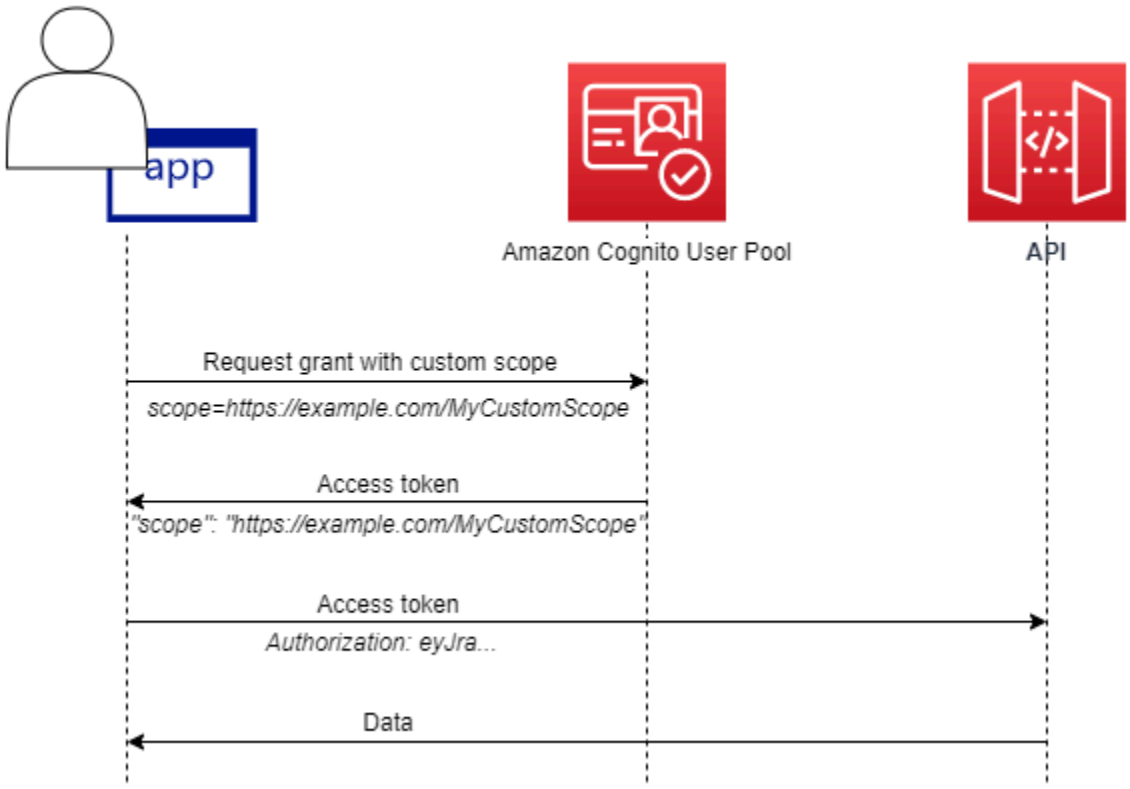
Os escopos personalizados são definidos por você e ampliam os recursos de autorização de um grupo de usuários para incluir propósitos não relacionados à consulta e modificação de usuários e seus atributos. Por exemplo, se você tiver um servidor de recursos para fotos, ele poderá definir dois escopos: `photos.read` para acesso de leitura às fotos e `photos.write` para write/delete acesso. É possível configurar uma API para aceitar tokens de acesso para autorização e conceder solicitações `HTTP GET` para acessar tokens com `photos.read` na reivindicação `scope`, e solicitações `HTTP POST` de tokens com `photos.write`. Estes são escopos personalizados.

> **Note**
>
> O servidor de recursos deve verificar a assinatura e a data de expiração do token de acesso antes de processar quaisquer reivindicações dentro do token. Para obter mais informações sobre como verificar tokens, consulte Como verificar tokens web JSON. Para obter mais informações sobre como verificar e usar tokens de grupos de usuários no Amazon API

Gateway, consulte o blog Integrating Amazon Cognito User Pools with API Gateway. O API Gateway é uma boa opção para inspecionar os tokens de acesso e proteger seus recursos. Para obter mais informações sobre autorizadores do Lambda do API Gateway, consulte Usar os autorizadores do Lambda do API Gateway.

Visão geral do

Com o Amazon Cognito, você pode criar servidores de recursos OAuth 2.0 e associar escopos personalizados a eles. Escopos personalizados em um token de acesso autorizam ações específicas na API. Você pode autorizar qualquer cliente de aplicação no grupo de usuários a emitir escopos personalizados de qualquer um dos servidores de recursos. Associe seus escopos personalizados a um cliente de aplicativo e solicite esses escopos em concessões de código de autorização OAuth 2.0, concessões implícitas e concessões de credenciais de cliente do. Endpoint de token O Amazon Cognito adiciona escopos personalizados na reivindicação `scope` em um token de acesso. Um cliente pode usar o token de acesso em seu servidor de recursos, o que faz com que a decisão de autorização baseada nos escopos esteja presente no token. Para obter mais informações sobre o escopo do token de acesso, consulte Usar tokens com grupos de usuários.



Para obter um token de acesso com escopos personalizados, a aplicação precisa fazer uma solicitação ao Endpoint de token para resgatar um código de autorização ou solicitar uma concessão de credenciais de cliente. No login gerenciado, você também pode solicitar escopos personalizados em um token de acesso por meio de uma concessão implícita.

> **Note**
>
> Porque eles foram projetados para autenticação interativa humana com o grupo de usuários como IdP, InitiateAuthe as AdminInitiateAuthsolicitações só produzem uma `scope` declaração no token de acesso com o valor único. `aws.cognito.signin.user.admin`

Gerenciar o servidor de recursos e os escopos personalizados

Ao criar um servidor de recursos, é necessário fornecer um nome e um identificador do servidor de recursos. Para cada escopo criado no servidor de recursos, é necessário fornecer o nome e a descrição do escopo.

- Nome do servidor de recursos: um nome fácil de lembrar para o servidor de recursos, como `Solar system object tracker` ou `Photo API`.
- Identificador do servidor de recursos: um identificador exclusivo do servidor de recursos. O identificador é qualquer nome que você deseja associar à API, por exemplo `solar-system-data`. É possível configurar identificadores mais longos, por exemplo `https://solar-system-data-api.example.com`, como uma referência mais direta aos caminhos de URI da API, mas strings mais longas aumentam o tamanho dos tokens de acesso.
- Nome do escopo: o valor que você quer nas reivindicações `scope`. Por exemplo, `.sunproximity.read`
- Descrição: uma descrição simples do escopo. Por exemplo, `.Check current proximity to sun`

O Amazon Cognito pode incluir escopos personalizados nos tokens de acesso para qualquer usuário, seja local para o grupo de usuários ou federado com um provedor de identidade de terceiros. Você pode escolher escopos para os tokens de acesso de seus usuários durante os fluxos de autenticação com o servidor de autorização OAuth 2.0 que inclui login gerenciado. A autenticação do usuário deve começar no Autorizar endpoint com `scope` como um dos parâmetros da solicitação. O formato a seguir é recomendado para servidores de recursos. Para um identificador, use um nome fácil para a API. Para um escopo personalizado, use a ação autorizada.

```
resourceServerIdentifier/scopeName
```

Por exemplo, você descobriu um novo asteroide no cinturão de Kuiper e deseja registrá-lo por meio da API `solar-system-data`. O escopo que autoriza operações de gravação no banco de dados de asteroides é `asteroids.add`. Ao solicitar o token de acesso que autorizará você a registrar sua descoberta, formate o parâmetro de solicitação HTTPS `scope` como `scope=solar-system-data/asteroids.add`.

Excluir um escopo de um servidor de recursos não exclui a sua associação com todos os clientes. Em vez disso, o escopo é marcado como inativo. O Amazon Cognito não adiciona escopos inativos aos tokens de acesso, mas continua normalmente caso a aplicação solicite um. Se você adicionar o escopo ao servidor de recursos novamente mais tarde, o Amazon Cognito o gravará novamente no token de acesso. Se você solicitar um escopo que não tenha associado ao cliente de aplicação, independentemente de tê-lo excluído do servidor de recursos do grupo de usuários, a autenticação falhará.

Você pode usar a Console de gerenciamento da AWS API ou a CLI para definir servidores de recursos e escopos para seu grupo de usuários.

Como definir um servidor de recurso para o grupo de usuários (Console de gerenciamento da AWS)

Você pode usar o Console de gerenciamento da AWS para definir um servidor de recursos para seu grupo de usuários.

Para definir um servidor de recursos

1. Faça login no console do Amazon Cognito.
2. No painel de navegação, escolha User Pools (Grupos de usuários) e escolha o grupo de usuários que deseja editar.
3. Clique no menu Domínio em Identidade visual e localize Servidores de recursos.
4. Escolha Create a resource server (Criar um servidor de recursos).
5. Insira um Resource server name (Nome do servidor de recursos). Por exemplo, .`Photo Server`
6. Insira um Resource server identifier (Identificador do servidor de recursos). Por exemplo, .`com.example.photos`

7. Insira os Custom scopes (Escopos personalizados) para seus recursos, como `read` e `write`.
8. Para cada Scope name (Nome de escopo), insira uma Description (Descrição), como `view your photos` e `update your photos`.
9. Escolha Criar.

Seus escopos personalizados podem ser revisados no menu Domínio em Servidores de recursos, na coluna Escopos personalizados. É possível habilitar escopos personalizados para clientes da aplicação no menu Clientes da aplicação em Aplicações. Selecione um cliente da aplicação, localize Páginas de login e clique em Editar. Adicione Custom scopes (Escopos personalizados) e escolha Save changes (Salvar alterações).

Definindo um servidor de recursos para seu grupo de usuários (AWS CLI e AWS API)

Use os comandos a seguir para especificar as configurações do servidor de recursos para o seu grupo de usuários.

Para criar um servidor de recursos

- AWS CLI: `aws cognito-idp create-resource-server`
- AWS API: CreateResourceServer

Para obter informações sobre as configurações do servidor de recursos

- AWS CLI: `aws cognito-idp describe-resource-server`
- AWS API: DescribeResourceServer

Para listar informações sobre todos os servidores de recursos do seu grupo de usuários

- AWS CLI: `aws cognito-idp list-resource-servers`
- AWS API: ListResourceServers

Para excluir um servidor de recursos

- AWS CLI: `aws cognito-idp delete-resource-server`
- AWS API: DeleteResourceServer

Para atualizar as configurações de um servidor de recursos

- AWS CLI: `aws cognito-idp update-resource-server`
- AWS API: UpdateResourceServer

Vinculação de recursos

Com a vinculação de recursos, também conhecida como indicadores de recursos, é possível solicitar concessões específicas da API ao servidor de autorização do grupo de usuários. A vinculação de recursos é uma extensão OAuth 2.0 definida na RFC 8707 que permite que os clientes especifiquem explicitamente qual servidor de recursos eles pretendem acessar durante as solicitações de autorização. Com a vinculação de recursos, as configurações de API podem negar o acesso a tokens que não são especificamente destinados a eles.

> **Note**
>
> Só é possível vincular tokens de acesso a recursos para usuários. Não é possível solicitar a vinculação de recursos com concessões M2M de credenciais de cliente.

Ao usar a vinculação de recursos com grupos de usuários do Amazon Cognito, os clientes podem incluir um parâmetro `resource` em suas solicitações de autenticação para o servidor de autorização do grupo de usuários. Seu grupo de usuários valida que o valor do recurso solicitado é uma URL, seguindo as mesmas regras de esquema do app client callback URLs:`https://`, `http://` with `localhost` only, ou um esquema personalizado, como. `myapp://` O Amazon Cognito define o URI solicitado como o público-alvo na declaração `aud` do token de acesso. Se o recurso solicitado for um servidor de recursos do grupo de usuários, o identificador do servidor de recursos deverá estar no formato de URL. É possível solicitar um recurso por solicitação de autenticação.

Esse recurso é exclusivo para autenticação de login gerenciada com seu servidor de autorização de grupo de usuários OAuth 2.0. Você pode solicitar a vinculação de recursos em concessões implícitas e de código de autorização por meio do Autorizar endpoint. As concessões de atualização de token no Endpoint de token mantêm a declaração `aud` da solicitação original. No momento, isso não está disponível nos modelos de autenticação do SDK.

Implementar a vinculação de recursos com o grupo de usuários do Amazon Cognito

1. Configure um ou mais servidores de recursos no grupo de usuários com identificadores exclusivos.
2. Na solicitação de autorização para `/oauth2/authorize`, solicite um código de autorização ou concessão implícita e inclua o parâmetro `resource`. O valor de `resource` deve ser um identificador de servidor de recursos formatado em URL ou um URL. Por exemplo, .`&resource=https://solar-system-data-api.example.com`
3. O servidor de autorização valida a solicitação do recurso, conclui a autenticação e define a declaração `aud` do token de acesso para o URL do recurso solicitado.
4. Para validar se os tokens foram emitidos especificamente para ele, o recurso que consome o token de acesso do usuário verifica a declaração `aud`.

Configurar recursos do grupo de usuários

Nos capítulos anteriores, você provavelmente configurou alguns recursos com a orientação do console do Amazon Cognito. As páginas desta seção exploram em mais detalhes os requisitos de configuração detalhados de alguns dos principais recursos dos grupos de usuários. Há informações de referência importantes sobre suas opções com clientes da aplicação, configuração de e-mail e SMS, memorização de dispositivos de usuários e muito mais.

Tópicos

- Como atualizar a configuração do grupo de usuários e do cliente da aplicação
- Configurações específicas da aplicação com clientes de aplicação
- Trabalhar com dispositivos de usuários no grupo de usuários
- Como usar o Amazon Pinpoint para análise de grupos de usuários
- Configurações de e-mail para grupos de usuários do Amazon Cognito
- Configurações de mensagens SMS para grupos de usuários do Amazon Cognito

Como atualizar a configuração do grupo de usuários e do cliente da aplicação

Para alterar uma configuração em um grupo de usuários ou cliente da aplicação, você pode aplicar a atualização no console do Amazon Cognito com alguns cliques. Navegue pelas guias baseadas em

recursos nas configurações do grupo de usuários e atualize os campos conforme descrito em outras áreas deste guia.

Muitas organizações gerenciam seus recursos de forma programática em AWS CloudFormation aplicativos desenvolvidos no AWS SDKs ou CDK e em outros softwares de automação. Se esse for seu modelo de gerenciamento de recursos, tome cuidado extra ao fazer alterações nos recursos.

A API opera UpdateUserPoole UpdateUserPoolClientfaz atualizações em um grupo de usuários ou cliente de aplicativo existente. Cada um acompanha um aviso na Referência da API: Se você não fornecer um valor para um atributo, o Amazon Cognito usará seu valor padrão. Quando você envia uma solicitação de atualização com apenas um parâmetro, o Amazon Cognito define esse parâmetro com o valor de sua escolha, depois define todos os outros como um valor padrão. Isso pode redefinir configurações, incluindo seu esquema de atributos, seus acionadores do Lambda e a configuração de e-mail e mensagens SMS.

Além disso, algumas configurações são bloqueadas após a criação do grupo de usuários ou do cliente da aplicação, e você não pode alterá-las a menos que crie um recurso.

Tópicos

- Configurações que não podem ser alteradas
- Configuração de SMS
- Atualização de um grupo de usuários com um AWS SDK ou AWS CDK API REST

Configurações que não podem ser alteradas

Após a criação de um grupo de usuários, não é possível alterar algumas configurações. Se quiser alterar as configurações a seguir, crie um grupo de usuários ou um cliente da aplicação.

> **Note**
>
> Anteriormente, não era possível alterar o nome de um grupo de usuários. Isso mudou. Agora você pode atribuir novos nomes amigáveis aos seus grupos de usuários.

ID do grupo de usuários

Nome do parâmetro da API: Id/ UserPoolId

O ID do grupo de usuários, por exemplo, `us-east-1_EXAMPLE`, é gerado automaticamente pelo Amazon Cognito e não pode ser alterado.

Opções de login do grupo de usuários do Amazon Cognito

Nomes dos parâmetros da API: AliasAttributese UsernameAttributes

Os atributos que seus usuários podem transmitir como nome de usuário ao fazerem login. Ao criar um grupo de usuários, você pode optar por permitir o login com nome de usuário, endereço de e-mail, número de telefone ou nome de usuário preferido. Para alterar as opções de login do grupo de usuários, crie outro grupo de usuários.

Make user name case sensitive (Diferenciar maiúsculas e minúsculas no nome de usuário)

Nome do parâmetro da API: UsernameConfiguration

Quando você cria um nome de usuário que corresponde a outro nome de usuário, exceto pelo uso de maiúsculas/minúsculas, o Amazon Cognito pode tratá-lo como o mesmo usuário ou como usuários únicos. Para obter mais informações, consulte Sensibilidade entre maiúsculas e minúsculas do grupo de usuários. Para alterar a distinção entre maiúsculas e minúsculas, crie outro grupo de usuários.

Segredo do cliente

Nome do parâmetro da API: GenerateSecret

Ao criar um cliente da aplicação, você pode gerar um segredo de cliente para que somente fontes confiáveis possam fazer solicitações ao grupo de usuários. Para obter mais informações, consulte Configurações específicas da aplicação com clientes de aplicação. Para alterar um segredo de cliente, crie outro cliente da aplicação no mesmo grupo de usuários.

Atributos obrigatórios

Nome do parâmetro da API: Schema

Os atributos aos quais seus usuários devem fornecer valores no cadastro ou quando você os cria. Para obter mais informações, consulte Trabalhar com atributos do usuário. Para alterar os atributos necessários, crie outro grupo de usuários.

Atributos personalizados (exclusão)

Nome do parâmetro da API: Schema

Atributos com nomes personalizados. Você pode alterar o valor do atributo personalizado de um usuário, mas não é possível excluir um atributo personalizado do grupo de usuários. Para obter

mais informações, consulte Trabalhar com atributos do usuário. Se você atingir o número máximo de atributos personalizados e quiser modificar a lista, crie outro grupo de usuários.

Configuração de SMS

Depois de ativar mensagens de SMS no grupo de usuários, não é possível desativá-las.

- Se você optar por configurar mensagens de SMS ao criar um grupo de usuários, não conseguirá desativar o SMS depois que concluir a configuração.
- Você pode ativar mensagens de SMS em um grupo de usuários que você criou, mas depois disso, não poderá desativar o SMS.
- O Amazon Cognito pode usar mensagens de SMS para convite e recuperação de contas de usuários, verificação de atributos e autenticação multifator (MFA). Depois de ativar as mensagens de SMS, você pode ativá-las ou desativá-las a qualquer momento para essas funções.
- A configuração de mensagens de SMS inclui um perfil do IAM que você delega ao Amazon Cognito para enviar mensagens com o Amazon SNS. É possível alterar o perfil atribuído a qualquer momento.

Atualização de um grupo de usuários com um AWS SDK ou AWS CDK API REST

No console do Amazon Cognito é possível alterar as configurações do grupo de usuários, um parâmetro por vez. Por exemplo, para adicionar um acionador do Lambda, você escolhe Adicionar acionador do Lambda e escolhe a função e o tipo de acionador. A API de grupos de usuários do Amazon Cognito é estruturada de forma que as operações de atualização para grupos de usuários e clientes de aplicações exijam o conjunto completo de parâmetros para o grupo de usuários. No entanto, o console automatiza de forma transparente essa operação de atualização com suas outras configurações do grupo de usuários.

Às vezes, você pode descobrir que uma alteração em outro lugar Conta da AWS pode fazer com que as atualizações gerem um erro quando não estão relacionadas à configuração que você deseja alterar. Uma identidade excluída do Amazon SES ou uma alteração em uma permissão do IAM para AWS WAF, por exemplo. Se um dos parâmetros atuais não for mais válido, você não poderá atualizar as configurações até corrigi-lo. Ao encontrar esse erro, analise a resposta do erro e valide a configuração mencionada.

A AWS Cloud Development Kit (AWS CDK)API REST de grupos de usuários do Amazon Cognito AWS SDKsé uma ferramenta para automação e configuração programática dos recursos do

Amazon Cognito. As solicitações com essas ferramentas também devem, assim como no console do Amazon Cognito, atualizar uma definição com uma configuração completa de recursos no corpo da solicitação. Em um nível mais alto, é necessário realizar o seguinte processo.

1. Capture o resultado de uma operação que descreve a configuração do recurso existente.
2. Modifique o resultado com a alteração das configurações.
3. Envie a configuração modificada em uma operação que atualiza seu recurso.

O procedimento a seguir atualiza sua configuração com a operação UpdateUserPoolda API. A mesma abordagem, com campos de entrada diferentes, se aplica UpdateUserPoolClienta.

> **Important**
>
> Se você não fornecer valores para os parâmetros existentes, o Amazon Cognito os definirá como valores padrão. Por exemplo, quando você já tiver uma `LambdaConfig` e enviar um `UpdateUserPool` com uma `LambdaConfig` em branco, exclua a atribuição de todas as funções do Lambda para acionadores do grupo de usuários. Planeje adequadamente quando quiser automatizar as alterações na configuração do grupo de usuários.

1. Capture o estado existente do seu grupo de usuários com DescribeUserPool.
2. Formate a saída do `DescribeUserPool` de forma que corresponda aos parâmetros da solicitação do `UpdateUserPool`. Remova os campos de nível superior a seguir e seus objetos secundários da saída JSON.

   - `Arn`
   - `CreationDate`
   - `CustomDomain`
     - Atualize esse campo com a operação UpdateUserPoolDomainda API.
   - `Domain`
     - Atualize esse campo com a operação UpdateUserPoolDomainda API.
   - `EmailConfigurationFailure`
   - `EstimatedNumberOfUsers`
   - `Id`
   - `LastModifiedDate`

- `Name`
- `SchemaAttributes`
- `SmsConfigurationFailure`
- `Status`

3. Confirme se o JSON resultante corresponde aos parâmetros da solicitação do `UpdateUserPool`.
4. Modifique todos os parâmetros que você deseja alterar no JSON resultante.
5. Envie uma solicitação de API `UpdateUserPool` com seu JSON modificado como entrada da solicitação.

Você também pode usar essa saída modificada do `DescribeUserPool` no parâmetro `--cli-input-json` do `update-user-pool` na AWS CLI.

Como alternativa, execute o AWS CLI comando a seguir para gerar JSON com valores em branco para os campos de entrada aceitos para. `update-user-pool` Depois, você pode preencher esses campos com os valores de seu grupo de usuários.

```
aws cognito-idp update-user-pool --generate-cli-skeleton --output json
```

Use o comando a seguir para gerar o mesmo objeto JSON para um cliente da aplicação.

```
aws cognito-idp update-user-pool-client --generate-cli-skeleton --output json
```

Configurações específicas da aplicação com clientes de aplicação

Um cliente de aplicação de grupo de usuários é uma configuração dentro de um grupo de usuários que interage com um aplicativo móvel ou uma aplicação web que se autentica no Amazon Cognito. Os clientes da aplicação podem chamar operações de API autenticadas e não autenticadas e ler ou modificar alguns ou todos os atributos dos usuários. A aplicação deve se identificar com o respectivo cliente nas operações de registro, login e tratamento de senhas esquecidas. Essas solicitações de API devem incluir autoidentificação com um ID do cliente da aplicação e autorização com um segredo opcional do cliente. Você deve proteger qualquer cliente de aplicativo IDs ou segredos para que somente aplicativos clientes autorizados possam chamar essas operações não autenticadas. Além disso, se você configurar seu aplicativo para assinar solicitações de API autenticadas com AWS credenciais, deverá protegê-las contra a inspeção do usuário.

É possível criar várias aplicações para um grupo de usuários. Um cliente da aplicação pode estar vinculado à plataforma de código de uma aplicação ou a um locatário separado no grupo de usuários. Por exemplo, você pode criar uma aplicação para uma aplicação do lado do servidor e uma aplicação Android diferente. Cada aplicativo possui o seu próprio ID de cliente do aplicativo.

Você pode aplicar configurações para os seguintes recursos do grupo de usuários no nível do cliente de aplicação:

1. Analytics
2. Login gerenciado IdPs, tipos de concessão URLs, retorno de chamada e personalização
3. Servidores de recursos e escopos personalizados
4. Proteção contra ameaças
5. Atribuir permissões de leitura e gravação
6. Expiração e revogação do token
7. Fluxos de autenticação

Tipos de cliente de aplicação

Ao criar um cliente de aplicativo no Amazon Cognito, você pode pré-preencher as opções com base nos tipos de OAuth cliente padrão: cliente público e cliente confidencial. Configure um cliente confidencial com um segredo do cliente. Para obter mais informações sobre os tipos de clientes, consulte IETF RFC 6749 #2.1.

Cliente público

Um cliente público é executado em um navegador ou em um dispositivo móvel. Como ele não tem recursos confiáveis no lado do servidor, não tem um segredo do cliente.

Cliente confidencial

Um cliente confidencial tem recursos no lado do servidor que podem ser confiáveis com um segredo do cliente para operações de API não autenticadas. A aplicação pode ser executada como um daemon ou script shell no servidor de backend.

Segredo do cliente

Um segredo do cliente, ou senha do cliente, é uma string fixa que a aplicação deve usar em todas as solicitações de API para o cliente da aplicação. O cliente da aplicação deve ter um segredo de

cliente para realizar concessões `client_credentials`. Para obter mais informações, consulte IETF RFC 6749 #2.3.1.

Cada cliente de aplicativo pode ter até dois segredos por vez, permitindo a rotação de segredos sem tempo de inatividade. Ao criar um cliente de aplicativo, você pode permitir que o Amazon Cognito gere um valor secreto ou forneça seu próprio valor secreto personalizado. Você não pode alterar os segredos depois de criar uma aplicação. Você pode adicionar um segundo segredo com a operação da AddUserPoolClientSecretAPI para alternar os segredos. Ao adicionar um segredo, você pode deixar o Amazon Cognito gerar um valor secreto ou fornecer seu próprio valor secreto personalizado. Para excluir um segredo, use a operação DeleteUserPoolClientSecretda API. Você não pode excluir o único segredo associado a um cliente de aplicativo. Também é possível excluir um aplicativo para bloquear o acesso de aplicativos que usam esse ID de cliente de aplicativo.

> **Note**
>
> O console do Amazon Cognito cria clientes de aplicativos com segredos de clientes quando você seleciona as opções de aplicativo web tradicional e achine-to-machine aplicativo M para o tipo de aplicativo. Escolha uma dessas opções para gerar um segredo do cliente ou crie o cliente programaticamente com CreateUserPoolCliente `GenerateSecret` defina como. `true`

Você pode usar um cliente confidencial e um segredo do cliente com uma aplicação pública. Use um CloudFront proxy da Amazon para adicionar um `SECRET_HASH` em trânsito. Para obter mais informações, consulte Proteger clientes públicos do Amazon Cognito usando um CloudFront proxy da Amazon no AWS blog.

Token JSON da web

Os clientes do aplicativo Amazon Cognito podem emitir tokens web JSON (JWTs) dos seguintes tipos.

Token de identidade (ID)

Uma declaração verificável de que o usuário está autenticado no grupo de usuários. O OpenID Connect (OIDC) adicionou a especificação do token de ID aos padrões de token de acesso e atualização definidos pela versão 2.0. OAuth O token de ID contém informações de identidade,

como atributos do usuário, que a aplicação pode usar para criar um perfil de usuário e provisionar recursos. Consulte Como entender o token de identidade (ID) para obter mais informações.

Token de acesso

Uma declaração verificável dos direitos de acesso do usuário. O token de acesso contém escopos, um recurso do OIDC e 2.0. OAuth A aplicação pode apresentar escopos para recursos de back-end e provar que o grupo de usuários autorizou um usuário ou uma máquina a acessar dados de uma API ou seus próprios dados de usuário. Um token de acesso com escopos personalizados, geralmente de uma concessão de credenciais de cliente M2M, autoriza o acesso a um servidor de recursos. Consulte Como entender o token de acesso para obter mais informações.

Token de atualização

Uma declaração criptografada da autenticação inicial que a aplicação pode apresentar ao grupo de usuários quando os tokens do usuário expirarem. Uma solicitação de token de atualização retorna tokens de acesso e ID novos e não expirados. Consulte Tokens de atualização para obter mais informações.

É possível definir a expiração desses tokens para cada cliente da aplicação no menu Clientes da aplicação do grupo de usuários no console do Amazon Cognito.

Termos do cliente da aplicação

Os seguintes termos são propriedades disponíveis para clientes da aplicação no console do Amazon Cognito.

Retorno de chamada permitido URLs

Um URL de retorno de chamada indica para onde o usuário será redirecionado após um acesso bem-sucedido. Escolha pelo menos um URL de retorno de chamada. O URL de retorno de chamada deve:

- Ser um URI absoluto.
- Estar pré-registrado com um cliente.
- Não incluir um componente de fragmento.

Consulte OAuth 2.0 - endpoint de redirecionamento.

O Amazon Cognito exige `HTTPS` em vez de `HTTP`, exceto `http://localhost` somente para fins de teste.

Também há suporte para retornos de chamadas de aplicativos, URLs como `myapp://example` os do tipo.

Sair permitido URLs

Um URL de saída indica para onde o usuário deve ser redirecionado após fazer logoff.

Atribuir permissões de leitura e gravação

Seu grupo de usuários pode ter muitos clientes, cada um com seu próprio cliente de aplicativo IdPs e. Você pode configurar o cliente da aplicação para ter acesso de leitura e gravação somente aos atributos de usuário relevantes para a aplicação. Em casos como autorização machine-to-machine (M2M), você não pode conceder acesso a nenhum dos seus atributos de usuário.

Considerações sobre a configuração de permissões de leitura e gravação de atributos

- Quando você cria um cliente de aplicação e não personaliza as permissões de leitura e gravação de atributos, o Amazon Cognito concede permissões de leitura e gravação a todos os atributos do grupo de usuários.
- É possível conceder acesso de gravação a atributos personalizados imutáveis. O cliente da aplicação pode gravar valores em um atributo imutável quando você cria ou cadastra um usuário. Depois disso, não é possível gravar valores em nenhum atributo personalizado imutável para o usuário.
- Os clientes da aplicação devem ter acesso de gravação aos atributos necessários em seu grupo de usuários. O console do Amazon Cognito define automaticamente os atributos necessários como graváveis.
- Não é possível permitir que um cliente de aplicação tenha acesso de gravação a `email_verified` ou `phone_number_verified`. O administrador do grupo de usuários pode modificar esses valores. Um usuário só pode alterar o valor desses atributos por meio da verificação de atributos.

Fluxos de autenticação

Os métodos que o cliente da aplicação permite para fazer login. Seu aplicativo pode oferecer suporte à autenticação com nome de usuário e senha, e-mail e mensagem SMS OTPs, autenticadores de chave de acesso, autenticação personalizada com acionadores Lambda e atualização de token. Como prática recomendada de segurança, use a autenticação SRP para autenticação de nome de usuário e senha em aplicações personalizadas.

Escopos personalizados

Um escopo personalizado é aquele definido para o seu próprio servidor de recursos em Resource Servers (Servidores de recursos). O formato é*`resource-server-identifier`*/*`scope`*. Consulte Escopos, M2M e servidores de recursos.

URI de redirecionamento padrão

Substitui o `redirect_uri` parâmetro nas solicitações de autenticação de usuários por terceiros IdPs. Defina essa configuração do cliente do aplicativo com o `DefaultRedirectURI` parâmetro de uma solicitação de UpdateUserPoolClientAPI CreateUserPoolClientou. Esse URL também deve ser membro de `CallbackURLs` para seu cliente de aplicação. O Amazon Cognito redireciona as sessões autenticadas para esse URL quando:

1. Seu cliente de aplicativo tem um provedor de identidade atribuído e vários retornos de chamada URLs definidos. Seu grupo de usuários redireciona as solicitações de autenticação para o servidor de autorização para o URI de redirecionamento padrão quando elas não incluem um parâmetro `redirect_uri`.
2. Seu cliente de aplicativo tem um provedor de identidade atribuído e um retorno de chamada URLs definido. Nesse cenário, não é necessário definir um URL de retorno de chamada padrão. Solicitações que não incluem um parâmetro `redirect_uri` redirecionam para o único URL de retorno de chamada disponível.

Provedores de identidade

Você pode escolher alguns ou todos os provedores de identidade externos (IdPs) do seu grupo de usuários para autenticar seus usuários. O cliente da aplicação também pode autenticar apenas usuários locais no grupo de usuários. Ao adicionar um IdP ao cliente da aplicação, é possível gerar links de autorização para o IdP e exibi-los na página de login do login gerenciado. Você pode atribuir vários IdPs, mas deve atribuir pelo menos um. Para obter mais informações sobre o uso externo IdPs, consulteLogin do grupo de usuários com provedores de identidades de terceiros.

Escopos do OpenID Connect

Selecione um ou mais dos seguintes escopos `0Auth` para especificar os privilégios de acesso que podem ser solicitados para tokens de acesso.

- O escopo `openid` declara que você deseja recuperar um token de ID e um ID exclusivo do usuário. Ele também solicita todos ou alguns atributos do usuário, dependendo dos escopos adicionais na solicitação. O Amazon Cognito não retorna um token de ID, a menos que você solicite o escopo `openid`. O escopo `openid` autoriza declarações de token de ID estrutural,

como expiração e ID da chave, e determina os atributos do usuário que você recebe em uma resposta do endpoint userinfo.

- Quando `openid` é o único escopo que você solicita, o Amazon Cognito preenche o token de ID com todos os atributos do usuário que o cliente atual da aplicação pode ler. A resposta `userInfo` a um token de acesso somente com esse escopo exibe todos os atributos do usuário.
- Quando você solicita `openid` com outros escopos, como `phone`, `email` ou `profile`, o token de ID e `userInfo` exibem o ID exclusivo do usuário e os atributos definidos pelos escopos adicionais.

- O escopo `phone` concede acesso às requisições `phone_number` e `phone_number_verified`. Esse escopo só pode ser solicitado com o escopo `openid`.
- O escopo `email` concede acesso às requisições `email` e `email_verified`. Esse escopo só pode ser solicitado com o escopo `openid`.
- O `aws.cognito.signin.user.admin` escopo concede acesso às operações de API dos grupos de usuários do Amazon Cognito que exigem tokens de acesso, como e. UpdateUserAttributesVerifyUserAttribute
- O escopo `profile` concede acesso a todos os atributos do usuário que são legíveis pelo cliente. Esse escopo só pode ser solicitado com o escopo `openid`.

Para obter mais informações sobre os escopos, consulte a lista de escopos OIDC padrão.

OAuth tipos de subsídios

Uma OAuth concessão é um método de autenticação que recupera tokens do grupo de usuários. O Amazon Cognito agora é compatível com seguintes tipos de concessões. Para integrar essas OAuth concessões ao seu aplicativo, você deve adicionar um domínio ao seu grupo de usuários.

Concessão de código de autorização

A concessão do código de autorização gera um código que a aplicação pode trocar por tokens do grupo de usuários com o Endpoint de token. Quando você troca um código de autorização, a aplicação recebe tokens de ID, acesso e atualização. Esse OAuth fluxo, como a concessão implícita, acontece nos navegadores dos seus usuários. Uma concessão de código de autorização é a concessão mais segura que o Amazon Cognito oferece, porque os tokens não são visíveis nas sessões dos usuários. Em vez disso, a aplicação gera a solicitação que retorna tokens e pode armazená-los em cache no armazenamento protegido. Para obter mais informações, consulte Authorization code no IETF RFC 6749 #1.3.1.

> **Note**
>
> Como melhor prática de segurança em aplicativos de clientes públicos, ative somente o OAuth fluxo de concessão do código de autorização e implemente o Proof Key for Code Exchange (PKCE) para restringir a troca de tokens. Com o PKCE, um cliente só pode trocar um código de autorização depois de fornecer ao endpoint do token o mesmo segredo apresentado na solicitação de autenticação original. Para obter mais informações sobre PKCE, consulte IETF RFC 7636.

Concessão implícita

A concessão implícita fornece um token de acesso e ID, mas não um token de atualização, à sessão do navegador do usuário diretamente do Autorizar endpoint. Uma concessão implícita remove a exigência de uma solicitação separada para o endpoint do token, mas não é compatível com o PKCE e não retorna tokens de atualização. Essa concessão acomoda cenários de teste e arquitetura de aplicação que não podem concluir concessões de código de autorização. Para obter mais informações, consulte Implicit grant em IETF RFC 6749 #1.3.2. É possível ativar tanto a concessão de código de autorização como a concessão implícita em um cliente da aplicação e usar cada concessão conforme necessário.

Concessão de credenciais do cliente

A concessão de credenciais do cliente é para comunicações machine-to-machine (M2M). O código de autorização e as concessões implícitas emitem tokens para usuários humanos autenticados. As credenciais do cliente concedem autorização baseada em escopo de um sistema não interativo para uma API. A aplicação pode solicitar credenciais do cliente diretamente do endpoint do token e receber um token de acesso. Para obter mais informações, consulte Client Credentials em IETF RFC 6749 #1.3.4. Você só pode ativar concessões de credenciais de cliente em clientes de aplicações que tenham um segredo de cliente e que não permitam códigos de autorização ou concessões implícitas.

> **Note**
>
> Como você não invoca o fluxo de credenciais do cliente como usuário, essa concessão só pode adicionar escopos personalizados a tokens de acesso. Um escopo personalizado é aquele definido para o seu próprio servidor de recursos. Os escopos-padrão, como `openid` e `profile`, não se aplicam a usuários não humanos.

Como os tokens de ID são uma validação dos atributos do usuário, eles não são relevantes para a comunicação M2M, e as concessões de credenciais de um cliente não os emitem. Consulte Escopos, M2M e servidores de recursos.

As concessões de credenciais do cliente adicionam custos à sua AWS fatura. Para mais informações, consulte Preço do Amazon Cognito.

Criar um cliente de aplicação

Console de gerenciamento da AWS

Para criar um cliente de aplicação (console)

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários. Ambas as opções solicitam que você configure um cliente da aplicação com configurações específicas da aplicação.
4. Selecione um Tipo de aplicação que reflita a arquitetura da aplicação.
5. Dê um nome para sua aplicação com um identificador amigável.
6. Insira um URL de retorno.
7. Escolha Criar cliente da aplicação. Você pode alterar as opções avançadas após criar o cliente da aplicação.
8. O Amazon Cognito retorna você aos detalhes do cliente da aplicação. Para acessar o código de exemplo da sua aplicação, selecione uma plataforma na guia Guia de configuração rápida.

AWS CLI

```
aws cognito-idp create-user-pool-client --user-pool-id MyUserPoolID --client-
name myApp
```

> **Note**
>
> Use o formato JSON para retorno de chamada e saída para evitar que URLs a CLI os trate como arquivos de parâmetros remotos:
>
> ```
> --callback-urls "["https://example.com"]"
> --logout-urls "["https://example.com"]"
> ```

Consulte a referência do AWS CLI comando para obter mais informações: create-user-pool-client

Amazon Cognito user pools API

Gere uma solicitação de CreateUserPoolClientAPI. Você deve especificar um valor para todos os parâmetros que não deseja definir como padrão.

Atualização de um cliente de aplicativo de grupo de usuários (AWS CLI e AWS API)

No AWS CLI, digite o seguinte comando:

```
aws cognito-idp update-user-pool-client --user-pool-id  "MyUserPoolID" --client-id
 "MyAppClientID" --allowed-o-auth-flows-user-pool-client --allowed-o-auth-flows "code"
 "implicit" --allowed-o-auth-scopes "openid" --callback-urls ["https://example.com"]"
 --supported-identity-providers "["MySAMLIdP", "LoginWithAmazon"]"
```

Se o comando for bem-sucedido, ele AWS CLI retornará uma confirmação:

```
{
    "UserPoolClient": {
        "ClientId": "MyClientID",
        "SupportedIdentityProviders": [
            "LoginWithAmazon",
            "MySAMLIdP"
        ],
        "CallbackURLs": [
            "https://example.com"
        ],
        "AllowedOAuthScopes": [
            "openid"
        ],
        "ClientName": "Example",
```

```
        "AllowedOAuthFlows": [
            "implicit",
            "code"
        ],
        "RefreshTokenValidity": 30,
        "AuthSessionValidity": 3,
        "CreationDate": 1524628110.29,
        "AllowedOAuthFlowsUserPoolClient": true,
        "UserPoolId": "MyUserPoolID",
        "LastModifiedDate": 1530055177.553
    }
}
```

Consulte a referência do AWS CLI comando para obter mais informações: update-user-pool-client.

AWS API: UpdateUserPoolClient

Obter informações sobre um cliente de aplicativo de grupo de usuários (AWS CLI e AWS API)

```
aws cognito-idp describe-user-pool-client --user-pool-id MyUserPoolID --client-
id MyClientID
```

Consulte a referência do AWS CLI comando para obter mais informações: describe-user-pool-client.

AWS API: DescribeUserPoolClient

Listar todas as informações do cliente do aplicativo em um grupo de usuários (AWS CLI e AWS API)

```
aws cognito-idp list-user-pool-clients --user-pool-id "MyUserPoolID" --max-results 3
```

Consulte a referência do AWS CLI comando para obter mais informações: list-user-pool-clients.

AWS API: ListUserPoolClients

Excluindo um cliente de aplicativo de grupo de usuários (AWS CLI e AWS API)

```
aws cognito-idp delete-user-pool-client --user-pool-id "MyUserPoolID" --client-id
 "MyAppClientID"
```

Consulte a referência do AWS CLI comando para obter mais informações: delete-user-pool-client

AWS API: DeleteUserPoolClient

Trabalhar com dispositivos de usuários no grupo de usuários

Ao fazer login de usuários de grupos de usuários locais com a API de grupos de usuários do Amazon Cognito, é possível associar os logs de atividades dos usuários, provenientes da proteção contra ameaças, a cada um dos dispositivos e, opcionalmente, permitir que os usuários ignorem a autenticação multifator (MFA) se estiverem em um dispositivo confiável. O Amazon Cognito inclui uma chave de dispositivo na resposta a qualquer login que ainda não inclua informações do dispositivo. A chave do dispositivo está no formato *Region*_*UUID*. Com uma chave de dispositivo, uma biblioteca de senha remota segura (SRP) e um grupo de usuários que permita a autenticação do dispositivo, é possível solicitar que os usuários da aplicação confiem no dispositivo atual e não solicitem mais um código de MFA no login.

Tópicos

- Como configurar dispositivos memorizados
- Obter uma chave do dispositivo
- Fazer login com um dispositivo
- Visualizar, atualizar e esquecer dispositivos

Como configurar dispositivos memorizados

Com os grupos de usuários do Amazon Cognito, é possível associar cada um dos dispositivos dos usuários a um identificador de dispositivo exclusivo: uma chave de dispositivo. Ao apresentar a chave do dispositivo e realizar a autenticação do dispositivo no login, é possível configurar a aplicação com um fluxo de autenticação de dispositivo confiável. Nesse fluxo, a aplicação pode apresentar aos usuários a opção de fazer login sem MFA em outro momento, conforme os requisitos de segurança da aplicação ou as preferências dos usuários. Ao final desse período, a aplicação deve alterar o status do dispositivo para não memorizado e o usuário deve fazer login com MFA até confirmar que deseja memorizar um dispositivo. Por exemplo, a aplicação pode solicitar que seus usuários confiem em um dispositivo por 30, 60 ou 90 dias. Você pode armazenar essa data em um atributo personalizado e, nessa data, alterar o status de memorização do dispositivo. Em seguida, é necessário solicitar novamente ao usuário que envie um código de MFA e configure o dispositivo para que seja novamente memorizado após a autenticação bem-sucedida.

1. Os dispositivos memorizados podem substituir a MFA somente em grupos de usuários com a MFA ativa.

Quando o usuário faz login com um dispositivo memorizado, é necessário realizar uma autenticação adicional do dispositivo durante o fluxo de autenticação. Para obter mais informações, consulte Fazer login com um dispositivo.

Configure o grupo de usuários para memorizar os dispositivos no menu Fazer login do grupo de usuários, em Monitoramento de dispositivos. Ao configurar a funcionalidade de dispositivos memorizados por meio do console do Amazon Cognito, você terá três opções: Always (Sempre), User Opt-In (Usuário opta por) e No (Não).

Não memorizar

O grupo de usuários não solicita que os usuários se lembrem dos dispositivos ao fazerem login.

Sempre memorizar

Quando a aplicação confirma o dispositivo de um usuário, o grupo de usuários sempre se lembra do dispositivo e não retorna desafios de MFA em futuros logins bem-sucedidos do dispositivo.

Opção do usuário

Quando a aplicação confirma o dispositivo de um usuário, o grupo de usuários não suprime automaticamente os desafios de MFA. É necessário solicitar que o usuário escolha se deseja memorizar o dispositivo.

Ao selecionar Sempre memorizar ou Opção do usuário, o Amazon Cognito gera uma chave e um segredo de identificação do dispositivo toda vez que um usuário faz login em um dispositivo não identificado. A chave do dispositivo é o identificador inicial que a aplicação envia ao grupo de usuários quando o usuário realiza a autenticação do dispositivo.

Com cada dispositivo de usuário confirmado, seja lembrado automaticamente ou por opção, é possível usar a chave e o segredo do identificador do dispositivo para autenticar um dispositivo em cada login de usuário.

Você também pode definir as configurações de dispositivos memorizados para seu grupo de usuários em uma solicitação de UpdateUserPoolAPI CreateUserPoolou API. Para obter mais informações, consulte a DeviceConfigurationpropriedade.

A API de grupos de usuários do Amazon Cognito tem operações adicionais para dispositivos memorizados.

1. ListDevicese AdminListDevicesretorne uma lista das chaves do dispositivo e seus metadados para um usuário.

2. GetDevicee AdminGetDeviceretorne a chave do dispositivo e os metadados de um único dispositivo.
3. UpdateDeviceStatuse AdminUpdateDeviceStatusdefina o dispositivo do usuário como lembrado ou não lembrado.
4. ForgetDevicee AdminForgetDeviceremova o dispositivo confirmado de um usuário do perfil dele.

As operações de API com nomes que começam com `Admin` são para uso em aplicações do lado do servidor e devem ser autorizadas com credenciais do IAM. Para obter mais informações, consulte Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado.

Obter uma chave do dispositivo

Sempre que o usuário faz login com a API de grupos de usuários e não inclui uma chave do dispositivo nos parâmetros de autenticação como `DEVICE_KEY`, o Amazon Cognito gera uma nova chave do dispositivo na resposta. Na aplicação pública do lado do cliente, coloque a chave do dispositivo no armazenamento da aplicação para que você possa incluí-la em futuras solicitações. Na aplicação confidencial do lado do servidor, defina um cookie do navegador ou outro token do lado do cliente com a chave do dispositivo do usuário.

Para que o usuário possa fazer login com o dispositivo confiável, a aplicação deve confirmar a chave do dispositivo e fornecer informações adicionais. Gere uma ConfirmDevicesolicitação para o Amazon Cognito que confirme o dispositivo do seu usuário com a chave do dispositivo, um nome amigável, um verificador de senha e um sal. Se você configurou o grupo de usuários para autenticação opcional de dispositivos, o Amazon Cognito responderá à solicitação `ConfirmDevice` pedindo que o usuário escolha se deseja memorizar o dispositivo atual. Responda com a seleção do seu usuário em uma UpdateDeviceStatussolicitação.

Ao confirmar o dispositivo do usuário, mas não o configurar como memorizado, o Amazon Cognito armazena a associação, mas prossegue com o login que não é do dispositivo quando você fornece a respectiva chave. Os dispositivos podem gerar logs úteis para a segurança e solução de problemas do usuário. Um dispositivo confirmado, mas não memorizado, não utiliza o recurso de login, e sim o de logs de monitoramento de segurança. Ao ativar a proteção contra ameaças para o cliente da aplicação e codificar uma impressão digital do dispositivo na solicitação, o Amazon Cognito associa os eventos do usuário ao dispositivo confirmado.

Como obter uma nova chave do dispositivo

1. Inicie a sessão de login do seu usuário com uma solicitação de InitiateAuthAPI.

2. Responda a todos os desafios de autenticação RespondToAuthChallengeaté receber tokens web JSON (JWTs) que marquem a sessão de login do usuário como concluída.
3. Na aplicação, registre os valores que o Amazon Cognito gera em `NewDeviceMetadata` na resposta `RespondToAuthChallenge` ou `InitiateAuth`: `DeviceGroupKey` e `DeviceKey`.
4. Gere um novo segredo de SRP para o usuário: um salt e um verificador de senha. Essa função está disponível em SDKs que fornecem bibliotecas SRP.
5. Solicite ao usuário um nome de dispositivo ou gere um com base nas características do dispositivo do usuário.
6. Forneça o token de acesso, a chave do dispositivo, o nome do dispositivo e o segredo SRP do usuário em uma solicitação de ConfirmDeviceAPI. Se o grupo de usuários estiver definido como Sempre memorizar os dispositivos, o registro do usuário estará concluído.
7. Se o Amazon Cognito respondeu a `ConfirmDevice` com `"UserConfirmationNecessary": true`, solicite que o usuário escolha se gostaria de memorizar o dispositivo. Se eles afirmarem que querem se lembrar do dispositivo, gere uma solicitação de UpdateDeviceStatusAPI com o token de acesso, a chave do dispositivo e. `"DeviceRememberedStatus": "remembered"`
8. Se você instruiu o Amazon Cognito a memorizar o dispositivo, na próxima vez em que ele fizer login, em vez de um desafio de MFA, será apresentado um desafio `DEVICE_SRP_AUTH`.

Fazer login com um dispositivo

Depois que o dispositivo de um usuário é configurado para ser memorizado, o Amazon Cognito não exige mais que ele envie um código de MFA ao fazer login com a mesma chave do dispositivo. A autenticação do dispositivo substitui apenas o desafio da autenticação MFA por um desafio de autenticação do dispositivo. Não é possível conectar os usuários somente com a autenticação do dispositivo. O usuário deve primeiro concluir a autenticação com a senha ou um desafio personalizado. Veja a seguir o processo de autenticação de um usuário em um dispositivo memorizado.

Para realizar a autenticação do dispositivo em um fluxo que usa gatilhos Lambda do desafio de autenticação personalizada, passe um `DEVICE_KEY` parâmetro na sua solicitação de API. InitiateAuth Depois que o usuário passar por todos os desafios e o desafio `CUSTOM_CHALLENGE` gerar um valor `issueTokens` de `true`, o Amazon Cognito vai gerar um desafio `DEVICE_SRP_AUTH` final.

Como fazer login com um dispositivo

1. Recupere a chave do dispositivo do usuário do armazenamento do cliente.
2. Inicie a sessão de login do seu usuário com uma solicitação de InitiateAuthAPI. Selecione um `AuthFlow` de `USER_SRP_AUTH`, `REFRESH_TOKEN_AUTH`, `USER_PASSWORD_AUTH` ou `CUSTOM_AUTH`. Em `AuthParameters`, adicione a chave do dispositivo do usuário ao parâmetro `DEVICE_KEY` e inclua os outros parâmetros necessários para o fluxo de login selecionado.
   a. Também é possível transmitir `DEVICE_KEY` nos parâmetros de uma resposta `PASSWORD_VERIFIER` a um desafio de autenticação.
3. Forneça as respostas do desafio até receber um desafio `DEVICE_SRP_AUTH` na resposta.
4. Em uma solicitação de RespondToAuthChallengeAPI, envie um `ChallengeName` de `DEVICE_SRP_AUTH` e parâmetros para `USERNAMEDEVICE_KEY`, `SRP_A` e.
5. O Amazon Cognito responde com um desafio `DEVICE_PASSWORD_VERIFIER`. Essa resposta ao desafio inclui valores para `SECRET_BLOCK` e `SRP_B`.
6. Com a biblioteca de SRP, gere e envie os parâmetros `PASSWORD_CLAIM_SIGNATURE`, `PASSWORD_CLAIM_SECRET_BLOCK`, `TIMESTAMP`, `USERNAME` e `DEVICE_KEY`. Envie-os em uma solicitação `RespondToAuthChallenge` adicional.
7. Complete desafios adicionais até receber os do usuário JWTs.

O pseudocódigo a seguir demonstra como calcular valores para a resposta `DEVICE_PASSWORD_VERIFIER` ao desafio. Para autenticação SRP com um dispositivo, gere um novo segredo de SRP para o usuário: uma nova senha de alta entropia `DeviceSecret`, um salt e o verificador de senha associado. Esses valores são distintos da senha, do salt e do verificador usados para a autenticação SRP do usuário. Eles são usados somente para autenticação do dispositivo e são armazenados somente no dispositivo. As funções para gerar os segredos SRP para os dispositivos dos usuários estão disponíveis em bibliotecas SRP que estão disponíveis em várias. SDKs

```
PASSWORD_CLAIM_SECRET_BLOCK = SECRET_BLOCK
TIMESTAMP = "Tue May 7 00:09:40 UTC 2025"
k = SHA256(N || g) as a non-negative integer in big-endian
u = SHA256(SRP_A || SRP_B) as a non-negative integer in big-endian
x = SHA256(salt || SHA256(DeviceGroupKey || DeviceKey || ":" || DeviceSecret)) as a
 non-negative integer in big-endian
S_USER = (SRP_B - k * g^x)^(a + u * x) % N
K_USER = HKDF_HMAC_SHA256(salt=u, ikm=S_USER, info="Caldera Derived Key", length=16
 bytes)
```

```
PASSWORD_CLAIM_SIGNATURE = Base64(HMAC_SHA256(key=K_USER, message=(DeviceGroupKey ||
 DeviceKey || PASSWORD_CLAIM_SECRET_BLOCK || TIMESTAMP)))
```

Visualizar, atualizar e esquecer dispositivos

Com a API do Amazon Cognito, é possível implementar os recursos a seguir na aplicação.

1. Exibir informações sobre o dispositivo atual do usuário.
2. Exiba uma lista de todos os dispositivos do usuário.
3. Esqueça um dispositivo.
4. Atualize o estado memorizado do dispositivo.

Os tokens de acesso que autorizam as solicitações de API nas descrições a seguir devem incluir o escopo `aws.cognito.signin.user.admin`. O Amazon Cognito adiciona uma reivindicação desse escopo a todos os tokens de acesso que você gera com a API de grupos de usuários do Amazon Cognito. IdPs Os terceiros devem gerenciar separadamente os dispositivos e o MFA para seus usuários que se autenticam no Amazon Cognito. No login gerenciado, é possível solicitar o escopo `aws.cognito.signin.user.admin`, mas o login gerenciado adiciona automaticamente as informações do dispositivo a logs de usuário de segurança avançados e não oferece a possibilidade de memorizar os dispositivos.

Exibir informações sobre um dispositivo

É possível consultar informações sobre o dispositivo de um usuário para determinar se ele ainda está em uso. Por exemplo, convém desativar dispositivos memorizados depois que eles não tiverem feito login por 90 dias.

- Para exibir as informações do dispositivo do usuário em um aplicativo cliente público, envie a chave de acesso e a chave do dispositivo do usuário em uma solicitação de GetDeviceAPI.
- Para exibir as informações do dispositivo do usuário em um aplicativo cliente confidencial, assine uma solicitação de AdminGetDeviceAPI com AWS credenciais e envie o nome de usuário, a chave do dispositivo e o grupo de usuários do usuário.

Exibir uma lista de todos os dispositivos do usuário.

É possível exibir uma lista de todos os dispositivos do usuário e as respectivas propriedades. Por exemplo, convém verificar se o dispositivo atual corresponde a um dispositivo memorizado.

- Em um aplicativo de cliente público, envie o token de acesso do usuário em uma solicitação de ListDevicesAPI.
- Em um aplicativo de cliente confidencial, assine uma solicitação de AdminListDevicesAPI com AWS credenciais e envie o nome de usuário e o grupo de usuários do seu usuário.

Esquecer um dispositivo

É possível excluir a chave do dispositivo de um usuário. Convém fazer isso ao constatar que o usuário não usa mais um dispositivo ou ao detectar atividades incomuns e solicitar que um usuário conclua a MFA novamente. Para registrar novamente o dispositivo em um momento posterior, é necessário gerar e armazenar uma nova chave do dispositivo.

- Em um aplicativo de cliente público, envie a chave do dispositivo e o token de acesso do usuário na solicitação ForgetDeviceda API.
- Em um aplicativo cliente confidencial, envie a chave do dispositivo e o token de acesso do usuário na solicitação AdminForgetDeviceda API.

Como usar o Amazon Pinpoint para análise de grupos de usuários

> **Note**
>
> Aviso de fim do suporte: em 30 de outubro de 2026, AWS encerrará o suporte para o Amazon Pinpoint. Após 30 de outubro de 2026, você não poderá mais acessar o console do Amazon Pinpoint nem seus recursos (endpoints, segmentos, campanhas, jornadas e analytics). Para obter mais informações, consulte Fim do suporte do Amazon Pinpoint. Observação: APIs relacionados a SMS, voz, push móvel, OTP e validação de número de telefone não são afetados por essa alteração e são compatíveis com o AWS End User Messaging.

Os grupos de usuários do Amazon Cognito são integrados ao Amazon Pinpoint para fornecer análise para grupos de usuários do Amazon Cognito e para enriquecer os dados do usuário para campanhas do Amazon Pinpoint. O Amazon Pinpoint fornece análise e campanhas direcionadas para promover o envolvimento dos usuários em aplicações móveis usando notificações por push. Com o suporte analítico do Amazon Pinpoint nos grupos de usuários do Amazon Cognito, você pode rastrear inscrições, logins, autenticações falhadas, usuários ativos diários () e usuários ativos mensais DAUs

() no console do Amazon Pinpoint. MAUs Você pode analisar os dados em diferentes faixas de datas ou de atributos, como plataforma de dispositivos, local do dispositivo e versão do aplicativo.

Também é possível configurar atributos personalizados para a aplicação. Eles poderão ser usados para segmentar seus usuários no Amazon Pinpoint e enviar notificações por push direcionadas a eles. Se você selecionar Compartilhar dados de atributos do usuário com o Amazon Pinpoint na configuração Analytics do seu cliente da aplicação no menu Clientes da aplicação no console do Amazon Cognito, o Amazon Pinpoint criará endpoints adicionais para os endereços de e-mail e números de telefone do usuário.

Ao ativar a análise do Amazon Pinpoint no grupo de usuários com o console do Amazon Cognito, você também cria um perfil vinculado ao serviço que o Amazon Cognito assume quando faz uma solicitação de API ao Amazon Pinpoint para o grupo de usuários. O diretor do IAM que adiciona sua configuração de análise deve ter CreateServiceLinkedRolepermissões. A função vinculada ao serviço é. AWSServiceRoleForAmazonCognitoIdp Para obter mais informações, consulte Como usar funções vinculadas a serviço para o Amazon Cognito.

Ao aplicar uma `AnalyticsConfiguration` ao cliente da aplicação na API do Amazon Cognito, você pode atribuir um perfil do IAM personalizado ao Amazon Pinpoint e um ID externo para assumir o perfil. O perfil deve confiar na entidade principal do serviço `cognito-idp` e, se a política de confiança do perfil exigir um ID externo, ela deverá corresponder à sua `AnalyticsConfiguration`. Você deve conceder as permissões `cognito-idp:Describe*` do perfil e as permissões a seguir a seu projeto do Amazon Pinpoint.

- `mobiletargeting:UpdateEndpoint`
- `mobiletargeting:PutEvents`

Disponibilidade de regiões do Amazon Cognito e Amazon Pinpoint

A tabela a seguir mostra os Região da AWS mapeamentos entre o Amazon Cognito e o Amazon Pinpoint que atendem a uma das seguintes condições.

- É possível usar somente um projeto do Amazon Pinpoint na região Leste dos EUA (Norte da Virgínia) (us-east-1).
- É possível usar um projeto do Amazon Pinpoint na mesma região ou na região Leste dos EUA (Norte da Virgínia) (us-east-1).

Por padrão, o Amazon Cognito só pode enviar análises para um projeto do Amazon Pinpoint na mesma Região da AWS. As exceções a essa regra são as regiões na tabela a seguir e as regiões em que o Amazon Pinpoint não está disponível.

O Amazon Pinpoint já está disponível nas regiões a seguir. Os grupos de usuários do Amazon Cognito nessas regiões não são compatíveis com a análise.

- Europa (Milão)
- Oriente Médio (Bahrein)
- Ásia-Pacífico (Osaka)
- Israel (Tel Aviv)
- África (Cidade do Cabo)
- Ásia-Pacífico (Jacarta)
- Ásia-Pacífico (Malásia)

A tabela mostra a relação entre a região em que você criou o grupo de usuários do Amazon Cognito e a região correspondente no Amazon Pinpoint. É necessário configurar o projeto do Amazon Pinpoint em uma região disponível para integrá-lo ao Amazon Cognito.

Região do grupo de usuários do Amazon Cognito	Região do projeto do Amazon Pinpoint
ap-northeast-1	us-east-1
ap-northeast-2	us-east-1
ap-south-1	us-east-1, ap-south-1
ap-southeast-1	us-east-1
ap-southeast-2	us-east-1, ap-southeast-2
ca-central-1	us-east-1
eu-central-1	us-east-1, eu-central-1
eu-west-1	us-east-1, eu-west-1

Região do grupo de usuários do Amazon Cognito	Região do projeto do Amazon Pinpoint
eu-west-2	us-east-1
us-east-1	us-east-1
us-east-2	us-east-1
us-west-2	us-east-1, us-west-2

Exemplos de mapeamento de região

- Se criar um grupo de usuários na região ap-northeast-1, você poderá criar o projeto do Amazon Pinpoint na região us-east-1.
- Se criar um grupo de usuários na região ap-south-1, você poderá criar o projeto do Amazon Pinpoint na região us-east-1 ou ap-south-1.

> **Note**
>
> Para todos Regiões da AWS , exceto aqueles na tabela anterior, o Amazon Cognito só pode usar um projeto Amazon Pinpoint na mesma região do seu grupo de usuários. Se o Amazon Pinpoint não estiver disponível na região onde você criou o grupo de usuários e não estiver listado na tabela, o Amazon Cognito não será compatível com as análises do Amazon Pinpoint nessa região. Para obter informações detalhadas sobre Região da AWS , consulte Amazon Pinpoint endpoints and quotas (Endpoints e cotas do Amazon Pinpoint).

Como especificar configurações de análise do Amazon Pinpoint (Console de gerenciamento da AWS)

É possível configurar o grupo de usuários do Amazon Cognito para enviar dados de análise ao Amazon Pinpoint. O Amazon Cognito só envia dados de análise ao Amazon Pinpoint para usuários locais. Depois de configurar o grupo de usuários para se associar a um projeto do Amazon Pinpoint, você deverá incluir `AnalyticsMetadata` em suas solicitações de API. Para obter mais informações, consulte Integrar sua aplicação ao Amazon Pinpoint.

Para especificar as configurações de análise

1. Acesse o console do Amazon Cognito. Podem ser solicitadas suas credenciais do AWS .
2. Selecione User Pools (Grupos de usuários) e escolha um grupo de usuários existente na lista.
3. Clique no menu Clientes da aplicação e selecione o cliente da aplicação que deseja atualizar.
4. Na guia Análise, em Análises do Pinpoint, selecione Habilitar.
5. Escolha uma Pinpoint Region (Região do Pinpoint).
6. Escolha um Amazon Pinpoint project (Projeto do Amazon Pinpoint) ou selecione Create Amazon Pinpoint project (Criar projeto do Amazon Pinpoint.

   > **Note**
   >
   > O ID de projeto do Amazon Pinpoint é uma string de 32 caracteres exclusiva para seu projeto do Amazon Pinpoint. Ele está listado no console do Amazon Pinpoint.
   > É possível mapear várias aplicações do Amazon Cognito em um único projeto do Amazon Pinpoint. No entanto, cada aplicação do Amazon Cognito pode ser mapeada somente em um projeto do Amazon Pinpoint.
   > No Amazon Pinpoint, cada projeto deve ser uma única aplicação. Por exemplo, se um desenvolvedor de jogos tiver dois jogos, cada jogo deverá ser um projeto do Amazon Pinpoint separado, mesmo se os dois jogos usarem o mesmo grupo de usuários do Amazon Cognito. Para obter mais informações sobre projetos do Amazon Pinpoint, consulte Criar um projeto no Amazon Pinpoint.

7. Em User data sharing (Compartilhamento de dados de usuários), selecione Share user data with Amazon Pinpoint (Compartilhar dados de usuários com o Amazon Pinpoint) se quiser que o Amazon Cognito envie endereços de e-mail e números de telefone ao Amazon Pinpoint e crie endpoints adicionais para os usuários. Depois que os usuários verificarem o endereço de e-mail e número de telefone, o Amazon Cognito só compartilhará esses dados com o Amazon Pinpoint se eles estiverem disponíveis para a conta do usuário.

   > **Note**
   >
   > Um endpoint identifica exclusivamente um dispositivo de usuário ao qual você pode enviar notificações por push com o Amazon Pinpoint. Para mais informações sobre endpoints, consulte Adicionar endpoints no Guia do desenvolvedor do Amazon Pinpoint.

8. Escolha Salvar alterações.

Especificação das configurações de análise AWS CLI (AWS e API) do Amazon Pinpoint

Use os comandos a seguir para especificar as configurações de análise do Amazon Pinpoint para seu grupo de usuários.

Para especificar as configurações de análise para o aplicativo cliente existente de seu grupo de usuários no momento da criação do aplicativo

- AWS CLI: `aws cognito-idp create-user-pool-client`
- AWS API: CreateUserPoolClient

Para atualizar as configurações de análise para o aplicativo cliente existente de seu grupo de usuários no momento da criação do aplicativo

- AWS CLI: `aws cognito-idp update-user-pool-client`
- AWS API: UpdateUserPoolClient

> **Note**
>
> O Amazon Cognito oferece suporte a integrações na região quando você usa o `ApplicationArn`

Integrar sua aplicação ao Amazon Pinpoint

Você pode publicar metadados de análise no Amazon Pinpoint para usuários nativos do Amazon Cognito na API de grupos de usuários.

Usuários locais

Usuários que se cadastraram em uma conta ou foram criados em seu grupo de usuários, em vez daqueles que fazem login por meio de um provedor de identidades (IdP) de terceiros.

API de grupos de usuários

As operações que você pode integrar a um AWS SDK usando um aplicativo com uma interface de usuário (UI) personalizada. Você não pode transmitir metadados de analytics para usuários federados ou locais que fazem login por meio do login gerenciado. Consulte Referência de API do Amazon Cognito para ter uma lista de operações da API de grupos de usuários.

Depois de configurar seu grupo de usuários para publicar em uma campanha, o Amazon Cognito transmite metadados ao Amazon Pinpoint para as operações de API a seguir.

- `AdminInitiateAuth`
- `AdminRespondToAuthChallenge`
- `ConfirmForgotPassword`
- `ConfirmSignUp`
- `ForgotPassword`
- `InitiateAuth`
- `ResendConfirmationCode`
- `RespondToAuthChallenge`
- `SignUp`

Para transmitir metadados sobre a sessão do usuário à sua campanha do Amazon Pinpoint, inclua um valor `AnalyticsEndpointId` no parâmetro `AnalyticsMetadata` da solicitação de API. JavaScript Por exemplo, consulte Por que minhas análises do grupo de usuários do Amazon Cognito não estão aparecendo no meu painel do Amazon Pinpoint? no Centro de AWS Conhecimento.

Configurações de e-mail para grupos de usuários do Amazon Cognito

Determinados eventos na aplicação podem fazer com que o Amazon Cognito envie e-mails para os usuários. Por exemplo, se você configurar o grupo de usuários para exigir verificação de e-mail, o Amazon Cognito enviará um e-mail quando um usuário se cadastrar em uma nova conta na aplicação ou redefinir a senha. Dependendo da ação que inicia o e-mail, o e-mail contém um código de verificação ou uma senha temporária.

Para processar a entrega de e-mails, você pode usar uma das seguintes opções:

- A configuração de e-mail padrão integrada ao serviço do Amazon Cognito.
- Sua configuração do Amazon Simple Email Service (Amazon SES).

Você pode alterar a opção de entrega depois de criar o grupo de usuários.

O Amazon Cognito envia mensagens de e-mail aos usuários com um código que eles podem inserir ou um link de URL que pode ser selecionado. A tabela a seguir mostra os eventos que podem gerar uma mensagem de e-mail.

Opções de mensagem

Atividade	Operação de API	Opções de entrega	Opções de formato	Personalizável	Modelo de mensagem
Esqueci a senha	ForgotPassword, AdminResetUserPassword	E-mail, SMS	código	Sim	Mensagem de verificação
Convite	AdminCreateUser	E-mail, SMS	código	Sim	Mensagem de convite
Autorregistro	SignUp, ResendConfirmationCode	E-mail, SMS	código, link	Sim	Mensagem de verificação
Verificação de endereço de e-mail ou número de telefone	UpdateUserAttributes, AdminUpdateUserAttributes, GetUserAttributeVerificationCode	E-mail, SMS	código	Sim	Mensagem de verificação
Autenticação multifatorial (MFA)	AdminInitiateAuth, InitiateAuth	E-mail[1], SMS, aplicativo autenticador	código	Sim[2]	Mensagem de MFA
Autenticação de senha única (OTP)	AdminInitiateAuth, InitiateAuth	E-mail[1], SMS	código	Sim	Mensagem de MFA [3]

[1] Requer o plano de recursos Essentials ou superior e a configuração de e-mail do Amazon SES.

[2] Para mensagens SMS e e-mail.

[3] Você só pode personalizar o modelo de mensagem de MFA quando a MFA é obrigatória ou opcional no grupo de usuários. Quando a MFA está inativa, o Amazon Cognito envia senhas de uso único com o modelo padrão.

O Amazon SES cobra por mensagens de e-mail. Para obter mais informações, consulte Definição de preço do Amazon SES.

Para saber mais sobre a MFA do e-mail, consulte MFA de mensagens SMS e e-mail.

O Amazon Cognito pode impedir a entrega de mensagens adicionais de e-mail ou SMS para um único destino em um curto período. Se você acredita que seu grupo de usuários foi afetado, configure e analise os logs de erros de entrega de mensagens e entre em contato com a equipe da sua conta.

Configuração de e-mail padrão

O Amazon Cognito pode usar sua configuração de e-mail padrão para processar entregas de e-mail para você. Quando você usa a opção padrão, o Amazon Cognito limita o número de e-mails que ele envia por dia para o grupo de usuários. Para obter mais informações sobre limites do serviço, consulte Cotas no Amazon Cognito. Para ambientes de produção típicos, o limite de e-mails padrão fica abaixo do volume de entrega necessário. Para habilitar um volume de entrega maior, você deve usar a configuração de e-mail do Amazon SES.

Ao usar a configuração padrão, você usa os recursos do Amazon SES que são gerenciados pela AWS para enviar mensagens de e-mail. O Amazon SES adiciona endereços de e-mail que retornam uma devolução definitiva para uma lista de supressão em nível de conta ou uma lista de supressão global. Se um endereço de e-mail impossível de entregar puder ser entregue posteriormente, você não poderá controlar sua remoção da lista de supressão enquanto o grupo de usuários estiver configurado para usar a configuração padrão. Um endereço de e-mail pode permanecer na lista de supressão AWS gerenciada indefinidamente. Para gerenciar endereços de e-mail que não podem ser entregues, use sua configuração de e-mail do Amazon SES com uma lista de supressão em nível de conta, conforme descrito na próxima seção.

Ao usar a configuração de e-mail padrão, você pode utilizar um dos seguintes endereços de e-mail como endereço DE:

- O endereço de e-mail padrão, no-reply@verificationemail.com.
- Um endereço de e-mail personalizado. Para poder usar seu próprio endereço de e-mail, verifique-o no Amazon SES e conceda permissão ao Amazon Cognito para usá-lo.

Configuração de e-mail do Amazon SES

O aplicativo pode exigir um volume de entrega maior do que está disponível com a opção padrão. Para aumentar o volume de entrega possível, use os recursos do Amazon SES com o grupo de usuários para enviar e-mail aos usuários. Você também pode monitorar a atividade de envio de e-mail ao enviar mensagens usando sua própria configuração do Amazon SES.

Antes de poder usar a configuração do Amazon SES, você deve verificar um ou mais endereços de e-mail ou de domínio no Amazon SES. Use um endereço de e-mail ou de domínio verificado como o endereço de e-mail FROM (DE) que você atribui ao grupo de usuários. Quando o Amazon Cognito envia um e-mail a um usuário, ele chama o Amazon SES para você e usa seu endereço de e-mail.

Quando você usa a configuração do Amazon SES, as seguintes condições se aplicam:

- Os limites de entrega de e-mail para seu grupo de usuários são os mesmos que se aplicam ao endereço de e-mail verificado do Amazon SES em sua Conta da AWS.
- Você pode gerenciar suas mensagens para endereços de e-mail que não podem ser entregues com uma lista de supressão em nível de conta no Amazon SES que substitui a lista de supressão global. Ao usar uma lista de supressão em nível de conta, as devoluções de mensagens de e-mail afetam a reputação de sua conta como remetente. Para obter mais informações, consulte Como usar a lista de supressão do Amazon SES por conta no Guia do desenvolvedor do Amazon Simple Email Service.

Regiões de configuração de e-mail do Amazon SES

O Região da AWS local onde você cria um grupo de usuários terá um dos três requisitos para a configuração de mensagens de e-mail com o Amazon SES. Você pode enviar mensagens de e-mail do Amazon SES na mesma região do seu grupo de usuários, em várias regiões, incluindo a mesma região, ou em uma ou mais regiões remotas. Para obter o melhor desempenho, envie mensagens de e-mail com uma identidade verificada do Amazon SES na mesma região do seu grupo de usuários quando você tiver a opção.

Categorias de requisitos regionais para identidades verificadas pelo Amazon SES

Somente na região

Seus grupos de usuários podem enviar mensagens de e-mail com identidades verificadas da Região da AWS mesma forma que o grupo de usuários. Na configuração de e-mail padrão

sem um endereço de e-mail personalizado `FROM`, o Amazon Cognito usa uma identidade `no-reply@verificationemail.com` verificada na mesma região.

Retrocompatibilidade

Seus grupos de usuários podem enviar mensagens de e-mail com identidades verificadas na mesma região Região da AWS ou em uma das seguintes regiões alternativas:

- Leste dos EUA (N. da Virgínia)
- Oeste dos EUA (Oregon)
- Europa (Irlanda)

Esse recurso mantém a continuidade para recursos de grupo de usuários criados para atender aos requisitos do Amazon Cognito quando o serviço foi iniciado. Os grupos de usuários desse período só podiam enviar mensagens de e-mail com identidades verificadas em um número limitado de Regiões da AWS. Na configuração de e-mail padrão sem um endereço de e-mail personalizado `FROM`, o Amazon Cognito usa uma identidade `no-reply@verificationemail.com` verificada na mesma região.

Região alternativa

Seus grupos de usuários podem enviar mensagens de e-mail com identidades verificadas em uma alternativa Região da AWS que esteja fora da região do grupo de usuários. Essa configuração ocorre quando o Amazon SES não está disponível em uma região onde o Amazon Cognito está disponível.

A política de autorização de envio do Amazon SES para sua identidade verificada na região alternativa deve confiar no responsável pelo serviço Amazon Cognito da região de origem. Para obter mais informações, consulte Para conceder permissões para usar a configuração de e-mail padrão.

Em algumas dessas regiões, o Amazon Cognito divide as mensagens de e-mail entre duas regiões alternativas para a configuração de e-mail padrão do `COGNITO_DEFAULT`. Nesses casos, para usar um endereço de e-mail personalizado do `FROM`, a política de autorização de envio do Amazon SES para sua identidade verificada em cada região alternativa deve confiar no responsável pelo serviço Amazon Cognito da região de origem. Para obter mais informações, consulte Para conceder permissões para usar a configuração de e-mail padrão. Com a configuração de e-mail do Amazon SES de `DEVELOPER` nessas regiões, você deve usar uma identidade verificada na primeira região listada e configurá-la para confiar na entidade principal do serviço do Amazon Cognito na região do grupo de usuários. Por exemplo, em um

grupo de usuários no Oriente Médio (EAU), configure uma identidade verificada na Europa (Frankfurt) para confiar em `cognito-idp.me-central-1.amazonaws.com`. Na configuração de e-mail padrão sem um endereço de e-mail personalizado `FROM`, o Amazon Cognito usa uma identidade `no-reply@verificationemail.com` verificada em cada região.

> **Note**
>
> Sob a seguinte combinação de condições, você deve especificar o `SourceArn` parâmetro EmailConfigurationcom um curinga no elemento Região, no formato`arn:${Partition}:ses:*:${Account}:identity/${IdentityName}`. Isso permite que seu grupo de usuários envie mensagens de e-mail com identidades verificadas idênticas às suas Conta da AWS em ambos. Regiões da AWS
>
> - O seu EmailSendingAccount é`COGNITO_DEFAULT`.
> - Você quer usar um endereço personalizado `FROM`.
> - Seu grupo de usuários envia e-mails em uma Região alternativa.
> - Seu grupo de usuários tem uma segunda[1] região alternativa especificada na tabela de Regiões permitidas do Amazon SES a seguir.

Se você criar um grupo de usuários programaticamente — com um SDK AWS , a API ou CLI do Amazon Cognito, o AWS CDK, ou — seu grupo de usuários enviará mensagens de e-mail com a AWS CloudFormation identidade do Amazon SES `SourceArn` que o parâmetro de especifica para seu grupo de usuários. EmailConfiguration A identidade do Amazon SES deve ocupar um espaço suportado Região da AWS. Se sua `EmailSendingAccount` for `COGNITO_DEFAULT` e você não especificar um parâmetro `SourceArn`, o Amazon Cognito enviará mensagens de e-mail de `no-reply@verificationemail.com` usando recursos na região onde você criou o grupo de usuários.

A tabela a seguir mostra Regiões da AWS onde você pode usar as identidades do Amazon SES com o Amazon Cognito.

Região do grupo de usuários	Opção de região	Regiões permitidas do Amazon SES
Leste dos EUA (Norte da Virgínia)	Retrocompatibilidade	Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Leste dos EUA (Ohio)	Retrocompatibilidade	Leste dos EUA (Ohio), Leste dos EUA (Norte da Virgínia) , Oeste dos EUA (Oregon), Europa (Irlanda)
Oeste dos EUA (N. da Califórnia)	Somente na região	Oeste dos EUA (N. da Califórnia)
Oeste dos EUA (Oregon)	Retrocompatibilidade	Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Canadá (Central)	Retrocompatibilidade	Canadá (Central), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Oeste do Canadá (Calgary)	Região alternativa	Canadá (Central), Oeste dos EUA (N. da Califórnia)[1]
México (Centro)	Região alternativa	Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon)[1]
Ásia-Pacífico (Tóquio)	Retrocompatibilidade	Ásia-Pacífico (Tóquio), Leste dos EUA (Norte da Virgínia) , Oeste dos EUA (Oregon), Europa (Irlanda)
Ásia-Pacífico (Hong Kong)	Região alternativa	Ásia-Pacífico (Singapura), Ásia-Pacífico (Tóquio)[1]

Região do grupo de usuários	Opção de região	Regiões permitidas do Amazon SES
Ásia-Pacífico (Seul)	Retrocompatibilidade	Ásia-Pacífico (Seul), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Ásia-Pacífico (Malásia)	Região alternativa	Ásia-Pacífico (Sydney), Ásia-Pacífico (Singapura)[1]
Ásia-Pacífico (Tailândia)	Região alternativa	Ásia-Pacífico (Singapura), Ásia-Pacífico (Mumbai)[1]
Ásia-Pacífico (Mumbai)	Retrocompatibilidade	Ásia-Pacífico (Mumbai), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Ásia-Pacífico (Hyderabad)	Região alternativa	Ásia-Pacífico (Mumbai), Ásia-Pacífico (Singapura)[1]
Ásia-Pacífico (Singapura)	Retrocompatibilidade	Ásia-Pacífico (Singapura), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Ásia-Pacífico (Sydney)	Retrocompatibilidade	Ásia-Pacífico (Sydney), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Ásia-Pacífico (Osaka)	Somente na região	Ásia-Pacífico (Osaka)
Ásia-Pacífico (Jacarta)	Somente na região	Ásia-Pacífico (Jacarta)
Ásia-Pacífico (Melbourne)	Região alternativa	Ásia-Pacífico (Sydney), Ásia-Pacífico (Singapura)[1]

Região do grupo de usuários	Opção de região	Regiões permitidas do Amazon SES
Europa (Irlanda)	Retrocompatibilidade	Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Europa (Londres)	Retrocompatibilidade	Europa (Londres), Leste dos EUA (Norte da Virgínia) , Oeste dos EUA (Oregon), Europa (Irlanda)
Europa (Paris)	Somente na região	Europa (Paris)
Europa (Frankfurt)	Retrocompatibilidade	Europa (Frankfurt), Leste dos EUA (Norte da Virgínia) , Oeste dos EUA (Oregon), Europa (Irlanda)
Europa (Zurique)	Região alternativa	Europa (Frankfurt), Europa (Londres)[1]
Europa (Estocolmo)	Somente na região	Europa (Estocolmo)
Europa (Milão)	Somente na região	Europa (Milão)
Europa (Espanha)	Região alternativa	Europa (Paris), Europa (Estocolmo)[1]
Oriente Médio (Bahrein)	Somente na região	Oriente Médio (Bahrein)
Oriente Médio (Emirados Árabes Unidos)	Região alternativa	Europa (Frankfurt), Europa (Londres)[1]
América do Sul (São Paulo)	Somente na região	América do Sul (São Paulo)
Israel (Tel Aviv)	Somente na região	Israel (Tel Aviv)
África (Cidade do Cabo)	Somente na região	África (Cidade do Cabo)

[1] Usado em grupos de usuários com a configuração de e-mail padrão. O Amazon Cognito distribui mensagens de e-mail entre identidades verificadas com o mesmo endereço de e-mail em cada região. Para usar um endereço personalizado `FROM`, configure `EmailConfiguration` com um parâmetro `SourceArn` no formato `arn:${Partition}:ses:*:${Account}:identity/${IdentityName}`.

Configurar e-mail para seu grupo de usuários

Execute as etapas a seguir para definir as configurações de e-mail do grupo de usuários. Dependendo das configurações que você usa, pode precisar de permissões do IAM no Amazon SES, o AWS Identity and Access Management (IAM) e o Amazon Cognito.

> **Note**
>
> Não é possível compartilhar os recursos criados nessas etapas entre Contas da AWS. Por exemplo, não é possível configurar um grupo de usuários em uma conta e depois usá-la com um endereço de e-mail do Amazon SES em outra conta. Se você usar o Amazon Cognito em várias contas, repita essas etapas em cada uma.

Etapa 1: verificar seu endereço de e-mail ou domínio com o Amazon SES

Antes de configurar o grupo de usuários, você deve verificar um ou mais domínios ou endereços de e-mail com o Amazon SES se quiser executar uma das seguintes ações:

- Usar seu endereço de e-mail como endereço FROM
- Usar a configuração do Amazon SES para processar a entrega de e-mails

Ao verificar seu endereço de e-mail ou domínio, você confirma que é o proprietário, o que ajuda a impedir o uso não autorizado.

Para obter mais informações sobre a verificação de um endereço de e-mail com o Amazon SES, consulte Verificar um endereço de e-mail no Guia do desenvolvedor do Amazon Simple Email Service. Para mais informações sobre como verificar um domínio com o Amazon SES, consulte Verificar domínios.

Etapa 2: retirar sua conta da sandbox do Amazon SES

Ignore esta etapa se estiver usando a configuração de e-mail padrão do Amazon Cognito.

Quando você usa o Amazon SES pela primeira vez em qualquer um Região da AWS, ele coloca você Conta da AWS na sandbox do Amazon SES dessa região. O Amazon SES usa a sandbox para evitar fraudes e uso abusivo. Se você usar a configuração do Amazon SES para processar a entrega de e-mails, deverá remover sua Conta da AWS da sandbox para que o Amazon Cognito possa enviar e-mails aos usuários.

Na sandbox, o Amazon SES impõe restrições sobre a quantidade de e-mails que você pode enviar e onde pode enviá-los. Você pode enviar e-mails somente para endereços e domínios que você já tenha verificado no Amazon SES ou pode enviá-los para endereços do simulador de caixa postal do Amazon SES. Enquanto você Conta da AWS permanecer no sandbox, não use sua configuração do Amazon SES para aplicativos que estão em produção. Nessa situação, o Amazon Cognito não consegue enviar mensagens para os endereços de e-mail de seus usuários.

Para removê-lo Conta da AWS da sandbox, consulte Como sair da sandbox do Amazon SES no Guia do desenvolvedor do Amazon Simple Email Service.

Etapa 3: conceder permissões de e-mail ao Amazon Cognito

Talvez você precise conceder permissões específicas ao Amazon Cognito para que ele possa enviar e-mails aos usuários. As permissões que você concede e o processo usado para concedê-las dependem de você estar usando a configuração de e-mail padrão ou a configuração do Amazon SES.

Para conceder permissões para usar a configuração de e-mail padrão

Conclua esta etapa somente se você configurar seu grupo de usuários para Enviar e-mail com o Cognito ou definir `EmailSendingAccount` como `COGNITO_DEFAULT`.

Com a configuração de e-mail padrão, seu grupo de usuários pode enviar mensagens de e-mail com qualquer um dos seguintes endereços:

- O endereço padrão `no-reply@verificationemail.com`.
- Um endereço FROM personalizado de seus endereços ou domínios de e-mail verificados no Amazon SES.

Se você usar um endereço personalizado, o Amazon Cognito precisará de permissões adicionais para enviar e-mail aos usuários a partir desse endereço. Essas permissões são concedidas por uma política de autorização de envio para o endereço ou domínio no Amazon SES. Se você usar o console do Amazon Cognito para adicionar um endereço personalizado ao grupo de usuários,

a política será anexada automaticamente ao endereço de e-mail verificado do Amazon SES. No entanto, se você configurar seu grupo de usuários fora do console, como usar a API AWS CLI ou a API do Amazon Cognito, deverá anexar a política usando o console do Amazon SES ou a PutIdentityPolicyAPI.

> **Note**
>
> Você só pode configurar um endereço FROM (Remetente) em um domínio verificado usando a AWS CLI ou a API do Amazon Cognito.

Uma política de autorização de envio permite ou nega o acesso com base nos recursos da conta que estão usando o Amazon Cognito para invocar o Amazon SES. Para obter mais informações sobre políticas baseadas em recursos, consulte o Manual do usuário do IAM. Você também pode encontrar exemplos de políticas baseadas em recursos no Guia do desenvolvedor do Amazon SES.

Example Política de autorização de envio

O exemplo de política de autorização de envio a seguir concede ao Amazon Cognito uma capacidade limitada de usar uma identidade verificada do Amazon SES. O Amazon Cognito só pode enviar mensagens de e-mail quando o fizer em nome do grupo de usuários na condição `aws:SourceArn` e da conta na condição `aws:SourceAccount`.

Regions with Amazon SES

Sua política de autorização de envio na região do grupo de usuários ou na região alternativa deve permitir que a entidade principal do serviço Amazon Cognito envie mensagens de e-mail. Para obter mais informações, consulte a tabela de regiões. Se sua Região do grupo de usuários corresponder a pelo menos um valor na Região do Amazon SES, configure sua política de autorização de envio com a entidade principal de serviço global no exemplo a seguir.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "stmnt1234567891234",
            "Effect": "Allow",
```

```
            "Principal": {
                "Service": [
                    "email.cognito-idp.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "arn:aws:ses:us-
east-1:111122223333:identity/support@example.com",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:cognito-idp:us-
east-1:111122223333:userpool/us-east-1_EXAMPLE"
                }
            }
        }
    ]
}
```

Opt-in Regions without Amazon SES

O Amazon SES não está disponível em todas as opções em que o Amazon Cognito Regiões da AWS está disponível. O Oriente Médio (EAU) é um exemplo e só pode enviar e-mails com identidades verificadas na Europa (Frankfurt) (`eu-central-1`). Em grupos de usuários com a configuração de e-mail padrão, o Amazon Cognito também envia mensagens de e-mail com uma identidade verificada em cada uma das duas regiões. No caso do Oriente Médio (EAU), a região adicional é Europa (Londres). Você deve atualizar a política de autorização de envio nas duas regiões.

Sua política de autorização de envio em cada região alternativa deve permitir que a entidade principal do serviço Amazon Cognito na opção de região do grupo de usuários envie mensagens de e-mail. Para obter mais informações, consulte a tabela de regiões. Se sua região estiver marcada como Região alternativa, configure suas políticas de autorização de envio com a entidade principal de serviço regional, como no exemplo a seguir. Substitua o identificador de região *`me-central-1`* de exemplo pelo ID de região necessário, conforme necessário.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "cognito-idp.me-central-1.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "arn:aws:ses:us-
east-1:111122223333:identity/support@example.com",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:cognito-idp:us-
east-1:111122223333:userpool/us-east-1_EXAMPLE"
                }
            }
        }
    ]
}
```

Para mais informações sobre sintaxe de políticas, consulte Políticas de autorização de envio do Amazon SES no Guia do desenvolvedor do Amazon Simple Email Service.

Para mais exemplos, consulte Exemplos de política de autorização de envio do Amazon SES no Guia do desenvolvedor do Amazon Simple Email Service.

Para conceder permissões para usar sua configuração do Amazon SES

Se você configurar o grupo de usuários para usar a configuração do Amazon SES, o Amazon Cognito precisará de permissões adicionais para chamar o Amazon SES em seu nome quando ele enviar e-mails aos usuários. Essa autorização é concedida com o serviço do IAM.

Quando você configura o grupo de usuários com essa opção, o Amazon Cognito cria uma função vinculada ao serviço, que é um tipo de função do IAM, em sua Conta da AWS. Essa função contém as permissões para que o Amazon Cognito acesse o Amazon SES e envie mensagens de e-mail com seu endereço.

O Amazon Cognito cria sua função vinculada ao serviço com as AWS credenciais da sessão do usuário que define a configuração. As permissões do IAM dessa sessão devem incluir a ação `iam:CreateServiceLinkedRole`. Para obter mais informações sobre permissões no IAM, consulte Gerenciamento de acesso para AWS recursos no Guia do usuário do IAM.

Para obter mais informações sobre a função vinculada ao serviço criada pelo Amazon Cognito, consulte Como usar funções vinculadas a serviço para o Amazon Cognito.

Etapa 4: configurar o grupo de usuários

Execute as etapas a seguir para configurar o grupo de usuários com qualquer um dos seguintes:

- Um endereço FROM personalizado exibido como remetente de e-mail
- Um endereço REPLY-TO personalizado que recebe as mensagens que os usuários enviam ao endereço FROM
- Sua configuração do Amazon SES

> **Note**
>
> Se a identidade verificada for um endereço de e-mail, ele será definido pelo Amazon Cognito como o endereço de e-mail FROM e REPLY-TO por padrão. Porém, se a identidade verificada for um domínio, você deverá fornecer um valor para o endereço de e-mail FROM.

Ignore este procedimento se quiser usar a configuração de e-mail e endereço padrão do Amazon Cognito.

Configurar o grupo de usuários para usar um endereço de e-mail personalizado

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Métodos de autenticação, localize Configuração de e-mail e selecione Editar.
5. Na página Edit email configuration (Editar configuração do e-mail), selecione Send email from Amazon SES (Enviar e-mail do Amazon SES) ou Send email with Amazon Cognito (Enviar e-mail com o Amazon Cognito). Só é possível personalizar a SES Region (Região SES), o Configuration Set (Conjunto de configurações) e o FROM sender name (Nome do remetente) quando você seleciona Send email from Amazon SES (Enviar e-mail do Amazon SES).
6. Para usar um endereço FROM (remetente) personalizado, conclua as seguintes etapas:

   a. Em SES Region (Região do SES), escolha a região que contém seu endereço de e-mail verificado.

   b. Em FROM email address (Endereço do e-mail remetente), escolha seu endereço de e-mail. Use um endereço de e-mail verificado com o Amazon SES.

   c. (Opcional) Em Configuration set (Conjunto de configurações), escolha um conjunto de configurações a ser usado pelo Amazon SES. Criar e salvar essa alteração cria uma função vinculada ao serviço.

   d. (Opcional) Em FROM sender address (Endereço do remetente), insira um endereço de e-mail. Você pode fornecer apenas um endereço de e-mail ou um endereço de e-mail e um nome amigável no formato `Jane Doe <janedoe@example.com>`.

   e. (Opcional) Em REPLY-TO email address (Endereço de e-mail para resposta), insira o endereço de e-mail no qual você deseja receber mensagens enviadas pelos usuários para o seu endereço FROM (Remetente).
7. Escolha Salvar alterações.

Related Topics

- Personalizar mensagens de verificação de e-mail
- Como personalizar mensagens de convite a usuários

Configurações de mensagens SMS para grupos de usuários do Amazon Cognito

Alguns eventos do Amazon Cognito para seu grupo de usuários podem fazer com que o Amazon Cognito envie mensagens de texto SMS para eles. Por exemplo, se você configurar o grupo de usuários para exigir verificação de telefone, o Amazon Cognito enviará um e-mail quando um usuário se cadastrar em uma nova conta na aplicação ou redefinir a senha. Dependendo da ação que inicia a mensagem de texto SMS, a mensagem contém um código de verificação, uma senha temporária ou uma mensagem de boas-vindas.

O Amazon Cognito usa o Amazon Simple Notification Service (Amazon SNS) para a entrega de mensagens de texto SMS. O Amazon SNS, por sua vez, entrega mensagens SMS para. AWS End User Messaging SMS Se você estiver enviando uma mensagem de texto pelo Amazon Cognito pela primeira vez, isso o AWS End User Messaging SMS colocará em um ambiente de sandbox. No ambiente de área restrita para testes, você pode testar suas aplicações para mensagens de texto SMS. No sandbox, só é possível simular o envio de mensagens.

> **Note**
>
> Em novembro de 2024, AWS substituiu as mensagens SMS do Amazon SNS por. AWS End User Messaging SMS Atualmente, o console do Amazon Cognito faz referência aos recursos do Amazon SNS. Grupos de usuários iniciam mensagens SMS com a operação Amazon SNS Publish, que é uma passagem para. AWS End User Messaging SMS Portanto, você ainda deve configurar permissões para `sns:Publish`, não para `sms-voice:SendTextMessage`.

AWS End User Messaging SMS cobranças por mensagens de texto SMS. Para obter mais informações, consulte Preços do AWS End User Messaging SMS.

O Amazon Cognito envia mensagens SMS aos usuários com um código a ser inserido. A tabela a seguir mostra os eventos que podem gerar uma mensagem SMS.

Opções de mensagem

Atividade	Operação de API	Opções de entrega	Opções de formato	Personalizável	Modelo de mensagem
Esqueci a senha	ForgotPassword, AdminResetUserPassword	E-mail, SMS	código	Sim	Mensagem de verificação
Convite	AdminCreateUser	E-mail, SMS	código	Sim	Mensagem de convite
Autorregistro	SignUp, ResendConfirmationCode	E-mail, SMS	código, link	Sim	Mensagem de verificação
Verificação de endereço de e-mail ou número de telefone	UpdateUserAttributes, AdminUpdateUserAttributes, GetUserAttributeVerificationCode	E-mail, SMS	código	Sim	Mensagem de verificação
Autenticação multifatorial (MFA)	AdminInitiateAuth, InitiateAuth	E-mail[1], SMS, aplicativo autenticador	código	Sim[2]	Mensagem de MFA
Autenticação de senha única (OTP)	AdminInitiateAuth, InitiateAuth	E-mail[1], SMS	código	Sim	Mensagem de MFA [3]

[1] Requer o plano de recursos Essentials ou superior e a configuração de e-mail do Amazon SES.

[2] Para mensagens SMS e e-mail.

[3] Você só pode personalizar o modelo de mensagem de MFA quando a MFA é obrigatória ou opcional no grupo de usuários. Quando a MFA está inativa, o Amazon Cognito envia senhas de uso único com o modelo padrão.

AWS End User Messaging SMS cobranças por mensagens SMS. Para obter mais informações, consulte Preços do AWS End User Messaging SMS.

Para saber mais sobre a MFA, consulte MFA de mensagens SMS e e-mail.

O Amazon Cognito pode impedir a entrega de mensagens adicionais de e-mail ou SMS para um único destino em um curto período. Se você acredita que seu grupo de usuários foi afetado, configure e analise os logs de erros de entrega de mensagens e entre em contato com a equipe da sua conta.

Práticas recomendadas

Devido ao volume de tráfego de SMS não solicitado ao redor do mundo, alguns governos impõem barreiras entre os remetentes e os destinatários das mensagens SMS. Ao usar mensagens SMS para MFA e atualizações de usuários, você deve tomar medidas adicionais para garantir que suas mensagens sejam entregues. Você também deve monitorar SMS-message-related as regulamentações nos países em que seus usuários possam morar e manter sua configuração de mensagens SMS atualizada. Para obter mais informações, consulte Recursos e limitações de SMS e MMS por país no Guia do usuário do AWS End User Messaging SMS .

O uso de mensagens SMS para autenticar e verificar usuários não é uma prática recomendada de segurança. Os números de telefone podem mudar de proprietário e podem não representar de maneira confiável o fator de MFA algo que você tem para seus usuários. Em vez disso, implemente a MFA TOTP na aplicação ou com um IdP de terceiros. Você também pode criar outros fatores de autenticação personalizados com Acionadores do Lambda de desafio personalizado de autenticação.

Consulte os links a seguir para obter informações sobre como proteger sua arquitetura de entrega de mensagens SMS.

- Reduce risks of user sign-up fraud and SMS pumping with Amazon Cognito user pools
- Defesa contra o bombeamento de SMS: novos AWS recursos para ajudar a combater o tráfego inflado artificialmente

Configurar mensagens SMS pela primeira vez nos grupos de usuários do Amazon Cognito

O Amazon Cognito usa o Amazon SNS, e AWS End User Messaging SMS indiretamente, para enviar mensagens SMS de seus grupos de usuários. Você também pode usar um Acionador do Lambda

de remetente personalizado de SMS para utilizar seus próprios recursos para enviar mensagens SMS. A primeira vez que você configura mensagens de texto SMS em uma determinada região Região da AWS, AWS End User Messaging SMS coloca você Conta da AWS na sandbox de SMS dessa região. AWS End User Messaging SMS usa o sandbox para evitar fraudes e abusos e para atender aos requisitos de conformidade. Quando você Conta da AWS está na sandbox, AWS End User Messaging SMS impõe algumas restrições. Por exemplo, você pode enviar mensagens de texto para até dez números de destino verificados se tiver uma identidade de origem, ou pode simular o envio de mensagens sem uma identidade de origem. Enquanto você Conta da AWS permanecer na sandbox, não envie mensagens SMS em produção. Quando você está na área restrita para testes, o Amazon Cognito não pode enviar mensagens para os números de telefone dos seus usuários.

Tópicos

- Prepare uma função do IAM que o Amazon Cognito possa usar para enviar mensagens SMS com AWS End User Messaging SMS
- Escolha o Região da AWS para mensagens SMS
- Obter uma identidade de origem para enviar mensagens SMS a números de telefone dos EUA
- Confirmar se você está na sandbox SMS
- Migrar sua conta do sandbox
- Use números de simulador ou números de telefone verificados com AWS End User Messaging SMS
- Concluir a configuração do grupo de usuários no Amazon Cognito

Prepare uma função do IAM que o Amazon Cognito possa usar para enviar mensagens SMS com AWS End User Messaging SMS

Quando você envia uma mensagem SMS de seu grupo de usuários, o Amazon Cognito assume um perfil do IAM em sua conta. O Amazon Cognito usa a permissão `sns:Publish` atribuída a esse perfil para enviar mensagens SMS aos usuários. No console do Amazon Cognito, você pode definir uma Seleção de perfil do IAM no menu Métodos de autenticação do grupo de usuários, em SMS, ou fazer essa seleção no assistente de criação do grupo de usuários.

A política de confiança do perfil do IAM de exemplo a seguir concede aos grupos de usuários do Amazon Cognito uma capacidade limitada para assumir uma função. O Amazon Cognito só pode assumir a função quando atende às seguintes condições:

- A operação assume-role está em nome do grupo de usuários na condição `aws:SourceArn`.

- A operação assume-role está em nome de um grupo de usuários na Conta da AWS definida pela condição `aws:SourceAccount`.
- A operação assume-role inclui o ID externo na condição `sts:externalId`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "cognito-idp.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "sts:ExternalId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
                    "aws:SourceAccount": "111122223333"
                },
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:cognito-idp:us-
west-2:111122223333:userpool/us-west-2_EXAMPLE"
                }
            }
        }
    ]
}
```

Você pode especificar um ARN do grupo de usuários exato ou um ARN curinga no valor da condição `aws:SourceArn`. Pesquise seus grupos ARNs de usuários no Console de gerenciamento da AWS ou com uma solicitação de DescribeUserPoolAPI.

Para enviar mensagens SMS para autenticação multifator, sua política de confiança do perfil do IAM deve ter uma condição `sts:ExternalId`. O valor dessa condição deve corresponder à `ExternalId` propriedade SmsConfigurationdo seu grupo de usuários. Quando você cria um perfil do IAM durante o processo de criação do grupo de usuários no console do Amazon Cognito, o Amazon

Cognito configura o ID externo para você no perfil e nas configurações do grupo de usuários. Isso não acontece quando você usa um perfil do IAM existente.

Você deve atualizar o `ExternalId` parâmetro do grupo de usuários em uma solicitação de UpdateUserPoolAPI e atualizar a política de confiança da função do IAM com uma `sts:externalId` condição com o mesmo valor. Para saber como usar a API para atualizar um grupo de usuários de forma a preservar a configuração original, consulte Como atualizar a configuração do grupo de usuários e do cliente da aplicação.

Para obter mais informações sobre políticas e perfis do IAM, consulte "Termos e conceitos das funções" no Guia do usuário do AWS Identity and Access Management .

Escolha o Região da AWS para mensagens SMS

> **Note**
>
> As mensagens SMS recebidas agora AWS são gerenciadas em AWS End User Messaging SMS.

Em alguns Regiões da AWS, você pode escolher a região que contém os recursos do Amazon SNS que você deseja usar para as mensagens SMS do Amazon Cognito. Em qualquer Região da AWS lugar em que o Amazon Cognito esteja disponível, exceto na Ásia-Pacífico (Seul), você pode usar os recursos do Amazon SNS no Região da AWS local em que criou seu grupo de usuários. Para tornar suas mensagens SMS mais rápidas e confiáveis quando você tiver uma opção de regiões, use os recursos do Amazon SNS na mesma região do grupo de usuários.

Escolha uma região para recursos de SMS na etapa Configurar a entrega de mensagens do novo assistente de grupo de usuários. Você também pode clicar em Editar em SMS no menu Métodos de autenticação de um grupo de usuários existente.

No lançamento, para alguns Regiões da AWS, o Amazon Cognito enviou mensagens SMS com recursos do Amazon SNS em uma região alternativa. Para definir sua região preferida, use o `SnsRegion` parâmetro do SmsConfigurationTypeobjeto para seu grupo de usuários. Quando você cria programaticamente um recurso de grupos de usuários do Amazon Cognito em uma Amazon Cognito Region (Região do Amazon Cognito) descrita na tabela a seguir e não fornece um parâmetro `SnsRegion`, seu grupo de usuários envia mensagens SMS com recursos do Amazon SNS em uma Amazon SNS Region (Região do Amazon SNS) herdada.

Os grupos de usuários do Amazon Cognito na Ásia-Pacífico (Seul) Região da AWS devem usar sua configuração do Amazon SNS na região Ásia-Pacífico (Tóquio).

O Amazon SNS (via AWS End User Messaging SMS) define a cota de gastos para todas as novas contas em 1,00 USD por mês. Você pode ter aumentado seu limite de gastos em um Região da AWS que você usa com o Amazon Cognito. Antes de alterar as Região da AWS mensagens SMS do Amazon SNS, abra um caso de aumento de cota no AWS Support Center para aumentar seu limite na nova região. Para obter mais informações, consulte Como migrar do sandbox do AWS End User Messaging SMS MMS e do Voice para a produção no Guia do AWS End User Messaging SMS usuário.

Você pode enviar mensagens SMS para qualquer região do Amazon Cognito na tabela a seguir com AWS End User Messaging SMS recursos na região de mensagens SMS correspondente.

Região do Amazon Cognito	Região de mensagens SMS
Leste dos EUA (Ohio)	Leste dos EUA (Ohio), Leste dos EUA (Norte da Virgínia)
Leste dos EUA (Norte da Virgínia)	Leste dos EUA (Norte da Virgínia)
Oeste dos EUA (N. da Califórnia)	Oeste dos EUA (N. da Califórnia)
Oeste dos EUA (Oregon)	Oeste dos EUA (Oregon)
Canadá (Central)	Canadá (Central), Leste dos EUA (Norte da Virgínia)
Oeste do Canadá (Calgary)	Oeste do Canadá (Calgary)
México (Centro)	México (Centro)
Europa (Frankfurt)	Europa (Frankfurt), Europa (Irlanda)
Europa (Londres)	Europa (Londres), Europa (Irlanda)
Europa (Irlanda)	Europa (Irlanda)
Europa (Paris)	Europa (Paris)
Europa (Estocolmo)	Europa (Estocolmo)

Região do Amazon Cognito	Região de mensagens SMS
Europa (Milão)	Europa (Milão)
Europa (Espanha)	Europa (Espanha)
Europa (Zurique)	Europa (Zurique)
Ásia-Pacífico (Malásia)	Ásia-Pacífico (Singapura)
Ásia-Pacífico (Tailândia)	Ásia-Pacífico (Mumbai)
Ásia-Pacífico (Mumbai)	Ásia-Pacífico (Mumbai), Ásia-Pacífico (Singapura)
Ásia-Pacífico (Hyderabad)	Ásia-Pacífico (Hyderabad)
Ásia-Pacífico (Hong Kong)	Ásia-Pacífico (Singapura)
Ásia-Pacífico (Seul)	Ásia-Pacífico (Tóquio)
Ásia-Pacífico (Singapura)	Ásia-Pacífico (Singapura)
Ásia-Pacífico (Sydney)	Ásia-Pacífico (Sydney)
Ásia-Pacífico (Tóquio)	Ásia-Pacífico (Tóquio)
Ásia-Pacífico (Jacarta)	Ásia-Pacífico (Jacarta)
Ásia-Pacífico (Osaka)	Asia Pacific (Osaka)
Ásia-Pacífico (Melbourne)	Ásia-Pacífico (Melbourne)
Oriente Médio (Bahrein)	Oriente Médio (Bahrein)
Oriente Médio (Emirados Árabes Unidos)	Oriente Médio (Emirados Árabes Unidos)
América do Sul (São Paulo)	América do Sul (São Paulo)
Israel (Tel Aviv)	Israel (Tel Aviv)
África (Cidade do Cabo)	África (Cidade do Cabo)

Obter uma identidade de origem para enviar mensagens SMS a números de telefone dos EUA

Se você pretende enviar mensagens de texto SMS para números de telefone dos EUA, deve obter uma identidade de origem, independentemente de criar um ambiente de área restrita para testes de SMS ou de um ambiente de produção.

As operadoras dos EUA exigem uma identidade de origem para o envio de mensagens para números de telefone dos EUA. Se você ainda não tiver uma identidade de origem, deverá obter uma. Para saber como obter uma identidade de origem, consulte Solicitar um número de telefone no Guia do usuário do AWS End User Messaging SMS .

Quando você tem mais de uma identidade de origem na mesma Região da AWS, AWS End User Messaging SMS escolhe um tipo de identidade de origem na seguinte ordem de prioridade: código curto, 10DLC, número gratuito. Não é possível alterar essa prioridade. Para obter mais informações, consulte AWS End User Messaging SMS FAQs.

Confirmar se você está na sandbox SMS

Use o procedimento a seguir para confirmar que você está na área restrita para testes de SMS. Repita o procedimento para cada um Região da AWS em que você tenha grupos de usuários de produção do Amazon Cognito.

Revise o status de área restrita para testes de SMS no console do Amazon Cognito.

Para confirmar que você está na área restrita para testes de SMS

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas credenciais da AWS .
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Clique no menu Métodos de autenticação.
5. Na seção SMS configuration (Configuração do SMS), expanda Move to Amazon SNS production environment (Migrar para o ambiente de produção do Amazon SNS). Se sua conta estiver na área restrita para testes de SMS, você verá a seguinte mensagem:

   Configure AWS service (Serviço da AWS) dependências para concluir a configuração da mensagem SMS

   Se você não vir essa mensagem, significa que alguém já configurou mensagens SMS em sua conta. Vá para Concluir a configuração do grupo de usuários no Amazon Cognito.

6. Selecione o link do Amazon SNS em Mover para o ambiente de produção do Amazon SNS. Isso abre o console do Amazon SNS em uma nova guia.
7. Verifique se você está no ambiente da área restrita para testes. A mensagem do console indica o status do seu sandbox e Região da AWS, da seguinte forma:

   ```
   This account is in the SMS sandbox in US East (N. Virginia).
   ```

Migrar sua conta do sandbox

Para usar sua aplicação em produção, mova sua conta da área restrita para testes de SMS para a produção. Depois de configurar uma identidade de origem na Região da AWS que contém os AWS End User Messaging SMS recursos que você deseja que o Amazon Cognito use, você pode verificar os números de telefone dos EUA enquanto permanece na Conta da AWS sandbox do SMS. Quando seu ambiente está em produção, você não precisa verificar números de telefone de usuários antes de enviar mensagens SMS para eles.

Você pode criar uma solicitação para sair da sandbox a partir do AWS End User Messaging SMS console ou do console do Amazon SNS. Para obter instruções detalhadas, consulte Migrar do sandbox de SMS no Guia do usuário do AWS End User Messaging SMS .

Use números de simulador ou números de telefone verificados com AWS End User Messaging SMS

Se você tiver removido sua conta da área restrita para testes de SMS, ignore esta etapa.

Se você estiver no sandbox, mas tiver configurado um número de origem, poderá enviar mensagens para números de destino verificados. Para configurar destinos verificados, consulte Adicionar um número de telefone de destino verificado no Guia do usuário do AWS End User Messaging SMS .

Você também pode enviar mensagens com remetentes e destinos simulados. As mensagens do simulador produzem logs, mas não são enviadas pela rede da operadora. No menu Atalhos, selecione Testar o envio de SMS com o simulador de SMS. Para obter mais informações, consulte Números de telefone do simulador no Guia do usuário do AWS End User Messaging SMS .

Concluir a configuração do grupo de usuários no Amazon Cognito

Retorne para a guia do navegador em que você estava criando ou editando o grupo de usuários. Conclua o procedimento . Quando você adiciona com êxito a configuração de SMS ao seu grupo de usuários, o Amazon Cognito envia uma mensagem de teste para um número de telefone interno a fim de verificar se sua configuração funciona. O Amazon SNS cobra por toda mensagem SMS de teste.

Usar atributos de segurança de grupos de usuários do Amazon Cognito

Você pode querer proteger a aplicação contra invasão de rede, adivinhação de senhas, falsificação de identidade de usuário e cadastros e logins mal-intencionados. Sua configuração dos recursos de segurança dos grupos de usuários do Amazon Cognito pode ser um componente essencial na arquitetura de segurança. A segurança do seu aplicativo é de responsabilidade do cliente "Segurança na nuvem", conforme descrito no Modelo de Responsabilidade AWS Compartilhada. As ferramentas deste capítulo contribuem para que o design de segurança da sua aplicação esteja alinhado com essas metas.

Uma decisão importante que você deve tomar ao configurar seu grupo de usuários é permitir o cadastro e o login públicos. Algumas opções de grupos de usuários, como clientes confidenciais, criação administrativa e confirmação de usuários e grupos de usuários sem domínio, estão sujeitas, em menor grau, a ataques pela Internet. No entanto, um caso de uso comum são clientes públicos que aceitam o cadastro de qualquer pessoa na Internet e enviam todas as operações diretamente para seu grupo de usuários. Em qualquer configuração, mas especialmente no caso dessas públicas, recomendamos que você planeje e implante seu grupo de usuários com os recursos de segurança em mente. A segurança insuficiente também pode afetar sua AWS fatura quando fontes indesejadas criam novos usuários ativos ou tentam explorar usuários existentes.

A MFA e a proteção contra ameaças se aplicam aos usuários locais. IdPs Os terceiros são responsáveis pela postura de segurança dos usuários federados.

Recursos de segurança de grupos de usuários.

Autenticação multifatorial (MFA)

Solicite um código que seu grupo de usuários envie por e-mail (com o plano de recursos Essentials ou Plus) ou mensagem SMS ou de uma aplicação autenticadora para confirmar o login do grupo de usuários.

Proteção contra ameaças

Procure no cadastro indicadores de risco e aplique a MFA ou bloqueie o login. Adicione declarações e escopos personalizados para acessar tokens. Envie códigos MFA do e-mail.

AWS WAF web ACLs

Inspecione o tráfego de entrada nos endpoints do grupo de usuários e na API de autenticação em busca de atividades indesejadas nas camadas da rede e de aplicação.

Diferenciação de letras maiúsculas e minúsculas

Impeça a criação de usuários cujo endereço de e-mail ou nome de usuário preferencial seja idêntico ao de outro usuário, a não ser pela diferenciação de letras maiúsculas e minúsculas.

Deletion protection (Proteção contra exclusão)

Evite que sistemas automatizados excluam acidentalmente seus grupos de usuários. Exija confirmação adicional da exclusão do grupo de usuários no Console de gerenciamento da AWS.

Erros de existência de usuários

Proteja-se contra a divulgação de nomes de usuário e aliases existentes no grupo de usuários. Retorne um erro genérico em resposta à autenticação malsucedida, independentemente de o nome de usuário ser válido ou não.

Tópicos

Adicionar MFA a um grupo de usuários

a MFA adiciona um fator de autenticação do tipo algo que você tem ao fator algo que você sabe inicial, geralmente um nome de usuário e senha. Você pode optar por mensagens de texto SMS, e-mails ou senhas de uso único com marcação temporal (TOTP) como fatores adicionais para o login dos usuários com senhas como o fator de autenticação primário.

A autenticação multifator (MFA) aumenta a segurança dos usuários locais na aplicação. No caso de usuários federados, o Amazon Cognito delega todos os processos de autenticação ao IdP e não oferece a eles fatores de autenticação adicionais.

> **Note**
>
> Na primeira vez que um novo usuário faz login no seu aplicativo, o Amazon Cognito emite tokens OAuth 2.0, mesmo que seu grupo de usuários exija MFA. O segundo fator de

autenticação quando o usuário faz login pela primeira vez é a confirmação da mensagem de verificação que o Amazon Cognito envia a ele. Se o grupo de usuários exigir MFA, o Amazon Cognito solicitará que o usuário inscreva um fator de login adicional para ser usado durante toda tentativa de login posterior à primeira.

Com a autenticação adaptável, você pode configurar o grupo de usuários para exigir um fator de autenticação adicional em resposta a um aumento no nível de risco. Para adicionar autenticação adaptável ao grupo de usuários, consulte Segurança avançada com proteção contra ameaças.

Quando você define a MFA como `required` para um grupo de usuários, todos os usuários devem concluir a MFA para fazer login. Para fazer login, cada usuário deve configurar pelo menos um fator de MFA. Quando a MFA é exigida, você deve incluir a configuração da MFA na integração dos usuários para que seu grupo de usuários permita que eles façam login.

O login gerenciado solicita que os usuários configurem a MFA quando você a define como obrigatória. Quando você define a MFA como opcional no grupo de usuários, o login gerenciado não a solicita aos usuários. Para trabalhar com a MFA opcional, você deve criar uma interface na aplicação que solicite que os usuários selecionem se desejam configurar a MFA e, depois, oriente-os durante as entradas da API para verificar o fator adicional de login.

Tópicos

Informações importantes sobre a MFA de grupo de usuários

Antes de configurar a MFA, considere o seguinte:

- Os usuários podem ter MFA ou fazer login sem senha, com uma exceção: chaves de acesso com verificação de usuário podem atender aos requisitos de MFA quando você configura em seu grupo de usuários. `FactorConfiguration` `MULTI_FACTOR_WITH_USER_VERIFICATION` `WebAuthnConfiguration`

- Você não pode definir o MFA como obrigatório em grupos de usuários que oferecem suporte a senhas de uso único.
- Você não pode adicionar `EMAIL_OTP` ou aumentar `AllowedFirstAuthFactors` quando `SMS_OTP` a MFA é necessária em seu grupo de usuários. Você pode adicionar `WEB_AUTHN` quando `FactorConfiguration` está definido como`MULTI_FACTOR_WITH_USER_VERIFICATION`.
- O login baseado em opções só oferece fatores `PASSWORD` e `PASSWORD_SRP` em todos os clientes da aplicação quando a MFA é necessária no grupo de usuários. Para obter mais informações sobre fluxos de nome de usuário e senha, consulte Fazer login com senhas persistentes e Fazer login com senhas persistentes e carga útil segura no capítulo Autenticação deste guia.
- Em grupos de usuários em que a MFA é opcional, os usuários que configuraram um fator de MFA só podem entrar com fluxos de autenticação de nome de usuário e senha no login baseado em opções. Esses usuários são elegíveis para todos os fluxos de login baseado no cliente.

A tabela a seguir descreve o efeito das configurações de MFA do grupo de usuários e da configuração dos fatores de MFA pelo usuário na capacidade de os usuários fazerem login com fatores sem senha.

Configuração de MFA do grupo de usuários	Status de MFA do usuário	WebAuthn/OTP disponível	Solicitada a MFA após o login com senha	Pode entrar com WebAuthn /OTP
Obrigatório	Configured	Não	Sim	Não
Obrigatório	Não configurado	Não	Não (não é possível fazer login)	Não
Opcional	Configured	Consegue configurar WebAuthn , mas não consegue entrar com a chave de acesso	Sim	Não

Configuração de MFA do grupo de usuários	Status de MFA do usuário	WebAuthn/OTP disponível	Solicitada a MFA após o login com senha	Pode entrar com WebAuthn /OTP
Opcional	Não configurado	Sim	Não	Sim
Opcional (com a chave de acesso MFA ativada)	Chave de acesso MFA configurada	Sim	Sim (após o login com senha)	Sim (a chave de acesso com verificação do usuário satisfaz o MFA)
Obrigatório (com a chave de acesso MFA ativada)	Chave de acesso MFA configurada	Sim	Sim (após o login com senha)	Sim (a chave de acesso com verificação do usuário satisfaz o MFA)
Desativado	Any	Sim	Não	Sim

- O método de MFA preferido de um usuário influencia os métodos que ele pode usar para recuperar a senha. Os usuários cujo MFA preferencial é por mensagem de e-mail não podem receber um código de redefinição de senha por e-mail. Os usuários cujo MFA preferencial é por mensagem SMS não podem receber um código de redefinição de senha por SMS.

  Suas configurações de recuperação de senha devem fornecer uma opção alternativa quando os usuários não estão qualificados para usar o método de redefinição de senha de sua preferência. Por exemplo, seus mecanismos de recuperação podem ter o e-mail como prioridade e o MFA do e-mail pode ser opcional no seu grupo de usuários. Nesse caso, adicione a recuperação da conta de mensagens SMS como uma segunda opção ou use operações administrativas da API para redefinir as senhas desses usuários.

  O Amazon Cognito responde às solicitações de redefinição de senha de usuários que não têm um método de recuperação válido com uma resposta de erro `InvalidParameterException`.

  O exemplo de corpo da solicitação UpdateUserPoolilustra um exemplo `AccountRecoverySetting` em que os usuários podem voltar à recuperação por mensagem SMS quando a redefinição de senha da mensagem de e-mail não está disponível.

- Os usuários não podem receber códigos de redefinição de senha e de MFA no mesmo endereço de e-mail ou número de telefone. Se eles usarem senhas de uso único (OTPs) de mensagens de e-mail para MFA, deverão usar mensagens SMS para recuperação da conta. Se OTPs usarem mensagens SMS para MFA, deverão usar mensagens de e-mail para recuperação da conta. Em grupos de usuários com MFA, talvez os usuários não consigam concluir a recuperação de senha por autoatendimento se tiverem o endereço de e-mail cadastrado, mas não tiverem o número de telefone, ou vice-versa.

  Para evitar que os usuários não consigam redefinir as senhas em grupos de usuários com essa configuração, defina os atributos `email` e `phone_number` como obrigatórios. Como alternativa, é possível configurar processos que sempre coletam e definem esses atributos quando os usuários se cadastram ou quando seus administradores criam perfis de usuário. Quando os usuários possuem ambos os atributos, o Amazon Cognito envia automaticamente códigos de redefinição de senha para o destino que não é o fator de MFA do usuário.
- Ao ativar a MFA em seu grupo de usuários e escolher Mensagem de texto SMS ou Mensagem de e-mail como um segundo fator, você pode enviar mensagens para um atributo de número de telefone ou e-mail que você não verificou no Amazon Cognito. Depois que o usuário conclui a MFA, o Amazon Cognito define o atributo `phone_number_verified` ou `email_verified` como `true`.
- Depois de cinco tentativas malsucedidas de apresentar um código de MFA, o Amazon Cognito inicia o processo de bloqueio de tempo limite exponencial descrito em Comportamento de bloqueio em tentativas fracassadas de login.
- Se sua conta estiver na sandbox de SMS Região da AWS que contém os recursos do Amazon Simple Notification Service (Amazon SNS) para seu grupo de usuários, você deve verificar os números de telefone no Amazon SNS antes de enviar uma mensagem SMS. Para obter mais informações, consulte Configurações de mensagens SMS para grupos de usuários do Amazon Cognito.
- Para alterar o status da MFA dos usuários em resposta aos eventos detectados com proteção contra ameaças, ative a MFA e a defina como opcional no console do grupo de usuários do Amazon Cognito. Para obter mais informações, consulte Segurança avançada com proteção contra ameaças.
- As mensagens de e-mail e SMS exigem que seus usuários tenham atributos de endereço de e-mail e número de telefone, respectivamente. Você pode definir `email` ou `phone_number` como atributos obrigatórios do seu grupo de usuários. Nesse caso, os usuários não podem concluir o cadastro a menos que informem um número de telefone. Se você não definir esses atributos como obrigatórios, mas quiser usar a MFA para e-mail ou SMS, solicite aos usuários o endereço de e-

mail ou número de telefone quando eles se cadastrarem. Como prática recomendada, configure seu grupo de usuários para enviar mensagens automáticas aos usuários para verificar esses atributos.

O Amazon Cognito considera um número de telefone ou endereço de e-mail como verificado se um usuário recebeu com sucesso um código temporário por SMS ou mensagem de e-mail e devolveu esse código em uma solicitação de VerifyUserAttributeAPI. Como alternativa, sua equipe pode definir números de telefone e marcá-los como verificados com um aplicativo administrativo que realiza solicitações de AdminUpdateUserAttributesAPI.

- Se você definiu a MFA como obrigatória e ativou mais de um fator de autenticação, o Amazon Cognito solicitará que novos os usuários selecionem um fator de MFA que queiram usar. Os usuários devem ter um número de telefone para configurar a MFA de mensagens SMS e um endereço de e-mail para configurar a MFA de mensagens de e-mail. Se um usuário não tiver o atributo definido para nenhuma MFA baseada em mensagem disponível, o Amazon Cognito solicitará que ele configure a MFA TOTP. A solicitação para escolher um fator de MFA (`SELECT_MFA_TYPE`) e configurar um fator escolhido (`MFA_SETUP`) surge como uma resposta desafiadora às operações InitiateAuthde AdminInitiateAuthAPI.

Preferências de MFA do usuário

Os usuários podem configurar vários fatores de MFA. Apenas um valor pode estar ativo. Você pode escolher a preferência efetiva de MFA para seus usuários nas configurações do grupo de usuários ou nas solicitações do usuário. Um grupo de usuários solicita que o usuário forneça códigos de MFA quando as configurações do grupo de usuários e suas próprias configurações em nível de usuário atendem às seguintes condições:

1. Você define a MFA como opcional ou obrigatória em seu grupo de usuários.
2. O usuário tem um atributo `phone_number` ou `email` válido ou configurou uma aplicação autenticadora para TOTP.
3. Pelo menos um fator de MFA está ativo.
4. Um fator de MFA é definido como preferencial.

Evite o uso do mesmo fator para login e MFA

É possível configurar seu grupo de usuários de forma que um fator de login seja a única opção de login e MFA disponível para alguns ou todos os usuários. Esse resultado pode ocorrer quando seu

principal caso de uso de login são senhas de uso único por mensagem de e-mail ou mensagem SMS (). OTPs O MFA preferido de um usuário pode ser o mesmo tipo de fator de seu login nas seguintes condições:

- O MFA é necessário no grupo de usuários.
- O OTP por e-mail e SMS estão disponíveis nas opções de login e MFA no grupo de usuários.
- O usuário faz login com e-mail ou mensagem SMS OTP.
- Eles têm um atributo de endereço de e-mail, mas nenhum atributo de número de telefone, ou um atributo de número de telefone, mas nenhum atributo de endereço de e-mail.

Nesse cenário, o usuário pode entrar com uma OTP de e-mail e concluir o MFA com uma OTP de e-mail. Essa opção cancela a função essencial do MFA. Os usuários que fazem login com senhas de uso único devem poder usar métodos de entrega diferentes para fazer login e para MFA. Quando os usuários têm opções de SMS e e-mail, o Amazon Cognito atribui automaticamente um fator diferente. Por exemplo, quando um usuário faz login com OTP de e-mail, seu MFA preferido é SMS OTP.

Siga as etapas a seguir para abordar a autenticação do mesmo fator quando seu grupo de usuários oferece suporte à autenticação OTP para login e MFA.

1. Ative o OTP por e-mail e SMS como fatores de login.
2. Ative o OTP por e-mail e SMS como fatores de MFA.
3. Coletar

Configurações do grupo de usuários e seus efeitos nas opções de MFA

A configuração do seu grupo de usuários influencia os métodos de MFA que os usuários podem escolher. A seguir estão algumas configurações do grupo de usuários que influenciam a capacidade dos usuários de configurar a MFA.

- Na configuração Autenticação multifator no menu Login do console do Amazon Cognito, você pode definir a MFA como opcional ou obrigatória, ou desativá-la. O equivalente de API dessa configuração é o MfaConfigurationparâmetro de `CreateUserPool`, `UpdateUserPool`, `SetUserPoolMfaConfig` e.

  Além disso, na configuração de Autenticação multifator, a configuração de métodos de MFA determina os fatores de MFA que os usuários podem configurar. O equivalente da API a essa configuração é a SetUserPoolMfaConfigoperação.

- No menu Login, em Recuperação de conta de usuário, você pode configurar como seu grupo de usuários envia mensagens aos usuários que esquecem a senha. O método de MFA de um usuário não pode ter o mesmo método de entrega de MFA do grupo de usuários para códigos de senha esquecida. O parâmetro da API para o método de entrega de senha esquecida é o AccountRecoverySettingparâmetro de e. `CreateUserPool` `UpdateUserPool`

  Por exemplo, usuários não podem configurar a MFA do e-mail quando a opção de recuperação é Somente e-mail. Isso ocorre porque você não pode habilitar a MFA do e-mail e definir a opção de recuperação como Somente e-mail no mesmo grupo de usuários. Quando você define essa opção como E-mail se disponível, ou SMS, o e-mail é a opção prioritária de recuperação, mas seu grupo de usuários pode recorrer à mensagem SMS quando um usuário não está qualificado para a recuperação de mensagens de e-mail. Nesse cenário, os usuários podem definir a MFA do e-mail como preferencial e só podem receber uma mensagem SMS quando tentarem redefinir sua senha.
- Se você definir apenas um método de MFA como disponível, não precisará gerenciar as preferências de MFA do usuário.
- Uma configuração ativa de SMS torna automaticamente as mensagens SMS um método de MFA disponível em seu grupo de usuários.

  Uma configuração de e-mail ativa com seus próprios recursos do Amazon SES em um grupo de usuários e o plano de recursos Essentials ou Plus torna automaticamente as mensagens de e-mail um método de MFA disponível em seu grupo de usuários.
- Quando você define a MFA como obrigatória em um grupo de usuários, os usuários não podem habilitar ou desabilitar nenhum método de MFA. Você só pode definir um método preferencial.
- Quando você define a MFA como opcional em um grupo de usuários, o login gerenciado não solicita que os usuários configurem a MFA, mas solicita que os usuários forneçam um código de MFA quando têm um método de MFA preferencial.
- Quando você ativa a proteção contra ameaças e configura respostas de autenticação adaptativa no modo de função completa, a MFA deve ser opcional em seu grupo de usuários. Uma das opções de resposta com a autenticação adaptativa é exigir MFA para um usuário cuja tentativa de login é avaliada como contendo um nível de risco.

  A configuração Atributos obrigatórios no menu Cadastrar-se do console determina se os usuários devem fornecer um endereço de e-mail ou número de telefone para se cadastrar na aplicação. Mensagens de e-mail e SMS se tornam fatores elegíveis de MFA quando um usuário tem o atributo correspondente. O parâmetro Schema de `CreateUserPool` define os atributos como obrigatórios.

- Quando você define a MFA como obrigatória em um grupo de usuários e um usuário faz login com o login gerenciado, o Amazon Cognito solicita que ele selecione um método de MFA dentre os métodos disponíveis para seu grupo de usuários. O login gerenciado trata da coleta de um endereço de e-mail ou um número de telefone e da configuração de TOTP. O diagrama a seguir demonstra a lógica por trás das opções que o Amazon Cognito apresenta aos usuários.

Configurar preferências de MFA para usuários

Você pode configurar as preferências de MFA para usuários em um modelo de autoatendimento com autorização de token de acesso ou em um modelo gerenciado pelo administrador com operações administrativas de API. Essas operações ativam ou desativam os métodos de MFA e definem um dos vários métodos como a opção preferencial. Depois que o usuário definir uma preferência de MFA, o Amazon Cognito solicitará que ele forneça um código do método de MFA preferencial no momento do login. Os usuários que não definiram uma preferência recebem uma solicitação para escolher um método preferencial em um desafio `SELECT_MFA_TYPE`.

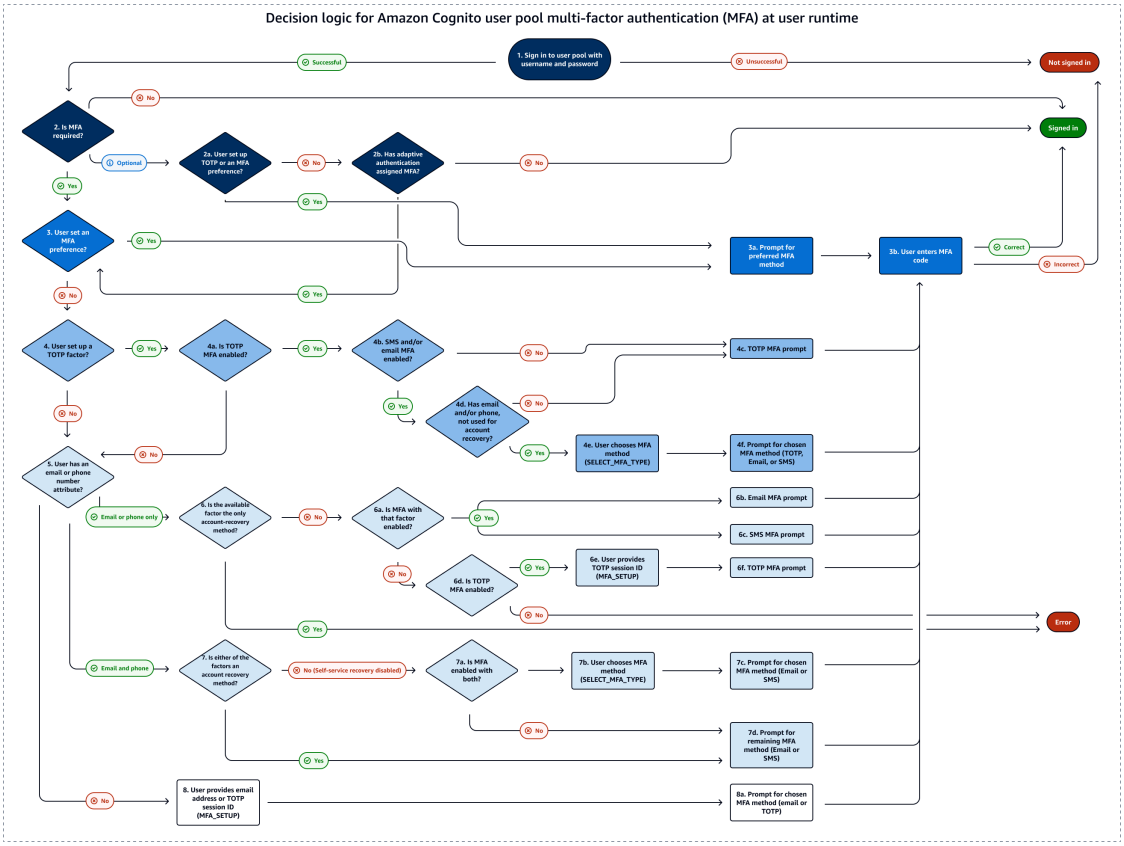
- Em um modelo de autoatendimento do usuário ou aplicativo público SetUserMfaPreference, autorizado com um token de acesso do usuário conectado, define a configuração da MFA.
- Em um aplicativo confidencial ou gerenciado pelo administrador, autorizado com AWS credenciais administrativas AdminSetUserPreference, define a configuração da MFA.

Você também pode definir as preferências de MFA do usuário no menu Usuários do console do Amazon Cognito. Para obter mais informações sobre os modelos de autenticação pública e confidencial na API de grupos de usuários do Amazon Cognito, consulte Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado.

Detalhes da lógica de MFA no runtime do usuário

Para determinar as etapas a serem tomadas quando os usuários fazem login, seu grupo de usuários avalia as preferências de MFA do usuário, os atributos do usuário, a configuração de MFA do grupo de usuários, as ações de proteção contra ameaças e as configurações de recuperação de contas de autoatendimento. Em seguida, ele conecta os usuários, solicita que eles escolham um método de MFA, solicita que configurem um método de MFA ou solicita a MFA. Para configurar um método de MFA, os usuários devem fornecer um endereço de e-mail ou número de telefone ou registrar um autenticador TOTP. Eles também podem configurar opções de MFA e registrar uma opção preferida com antecedência. O diagrama a seguir lista os efeitos detalhados da configuração do grupo de usuários nas tentativas de login imediatamente após a inscrição inicial.

A lógica ilustrada aqui se aplica às aplicações baseados em SDK e ao login gerenciado, mas é menos visível no login gerenciado. Ao solucionar problemas de MFA, retroceda dos resultados dos usuários para as configurações do perfil do usuário e do grupo de usuários que contribuíram para a decisão.



A lista a seguir corresponde à numeração no diagrama lógico de decisão e descreve cada etapa em detalhes. Um

indica uma autenticação bem-sucedida e a conclusão do fluxo. Um

indica uma autenticação malsucedida.

1. Um usuário apresenta o nome de usuário ou nome de usuário e senha na tela de login. Se ele não apresentar credenciais válidas, a solicitação de login será negada.
2. Se ele conseguir a autenticação por nome de usuário e senha, determine se a MFA será obrigatória, opcional ou desativada. Se ela estiver desativada, o

nome de usuário e a senha corretos resultarão em autenticação bem-sucedida.

a. Se a MFA for opcional, determine se o usuário configurou anteriormente um autenticador TOTP. Se ele tiver configurado o TOTP, solicite a MFA com TOTP. Se ele responder com sucesso ao desafio de MFA, o login será bem-sucedido.

b. Determine se o recurso de autenticação adaptável da proteção contra ameaças exigiu que o usuário configurasse a MFA. Se ele não tiver atribuído a MFA, o login será bem-sucedido.

3. Se a MFA for necessária ou a autenticação adaptativa tiver atribuído a MFA, determine se o usuário definiu um fator de MFA como habilitado e preferencial. Se ele tiver, solicite a MFA com esse fator. Se ele responder com sucesso ao desafio de MFA, o login será bem-sucedido.

4. Se o usuário não tiver definido uma preferência de MFA, determine se ele registrou um autenticador TOTP.

   a. Se o usuário registrou um autenticador TOTP, determine se a MFA com TOTP está disponível no grupo de usuários (a MFA com TOTP pode ser desabilitada após os usuários terem configurado previamente os autenticadores).

   b. Determine se a MFA por mensagem SMS ou de e-mail também está disponível no grupo de usuários.

   c. Se nem a MFA do e-mail nem SMS estiverem disponíveis, solicite ao usuário a MFA com TOTP. Se ele responder com sucesso ao desafio de MFA, o login será bem-sucedido.

   d. Se a MFA do e-mail ou SMS estiver disponível, determine se o usuário tem o atributo `email` ou `phone_number` correspondente. Nesse caso, qualquer atributo que não seja o método principal de recuperação de contas de autoatendimento e esteja habilitado para MFA estará disponível para ele.

   e. Solicite ao usuário um desafio `SELECT_MFA_TYPE` com opções `MFAS_CAN_SELECT` que incluem TOTP e os fatores de MFA disponíveis por SMS ou e-mail.

f. Solicite ao usuário o fator que ele selecionou em resposta ao desafio `SELECT_MFA_TYPE`. Se ele responder com sucesso ao desafio de MFA, o login será bem-sucedido.

5. Se o usuário não registrou um autenticador TOTP, ou se o fez, mas a MFA com TOTP está atualmente desabilitada, determine se o usuário tem um atributo `email` ou `phone_number`.
6. Se o usuário tiver somente um endereço de e-mail ou somente um número de telefone, determine se esse atributo também é o método que o grupo de usuários implementa para enviar mensagens de recuperação de conta para redefinição de senha. Nesse caso, ele não conseguirá concluir o login com a exigência de MFA e o Amazon Cognito retornará um erro. Para ativar o login desse usuário, você deverá adicionar um atributo de não recuperação ou registrar um autenticador TOTP para ele.

a. Se ele tiver um endereço de e-mail ou número de telefone de não recuperação disponível, determine se o fator MFA do e-mail ou SMS correspondente está habilitado.

b. Se ele tiver um atributo de endereço de e-mail de não recuperação e a MFA do e-mail estiver habilitada, solicite um desafio `EMAIL_OTP`. Se ele responder com sucesso ao desafio de MFA, o login será bem-sucedido.

c. Se ele tiver um atributo de número de telefone de não recuperação e a MFA do SMS estiver habilitada, solicite um desafio `SMS_MFA`. Se ele responder com sucesso ao desafio de MFA, o login será bem-sucedido.

d. Se ele não tiver um atributo qualificado para um fator de MFA do e-mail ou SMS habilitado, determine se a MFA com TOTP está habilitada. Se a MFA com TOTP estiver desabilitada, ele não conseguirá concluir o login com a exigência de MFA e o Amazon Cognito retornará um erro. Para ativar o login desse usuário, você deverá adicionar um atributo de não recuperação ou registrar um autenticador TOTP para ele.

> **Note**
>
> Essa etapa já foi avaliada como Não se o usuário tiver um autenticador TOTP, mas a MFA com TOTP estiver desabilitada.

e. Se a MFA com TOTP estiver habilitada, apresente ao usuário um desafio `MFA_SETUP` com `SOFTWARE_TOKEN_MFA` nas opções `MFAS_CAN_SETUP`. Para concluir esse desafio, você deve registrar separadamente um autenticador TOTP para o usuário e responder com `"ChallengeName": "MFA_SETUP", "ChallengeResponses": {"USERNAME": "[username]", "SESSION": "[Session ID from VerifySoftwareToken]}"`.

f. Depois que o usuário responder ao `MFA_SETUP` desafio com o token de sessão de uma VerifySoftwareTokensolicitação, solicite a ele um `SOFTWARE_TOKEN_MFA` desafio. Se ele responder com sucesso ao desafio de MFA, o login será bem-sucedido.

7. Se o usuário tiver um endereço de e-mail e um número de telefone, determine qual atributo, se houver, é o principal método para mensagens de recuperação de conta para redefinição de senha.

   a. Se a recuperação de conta de autoatendimento estiver desabilitada, qualquer um dos atributos poderá ser usado para a MFA. Determine se um ou ambos os fatores de MFA do e-mail e SMS estão habilitados.

   b. Se ambos os atributos estiverem habilitados como um fator de MFA, solicite ao usuário um desafio `SELECT_MFA_TYPE` com as opções `MFAS_CAN_SELECT` `SMS_MFA` e `EMAIL_OTP`.

   c. Solicite ao usuário o fator que ele selecionou em resposta ao desafio `SELECT_MFA_TYPE`. Se ele responder com sucesso ao desafio de MFA, o login será bem-sucedido.

   d. Se somente um atributo for um fator de MFA elegível, solicite que ele responda a um desafio para o fator restante. Se ele responder com sucesso ao desafio de MFA, o login será bem-sucedido.

   Esse resultado acontece nos cenários a seguir.

   i. Quando ele tem os atributos `email` e `phone_number`, a MFA do SMS e e-mail estão habilitadas, e o principal método de recuperação da conta é por e-mail ou mensagem SMS.

   ii. Quando ele tem os atributos `email` e `phone_number`, somente a MFA do SMS ou a MFA do e-mail está habilitada e a recuperação de conta de autoatendimento está desabilitada.

8. Se o usuário não tiver registrado um autenticador TOTP e não tiver um atributo `email` nem `phone_number`, solicite a ele um desafio `MFA_SETUP`. A lista em `MFAS_CAN_SETUP` inclui todos os fatores de MFA habilitados no grupo de usuários que não são a principal opção de recuperação de conta. Ele pode responder a esse desafio com `ChallengeResponses` para MFA do e-mail ou com TOTP. Para configurar a MFA do SMS, adicione um atributo de número de telefone separadamente e reinicie a autenticação.

   Para a MFA com TOTP, responda com `"ChallengeName": "MFA_SETUP", "ChallengeResponses": {"USERNAME": "[username]", "SESSION": "[Session ID from VerifySoftwareToken]"}`.

   Para a MFA do e-mail, responda com `"ChallengeName": "MFA_SETUP", "ChallengeResponses": {"USERNAME": "[username]", "email": "[user's email address]"}`.

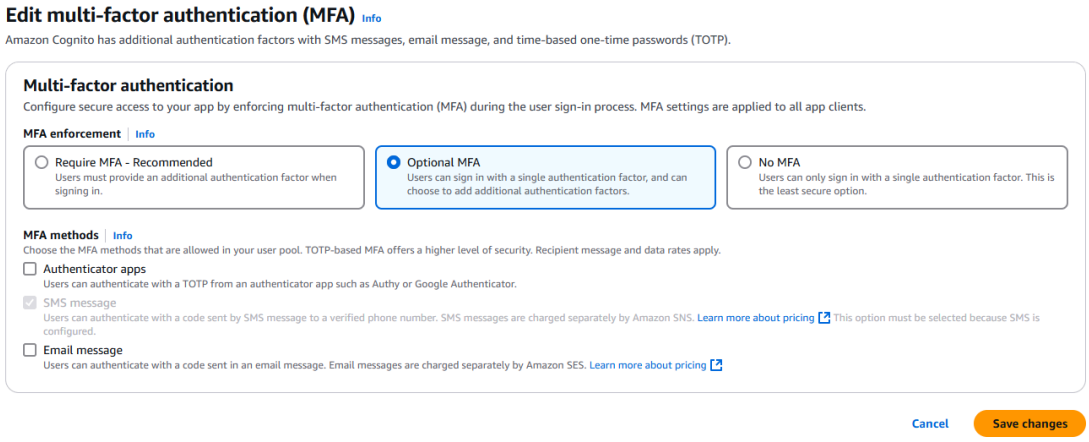
   a. Solicite ao usuário o fator que ele selecionou em resposta ao desafio `SELECT_MFA_TYPE`. Se ele responder com sucesso ao desafio de MFA, o login será bem-sucedido.

Configurar um grupo de usuários para a autenticação multifator

Você pode configurar o MFA no console do Amazon Cognito ou com a operação SetUserPoolMfaConfigda API e os métodos do SDK.

Para configurar MFA no console do Amazon Cognito

1. Faça login no console do Amazon Cognito.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Fazer login. Encontre Autenticação multifator e selecione Editar.
5. Escolha o método MFA enforcement (Aplicação de MFA) que você deseja usar com o grupo de usuários.



a. Solicite a MFA. Todos os usuários do grupo de usuários devem fazer login com um código adicional de SMS, e-mail ou senha de uso único com marcação temporal (TOTP) como um fator de autenticação adicional.

b. MFA opcional. MFA opcional: é possível oferecer aos usuários a opção de cadastrar um fator adicional de acesso e ainda permitir o acesso por usuários sem MFA configurada. Se você usar a autenticação adaptativa, escolha essa opção. Para obter mais informações sobre autenticação adaptativa, consulte Segurança avançada com proteção contra ameaças.

c. Sem MFA. Os usuários não podem registrar um fator adicional de login.

6. Escolha os MFA methods (Métodos de MFA) que você aceitará em sua aplicação. Você pode definir Mensagem de e-mail, Mensagem SMS ou Aplicações autenticadoras geradoras de TOTP como segundo fator.

7. Se usar mensagens de texto SMS como segundo fator e não tiver uma função do IAM configurada para usar com o Amazon Simple Notification Service(Amazon SNS) para mensagens de SMS, você poderá criar uma no console. No menu Métodos de autenticação do seu grupo de usuários, localize SMS e selecione Editar. Você também pode usar uma função existente que permita que o Amazon Cognito envie mensagens SMS aos usuários por você. Para obter mais informações, consulte Perfis do IAM.

   Se usar mensagens de e-mail como segundo fator e não tiver uma identidade de origem configurada para usar com o Amazon Simple Email Service (Amazon SES) para mensagens de e-mail, você poderá criar uma no console. Você deve escolher a opção Enviar e-mail com SES. No menu Métodos de autenticação do seu grupo de usuários, localize E-mail e selecione Editar. Selecione um Endereço de e-mail do remetente entre as identidades verificadas disponíveis na lista. Se você escolher um domínio verificado, por exemplo `example.com`,

também deverá configurar um Nome do remetente no domínio verificado, por exemplo `admin-noreply@example.com`.

8. Escolha Salvar alterações.

MFA de mensagens SMS e e-mail

As mensagens de MFA por SMS e e-mail confirmam que os usuários têm acesso a um destino de mensagem antes de poderem fazer login. Elas confirmam tanto o acesso a uma senha como às mensagens SMS ou à caixa de entrada de e-mail do usuário original. O Amazon Cognito solicita que os usuários informem um código curto que seu grupo de usuários envia após fornecerem com sucesso um nome de usuário e uma senha.

A MFA por SMS e e-mail não requer configuração adicional depois que o usuário adiciona um endereço de e-mail ou um número de telefone ao perfil. O Amazon Cognito pode enviar mensagens para endereços de e-mail e números de telefone não verificados. Quando um usuário conclui sua primeira MFA, o Amazon Cognito marca seu endereço de e-mail ou número de telefone como verificado.

A autenticação de MFA começa quando um usuário com MFA insere seu nome de usuário e senha na aplicação. Seu aplicativo envia esses parâmetros iniciais em um método SDK que invoca uma solicitação de API ou. InitiateAuthAdminInitiateAuth Os `ChallengeParameters` na resposta da API incluem um valor `CODE_DELIVERY_DESTINATION` que indica para onde o código de autorização foi enviado. Na aplicação, exiba um formulário que solicite que o usuário verifique o telefone e inclua um elemento de entrada para o código. Quando ele inserir o código, envie-o em uma solicitação de API de desafio-resposta para concluir o processo de login.

Depois que o usuário com MFA faz login com nome de usuário e senha nas páginas do login gerenciado, ele automaticamente precisa fornecer o código de MFA.

Os grupos de usuários enviam mensagens SMS para a MFA e outras notificações do Amazon Cognito com os recursos do Amazon Simple Notification Service (Amazon SNS) na Conta da AWS. Da mesma forma, grupos de usuários enviam mensagens de e-mail com os recursos do Amazon Simple Email Service (Amazon SES) em sua conta. Esses serviços vinculados incorrem em seus próprios custos em sua AWS fatura de entrega de mensagens. Eles também têm requisitos adicionais para enviar mensagens em volumes de produção. Para obter mais informações, consulte os seguintes links:

- Configurações de mensagens SMS para grupos de usuários do Amazon Cognito

- Preços de SMS no mundo
- Configurações de e-mail para grupos de usuários do Amazon Cognito
- Definição de preços do Amazon SES

Considerações sobre MFA para SMS e mensagens de e-mail

- Para permitir que os usuários façam login com a MFA do e-mail, seu grupo de usuários deve ter as seguintes opções de configuração:
  1. Você tem o plano de recursos Plus ou Essentials em seu grupo de usuários. Para obter mais informações, consulte Planos de recursos de grupos de usuários.
  2. Seu grupo de usuários envia mensagens de e-mail com seus próprios recursos do Amazon SES. Para obter mais informações, consulte Configuração de e-mail do Amazon SES.
- O código de MFA é válido para a Duração da sessão de fluxo de autenticação que você definiu para o cliente da aplicação.

  Defina a duração de uma sessão de fluxo de autenticação no console do Amazon Cognito na guia Clientes da aplicação ao Editar o cliente da aplicação. Você também pode definir a duração da sessão do fluxo de autenticação em uma solicitação de API `CreateUserPoolClient` ou `UpdateUserPoolClient`. Para obter mais informações, consulte Um exemplo de sessão de autenticação.
- Quando um usuário envia corretamente um código de uma mensagem SMS ou de e-mail que o Amazon Cognito enviou para um número de telefone ou endereço de e-mail não verificado, o Amazon Cognito marca o atributo correspondente como verificado.
- Para que um usuário faça uma alteração por autoatendimento no valor de um número de telefone ou endereço de e-mail associado à MFA, ele deve entrar e autorizar a solicitação com um token de acesso. Se não conseguir acessar o número de telefone ou endereço de e-mail atual, ele não conseguirá fazer login. Sua equipe deve alterar esses valores com AWS as credenciais de administrador nas solicitações de AdminUpdateUserAttributesAPI.
- Depois de configurar o SMS em seu grupo de usuários, você não pode desativar as mensagens SMS como um fator de MFA disponível.

MFA de token de software TOTP

Quando você configura a MFA de token de software TOTP no grupo de usuários, o usuário faz login com um nome de usuário e senha e usa uma TOTP para concluir a autenticação. Depois que o

usuário definir e verificar um nome de usuário e uma senha, ele poderá ativar um token de software TOTP para MFA. Se a sua aplicação usar o login gerenciado do Amazon Cognito para fazer login de usuários, o usuário enviará o nome de usuário e a senha e enviará a senha TOTP em uma página de login adicional.

Você pode ativar a MFA com TOTP para seu grupo de usuários no console do Amazon Cognito ou usar as operações da API do Amazon Cognito. No nível do grupo de usuários, você pode ligar SetUserPoolMfaConfigpara configurar o MFA e habilitar o TOTP MFA.

> **Note**
>
> Se a MFA de token do software TOTP não estiver habilitada para o grupo de usuários, o Amazon Cognito não poderá usar o token para associar nem verificar usuários. Nesse caso, os usuários recebem uma exceção `SoftwareTokenMFANotFoundException` com a descrição `Software Token MFA has not been enabled by the userPool`. Se você desativar a MFA do token de software mais tarde para o grupo de usuários, os usuários que já tiverem associado e verificado um token TOTP poderão continuar a usá-lo para a MFA.

A configuração da TOTP do usuário é um processo de várias etapas no qual o usuário recebe um código secreto que é validado com a digitação de uma senha de uso único. Em seguida, você pode ativar a MFA da TOTP para o usuário ou definir a TOTP como método de MFA preferencial para o seu usuário.

Quando você configura seu grupo de usuários para exigir a MFA com TOTP e os usuários se cadastram em sua aplicação no login gerenciado, o Amazon Cognito automatiza o processo do usuário. O Amazon Cognito solicita que o usuário selecione um método de MFA, exibe um código QR para configurar a aplicação autenticadora e verifica o registro de MFA. Em grupos de usuários em que você permitiu a escolha entre MFA por SMS e TOTP, o Amazon Cognito também oferece ao usuário uma opção de método.

> **Important**
>
> Quando você tem uma ACL AWS WAF da web associada a um grupo de usuários e uma regra na sua ACL da web apresenta um CAPTCHA, isso pode causar um erro irrecuperável no registro do TOTP de login gerenciado. Para criar uma regra que tenha uma ação de CAPTCHA e não afete a TOTP do login gerenciado, consulte Configurando sua ACL AWS WAF da web para login gerenciado TOTP MFA. Para obter mais informações sobre a AWS

WAF web ACLs e o Amazon Cognito, consulte. Associar uma ACL AWS WAF da web a um grupo de usuários

Para implementar o TOTP MFA em uma interface de usuário personalizada com AWS um SDK e a API de grupos de usuários do Amazon Cognito, consulte. Configurar a MFA com TOTP para um usuário

Para adicionar MFA ao grupo de usuários, consulte Adicionar MFA a um grupo de usuários.

Considerações e limitações da MFA com TOTP

1. O Amazon Cognito comporta MFA de token de software por meio de uma aplicação autenticadora que gera códigos TOTP. O Amazon Cognito não comporta MFA baseada em hardware.
2. Quando seu grupo de usuários requer uma TOTP para um usuário que não a configurou, o usuário recebe um token de acesso único que sua aplicação pode usar para ativar a MFA com TOTP para ele. Ocorrerá uma falha nas tentativas de login subsequentes enquanto o usuário não registrar um fator de login TOTP adicional.
   - O usuário que se inscreve em seu grupo de usuários com a operação de API `SignUp` ou pelo login gerenciado recebe tokens únicos ao concluir o cadastro.
   - Depois que você cria um usuário e o usuário define a senha inicial, o Amazon Cognito emite tokens únicos do login gerenciado para o usuário. Se você definir uma senha permanente para o usuário, o Amazon Cognito emitirá tokens únicos quando ele fizer login pela primeira vez.
   - O Amazon Cognito não emite tokens únicos para um usuário criado pelo administrador que faz login com as operações da API ou da API. InitiateAuthAdminInitiateAuth Depois que seu usuário tiver êxito no desafio de definir a senha inicial ou se você definir uma senha permanente para ele, o Amazon Cognito imediatamente convidará o usuário a configurar a MFA.
3. Se um usuário em um grupo de usuários que requer MFA já tiver recebido um token de acesso único, mas não tiver configurado a MFA com TOTP, ele não poderá fazer login com o login gerenciado enquanto não configurar a MFA. Em vez do token de acesso, você pode usar o valor da `session` resposta de um `MFA_SETUP` desafio para InitiateAuthou AdminInitiateAuthem uma AssociateSoftwareTokensolicitação.
4. Se os usuários tiverem configurado a TOTP, eles poderão usá-la para MFA, mesmo que, posteriormente, você a função do Lambda para o grupo de usuários.
5. O Amazon Cognito só aceita TOTPs aplicativos autenticadores que geram códigos com a função hash HMAC. SHA1 Os códigos gerados com o hash SHA-256 geram um erro `Code mismatch`.

Configurar a MFA com TOTP para um usuário

Quando um usuário faz login pela primeira vez, sua aplicação usa o token de acesso único para gerar a chave privada TOTP e apresentá-la ao usuário em formato de texto ou código QR. O usuário configura a aplicação autenticadora e fornece uma TOTP para tentativas de login subsequentes. Sua aplicação ou login gerenciado apresenta o TOTP para o Amazon Cognito nas respostas do desafio de MFA.

Em algumas circunstâncias, o login gerenciado solicita que novos usuários configurem um autenticador TOTP. Para obter mais informações, consulte Detalhes da lógica de MFA no runtime do usuário.

Tópicos

- Associar o token de software TOTP
- Verificar o token TOTP
- Faça login com MFA de TOTP
- Remover o token de TOTP

Associar o token de software TOTP

Para associar o token TOTP, envie ao usuário um código secreto que ele deve validar com uma senha única. A associação do token requer três função do Lambdas.

1. Quando seu usuário escolher o token de software TOTP MFA, ligue AssociateSoftwareTokenpara retornar um código-chave secreto compartilhado gerado exclusivo para a conta do usuário. Você pode autorizar AssociateSoftwareToken com um token de acesso ou uma string de sessão.
2. Sua aplicação apresenta ao usuário a chave privada ou um código QR gerado por meio da chave privada. Seu usuário deve inserir a chave em uma aplicação geradora de TOTP, como o Google Authenticator, digitalizando o código QR que sua aplicação gera com base na chave privada ou inserindo a chave manualmente.
3. O usuário insere a chave ou digitaliza o código QR em uma aplicação autentiticadora, como o Google Authenticator, e a aplicação começa a gerar códigos.

Verificar o token TOTP

Depois, verifique o token TOTP. Solicite códigos de exemplo de seu usuário e os forneça ao serviço Amazon Cognito para confirmar se o usuário está gerando códigos TOTP com êxito, da forma a seguir.

1. Sua aplicação solicita um código ao usuário para demonstrar que ele configurou a aplicação autenticadora corretamente.
2. A aplicação autenticadora do usuário exibe uma senha temporária. A aplicação autenticadora usa a chave secreta que você forneceu ao usuário como base para a senha.
3. O usuário insere a senha temporária. Sua aplicação transmite a senha temporária para o Amazon Cognito em uma solicitação de API `VerifySoftwareToken`.
4. O Amazon Cognito mantém a chave secreta associada ao usuário e gera uma TOTP e a compara com a que o usuário forneceu. Se elas corresponderem, o `VerifySoftwareToken` retornará uma resposta `SUCCESS`.
5. O Amazon Cognito associa o fator TOTP ao usuário.
6. Se a operação `VerifySoftwareToken` retornar uma resposta `ERROR`, verifique se o relógio do usuário está correto e se ele não excedeu o número máximo de novas tentativas. O Amazon Cognito aceita tokens TOTP 30 segundos antes ou depois da tentativa, para que haja uma distorção mínima no relógio. Depois de resolver o problema, tente a VerifySoftwareToken operação novamente.

Faça login com MFA de TOTP

Nesse ponto, o usuário faz login com a senha única baseada em tempo. O processo ocorre conforme a seguir.

1. O usuário digita o nome de usuário e a senha para fazer login em sua aplicação cliente.
2. O desafio da MFA de TOTP é invocado e o usuário é solicitado pela sua aplicação a inserir uma senha temporária.
3. O usuário obtém a senha temporária de um aplicativo gerador de TOTP associado.
4. O usuário informa o código da TOTP no seu aplicativo cliente. A aplicação notifica o serviço do Amazon Cognito para verificá-lo. Para cada login, RespondToAuthChallengedeve ser chamado para obter uma resposta ao novo desafio de autenticação TOTP.
5. Se o token for verificado pelo Amazon Cognito, o login será bem-sucedido e o usuário continuará com o fluxo de autenticação.

Remover o token de TOTP

Por fim, a aplicação deve permitir que o usuário desative a configuração do TOTP. No momento, você não poderá excluir o token de software TOTP de um usuário. Para substituir o token de software do usuário, associe e confirme um novo token de software. Para desativar o TOTP MFA para um usuário, ligue para modificar seu usuário SetUserMFAPreferencepara não usar nenhum MFA ou somente MFA por SMS.

1. Crie uma interface na aplicação para usuários que desejam redefinir a MFA. Solicite que um usuário nessa interface insira a senha.
2. Se o Amazon Cognito retornar um desafio de MFA TOTP, atualize a preferência de MFA do seu usuário com. SetUserMFAPreference
3. Na aplicação, comunique ao usuário que ele desativou a MFA e solicite que ele faça login novamente.

Configurando sua ACL AWS WAF da web para login gerenciado TOTP MFA

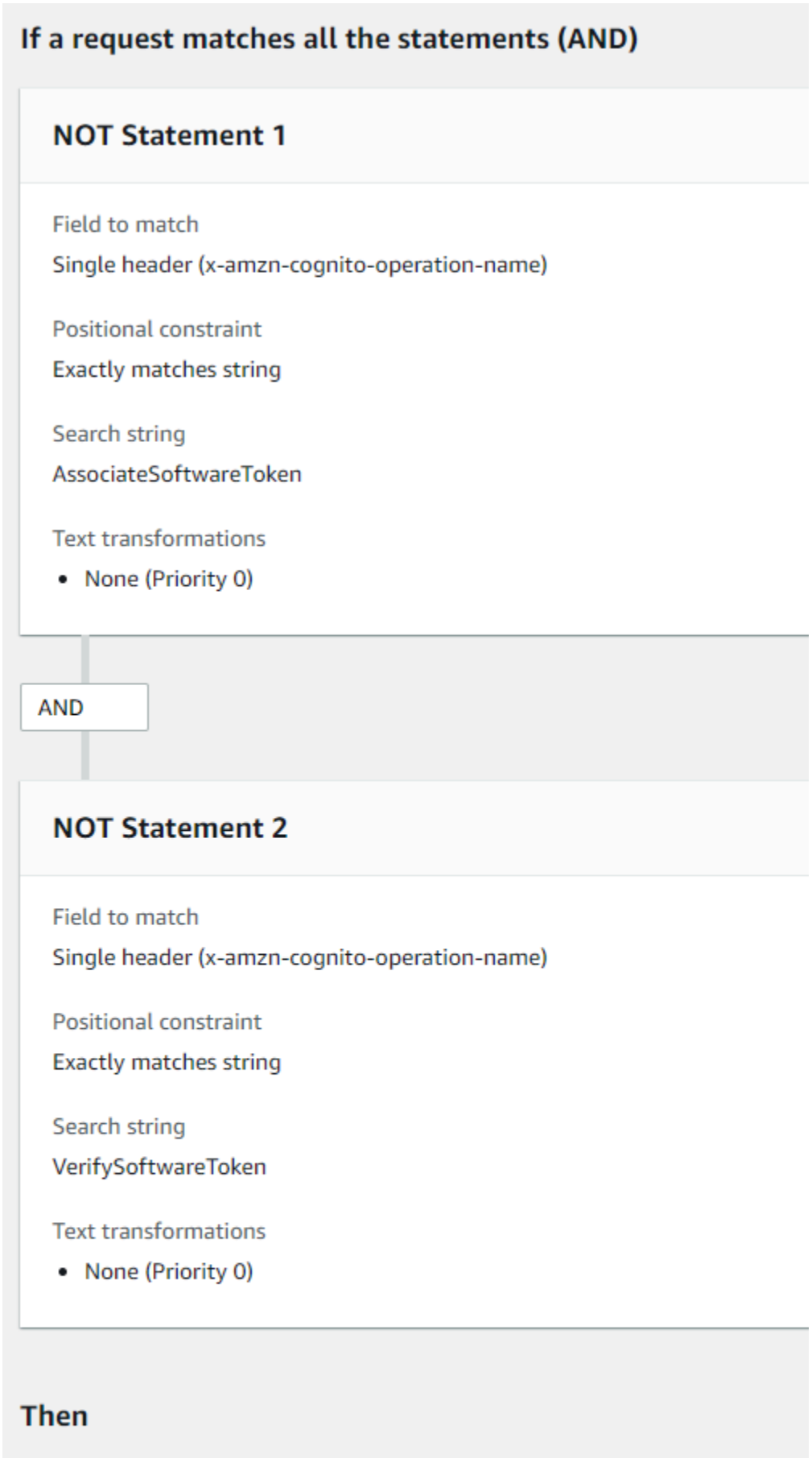
Quando você tem uma ACL AWS WAF da web associada a um grupo de usuários e uma regra na sua ACL da web apresenta um CAPTCHA, isso pode causar um erro irrecuperável no registro do TOTP de login gerenciado. AWS WAF As regras de CAPTCHA só têm esse efeito no TOTP MFA no login gerenciado e na interface de usuário hospedada clássica. A MFA por SMS não é afetada.

O Amazon Cognito exibe o erro a seguir quando a regra de CAPTCHA não permite que um usuário conclua a configuração da MFA com TOTP.

Solicitação não permitida devido ao captcha do WAF.

Esse erro ocorre quando AWS WAF solicita um CAPTCHA em resposta a AssociateSoftwareTokensolicitações de VerifySoftwareTokenAPI que seu grupo de usuários faz em segundo plano. Para criar uma regra que tenha uma ação de CAPTCHA e não afete o TOTP do login gerenciado, exclua os valores `AssociateSoftwareToken` e `VerifySoftwareToken` do cabeçalho `x-amzn-cognito-operation-name` da ação de CAPTCHA em sua regra.

A captura de tela a seguir mostra um exemplo de AWS WAF regra que aplica uma ação CAPTCHA a todas as solicitações que não têm um valor de `x-amzn-cognito-operation-name` cabeçalho de ou. `AssociateSoftwareToken` `VerifySoftwareToken`


If a request matches all the statements (AND)
NOT Statement 1
Field to match
Single header (x-amzn-cognito-operation-name)
Positional constraint
Exactly matches string
Search string
AssociateSoftwareToken
Text transformations
None (Priority 0)
AND
NOT Statement 2
Field to match
Single header (x-amzn-cognito-operation-name)
Positional constraint
Exactly matches string
Search string
VerifySoftwareToken
Text transformations
None (Priority 0)
Then
Action
The action to take when a web request matches the rule statement.

Para obter mais informações sobre a AWS WAF web ACLs e o Amazon Cognito, consulte. Associar uma ACL AWS WAF da web a um grupo de usuários

Segurança avançada com proteção contra ameaças

Depois de criar o grupo de usuários, você terá acesso à Proteção contra ameaças no menu de navegação do console do Amazon Cognito. Você pode ativar os recursos de proteção contra ameaças e personalizar as ações executadas em resposta a riscos diferentes. Outra opção é usar o modo de auditoria para coletar métricas sobre riscos detectados sem aplicar mitigação de segurança. No modo de auditoria, a proteção contra ameaças publica métricas na Amazon CloudWatch. Você verá métricas depois que o Amazon Cognito gerar o primeiro evento. Consulte Como exibir métricas de proteção contra ameaças.

A proteção contra ameaças, anteriormente chamada de recursos avançados de segurança, é um conjunto de ferramentas de monitoramento de atividades indesejadas em seu grupo de usuários e ferramentas de configuração para encerrar automaticamente atividades possivelmente mal-intencionadas. A proteção contra ameaças tem diferentes opções de configuração para operações de autenticação padrão e personalizadas. Por exemplo, você pode querer enviar uma notificação a um usuário com um login suspeito de autenticação personalizada, no qual você configurou fatores de segurança adicionais, mas bloqueou o usuário no mesmo nível de risco com a autenticação básica por nome de usuário e senha.

A proteção contra ameaças está disponível no plano de recursos Plus. Para obter mais informações, consulte Planos de recursos de grupos de usuários.

As opções de grupos de usuários a seguir são os componentes da proteção contra ameaças.

Credenciais comprometidas

> Os usuários reutilizam senhas para várias contas de usuário. O recurso de credenciais comprometidas do Amazon Cognito compila dados de vazamentos públicos de nomes de usuário e senhas e compara as credenciais de seus usuários com listas de credenciais vazadas. A detecção de credenciais comprometidas também verifica se há senhas que possam ser deduzidas com facilidade. Você pode verificar se há credenciais comprometidas em fluxos de autenticação username-and-password padrão em grupos de usuários. O Amazon Cognito não detecta credenciais comprometidas na senha remota segura (SRP) ou na autenticação personalizada.
>
> Você pode selecionar as ações do usuário que solicitam a verificação de credenciais comprometidas e a ação que você deseja que o Amazon Cognito realize em resposta.

Para eventos de login, inscrição e alteração de senha, o Amazon Cognito pode Bloquear login ou Permitir login. Nos dois casos, o Amazon Cognito gera um log de atividades do usuário. Nele, você pode encontrar mais informações sobre o evento.

Saiba mais

Trabalhar com a detecção de credenciais comprometidas

Autenticação adaptável

O Amazon Cognito pode revisar as informações de localização e dispositivo das solicitações de login dos usuários e aplicar uma resposta automática para proteger as contas de usuário no grupo de usuários contra atividades suspeitas. Você pode monitorar a atividade do usuário e automatizar as respostas aos níveis de risco detectados no nome de usuário, senha e SRP, além da autenticação personalizada.

Quando você ativa a proteção contra ameaças, o Amazon Cognito atribui uma pontuação de risco à atividade do usuário. Você pode atribuir uma resposta automática a atividades suspeitas: é possível Exigir MFA, Bloquear login ou apenas registrar os detalhes da atividade e a pontuação de risco. Você também pode enviar automaticamente mensagens de e-mail que notificam o usuário sobre a atividade suspeita para que ele possa redefinir a senha ou realizar outras ações autoguiadas.

Saiba mais

Trabalhar com autenticação adaptável

Lista de endereços IP permitidos e negados

Com a proteção contra ameaças do Amazon Cognito no Modo de função completa, você pode criar as exceções Sempre bloquear e Sempre permitir para o endereço IP. Uma sessão de um endereço IP na lista de exceções Always block (Bloquear sempre) não recebe um nível de risco por autenticação adaptativa e não pode fazer login no grupo de usuários.

O que você deve saber sobre listas de permissões e listas de bloqueio de endereços IP

- Você deve expressar Sempre bloquear e Sempre permitir no formato CIDR, por exemplo `192.0.2.0/24`, uma máscara de 24 bits ou `192.0.2.252/32`, um único endereço IP.
- Dispositivos com endereços IP em um intervalo de IP Always block não podem se inscrever ou fazer login com aplicativos de login gerenciados ou baseados em SDK, mas podem fazer login com terceiros. IdPs

- As listas Sempre permitir e Sempre bloquear não afetam a atualização do token.
- O Amazon Cognito não aplica regras de MFA de autenticação adaptável a dispositivos de um intervalo de IP Sempre permitir, mas aplica regras de credenciais comprometidas.

Exportação de log

A proteção contra ameaças registra detalhes granulares das solicitações de autenticação dos usuários em seu grupo de usuários. Esses registros apresentam avaliações de ameaças, informações do usuário e metadados da sessão, como localização e dispositivo. Você pode criar arquivos externos desses logs para retenção e análise. Os grupos de usuários do Amazon Cognito exportam registros de proteção contra ameaças para o Amazon S3 CloudWatch , o Logs e o Amazon Data Firehose. Para obter mais informações, consulte Como exibir e exportar o histórico de eventos do usuário.

Saiba mais

Exportar logs de atividade de usuários de proteção contra ameaças

Tópicos

Considerações e limitações da proteção contra ameaças

As opções de proteção contra ameaças diferem entre os fluxos de autenticação

O Amazon Cognito aceita tanto a autenticação adaptativa quanto a detecção de credenciais comprometidas com os fluxos de autenticação `USER_PASSWORD_AUTH` e

`ADMIN_USER_PASSWORD_AUTH`. Você só pode habilitar a autenticação adaptável para `USER_SRP_AUTH`. Não é possível usar a proteção contra ameaças com login federado.

Sempre bloqueie a IPs contribuição para solicitar cotas

Solicitações bloqueadas de endereços IP em uma lista de exceções Always block (Bloquear sempre) em seu grupo de usuários contribuem para as cotas de taxas de solicitação de seus grupos de usuários.

A proteção contra ameaças não aplica limites de taxa

Alguns tráfegos maliciosos têm a característica de um alto volume de solicitações, como ataques distribuídos de negação de serviço (DDoS). As classificações de risco que o Amazon Cognito aplica ao tráfego de entrada são por solicitação e não levam em conta o volume de solicitações. Solicitações individuais em um evento de alto volume podem receber uma pontuação de risco e uma resposta automática por motivos da camada de aplicação que não estão relacionados à sua função em um ataque volumétrico. Para implementar defesas contra ataques volumétricos em seus grupos de usuários, adicione web. AWS WAF ACLs Para obter mais informações, consulte Associar uma ACL AWS WAF da web a um grupo de usuários.

A proteção contra ameaças não afeta as solicitações M2M

As concessões de credenciais do cliente são destinadas à autorização machine-to-machine (M2M) sem conexão com contas de usuário. A proteção contra ameaças monitora somente contas e senhas de usuários em seu grupo de usuários. Para implementar recursos de segurança com sua atividade M2M, considere os recursos de AWS WAF monitorar as taxas e o conteúdo das solicitações. Para obter mais informações, consulte Associar uma ACL AWS WAF da web a um grupo de usuários.

Ativar a proteção contra ameaças em grupos de usuários

Amazon Cognito user pools console

Como ativar a proteção contra ameaças para um grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Se ainda não o fez, ative o plano de recursos Plus no menu Configurações.
5. Clique no menu Proteção contra ameaças e selecione Ativar.

6. Escolha Salvar alterações.

API

Defina seu plano de recursos como Plus em uma solicitação de UpdateUserPoolAPI CreateUserPoolou API. O exemplo parcial de corpo de solicitação a seguir define a proteção contra ameaças para o modo de função completa. Para ver um exemplo completo de solicitação, consulte Exemplos.

```
"UserPoolAddOns": {
      "AdvancedSecurityMode": "ENFORCED"
   }
```

Proteção contra ameaças é o termo coletivo para os recursos que monitoram as operações do usuário em busca de sinais de invasão da conta e respondem automaticamente para proteger as contas de usuários afetadas. Você pode aplicar configurações de proteção contra ameaças aos usuários quando eles fazem login com fluxos de autenticação padrão e personalizados.

A proteção contra ameaças gera registros que detalham o login, a saída e outras atividades dos usuários. Você poderá exportar esses logs para um sistema de terceiros. Para obter mais informações, consulte Como exibir e exportar o histórico de eventos do usuário.

Conceitos de aplicação da proteção contra ameaças

A proteção contra ameaças começa em um modo somente de auditoria, em que seu grupo de usuários monitora a atividade do usuário, atribui níveis de risco e gera logs. Como prática recomendada, execute no modo somente de auditoria por duas semanas ou mais antes de ativar o modo de função completa. O modo de função completa inclui um conjunto de reações automáticas às atividades de risco detectadas e senhas comprometidas. Com o modo somente de auditoria, você pode monitorar as avaliações de ameaças que o Amazon Cognito está realizando. Você também pode fornecer feedback para treinar o recurso sobre falsos positivos e negativos.

Você pode configurar a aplicação da proteção contra ameaças no nível do grupo de usuários para cobrir todos os clientes da aplicação no grupo de usuários e no nível de clientes de aplicações individuais. As configurações de proteção contra ameaças do cliente de aplicação substituem a configuração do grupo de usuários. Para configurar a proteção contra ameaças para um cliente de aplicação, navegue até as configurações do cliente de aplicação no menu Clientes da aplicação do

seu grupo de usuários no console do Amazon Cognito. Lá, você pode Usar s configurações no nível do cliente e configurar a aplicação exclusiva para o cliente de aplicação.

Além disso, você pode configurar a proteção contra ameaças separadamente para os tipos de autenticação padrão e personalizada.

Proteção contra ameaças para autenticação padrão e autenticação personalizada

As formas de configurar a proteção contra ameaças dependem do tipo de autenticação que você está fazendo em seu grupo de usuários e clientes de aplicação. Cada um dos seguintes tipos de autenticação pode ter seu próprio modo de aplicação e respostas automatizadas:

Autenticação padrão

A autenticação padrão é o gerenciamento de login, saída e senhas do usuário com fluxos de nome de usuário e senha e no login gerenciado. A proteção contra ameaças do Amazon Cognito monitora as operações em busca de indicadores de risco quando elas fazem login com o login gerenciado ou usam os seguintes parâmetros da API `AuthFlow`:

InitiateAuth

`USER_PASSWORD_AUTH`, `USER_SRP_AUTH`. O recurso de credenciais comprometidas não tem acesso às senhas no login `USER_SRP_AUTH` e não monitora nem gerencia eventos com esse fluxo.

AdminInitiateAuth

`ADMIN_USER_PASSWORD_AUTH`, `USER_SRP_AUTH`. O recurso de credenciais comprometidas não tem acesso às senhas no login `USER_SRP_AUTH` e não monitora nem gerencia eventos com esse fluxo.

Você pode definir o Modo de imposição para autenticação padrão como Somente auditoria ou Função completa. Para desabilitar o monitoramento de ameaças para autenticação padrão, defina proteção contra ameaças como Sem imposição.

Autenticação personalizada

A Autenticação personalizada é o login do usuário com acionadores personalizados do Lambda de desafio. Não é possível fazer autenticação personalizada no login gerenciado. A proteção contra ameaças do Amazon Cognito monitora as operações em busca de indicadores de risco quando elas fazem login com o parâmetro `AuthFlow` da API de `InitiateAuth` e `AdminInitiateAuth`.

Você pode definir o Modo de imposição para autenticação personalizada como Somente auditoria, Função completa ou Sem imposição. A opção Sem imposição desabilita o monitoramento de ameaças para autenticação personalizada sem afetar outros recursos da proteção contra ameaças.

Pré-requisitos de proteção contra ameaças

Antes de começar, você precisará fazer o seguinte:

- Um grupo de usuários com um cliente de aplicativo. Para obter mais informações, consulte Conceitos básicos dos grupos de usuários.
- Defina a autenticação multifator (MFA) como Optional (Opcional) no console do Amazon Cognito para usar o recurso de autenticação adaptável com base em risco. Para obter mais informações, consulte Adicionar MFA a um grupo de usuários.
- Se você estiver usando notificações por e-mail, acesse o console do Amazon SES para configurar e verificar um endereço de e-mail ou um domínio a ser usado com suas notificações. Para obter mais informações sobre o Amazon SES, consulte Verificar identidades no Amazon SES.

Configurar a proteção contra ameaças

Siga estas instruções para configurar a proteção contra ameaças do grupo de usuários.

> **Note**
>
> Para definir uma configuração diferente de proteção contra ameaças para um cliente de aplicação no console de grupos de usuários do Amazon Cognito, selecione o cliente de aplicação no menu Clientes da aplicação e escolha Usar configurações no nível do cliente.

Console de gerenciamento da AWS

Como configurar a proteção contra ameaças para um grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Proteção contra ameaças e selecione Ativar.

5. Escolha o método de proteção contra ameaças que você deseja configurar: Autenticação padrão e personalizada. Você pode definir diferentes modos de aplicação para autenticações personalizada e padrão, mas eles compartilham a configuração de respostas automatizadas no modo de Função completa.
6. Selecione Editar.
7. Escolha um Modo de aplicação. Para começar a responder imediatamente aos riscos detectados, selecione Função completa e configure as respostas automatizadas para credenciais comprometidas e autenticação adaptável. Para coletar informações em registros e entradas em nível de usuário CloudWatch, selecione Somente auditoria.

   Recomendamos manter a proteção contra ameaças no modo de auditoria por duas semanas antes de ativar as ações. Durante esse tempo, o Amazon Cognito pode aprender os padrões de uso dos usuários da aplicação, e você pode fornecer feedback de eventos para ajustar as respostas.
8. Se tiver selecionado Audit only (Somente auditoria), escolha Save changes (Salvar alterações). Se tiver selecionado Full function (Função completa):

   a. Selecione se vai executar uma ação Custom (Personalizada) ou usar Cognito defaults (Padrões do Cognito) para responder a Compromised credentials (Credenciais comprometidas) suspeitas. Os padrões do Cognito são:

      i. Detectar credenciais comprometidas ao Acessar, Cadastrar-se, e Alterar senha.

      ii. Responder a credenciais comprometidas com a ação Block sign-in (Bloquear acesso).

   b. Se tiver selecionado ações Personalizadas para Credenciais comprometidas, escolha as ações do grupo de usuários que o Amazon Cognito usará para Detecção de eventos e as Respostas a credenciais comprometidas que deseja que o Amazon Cognito adote. É possível Block sign-in (Bloquear acesso) ou Allow sign-in (Permitir acesso) com credenciais comprometidas suspeitas.

   c. Escolha como responder a tentativas maliciosas de acesso em Adaptive authentication (Autenticação adaptável). Selecione se vai executar uma ação Custom (Personalizada) ou usar Cognito defaults (Padrões do Cognito) para responder a atividades maliciosas suspeitas. Quando você seleciona Cognito defaults (Padrões do Cognito), o Amazon Cognito bloqueia o acesso em todos os níveis de risco e não notifica o usuário.

   d. Se tiver selecionado ações Custom (Personalizadas) para Adaptive authentication (Autenticação adaptável), escolha as ações de Automatic risk response (Resposta

automática a riscos) que o Amazon Cognito adotará em resposta aos riscos detectados com base no nível de gravidade. Quando você atribui uma resposta a um nível de risco, não é possível atribuir uma resposta menos restritiva a um nível de risco mais alto. Você pode atribuir as seguintes respostas aos níveis de risco:

i. Allow sign-in (Permitir acesso): não tomar nenhuma ação preventiva.

ii. Optional MFA (MFA opcional): se o usuário tiver a MFA configurada, o Amazon Cognito sempre vai exigir que o usuário forneça um fator adicional de SMS ou senha de uso único com marcação temporal (TOTP) quando fizer o acesso. Se o usuário não tiver a MFA configurada, ele poderá continuar fazendo o acesso normalmente.

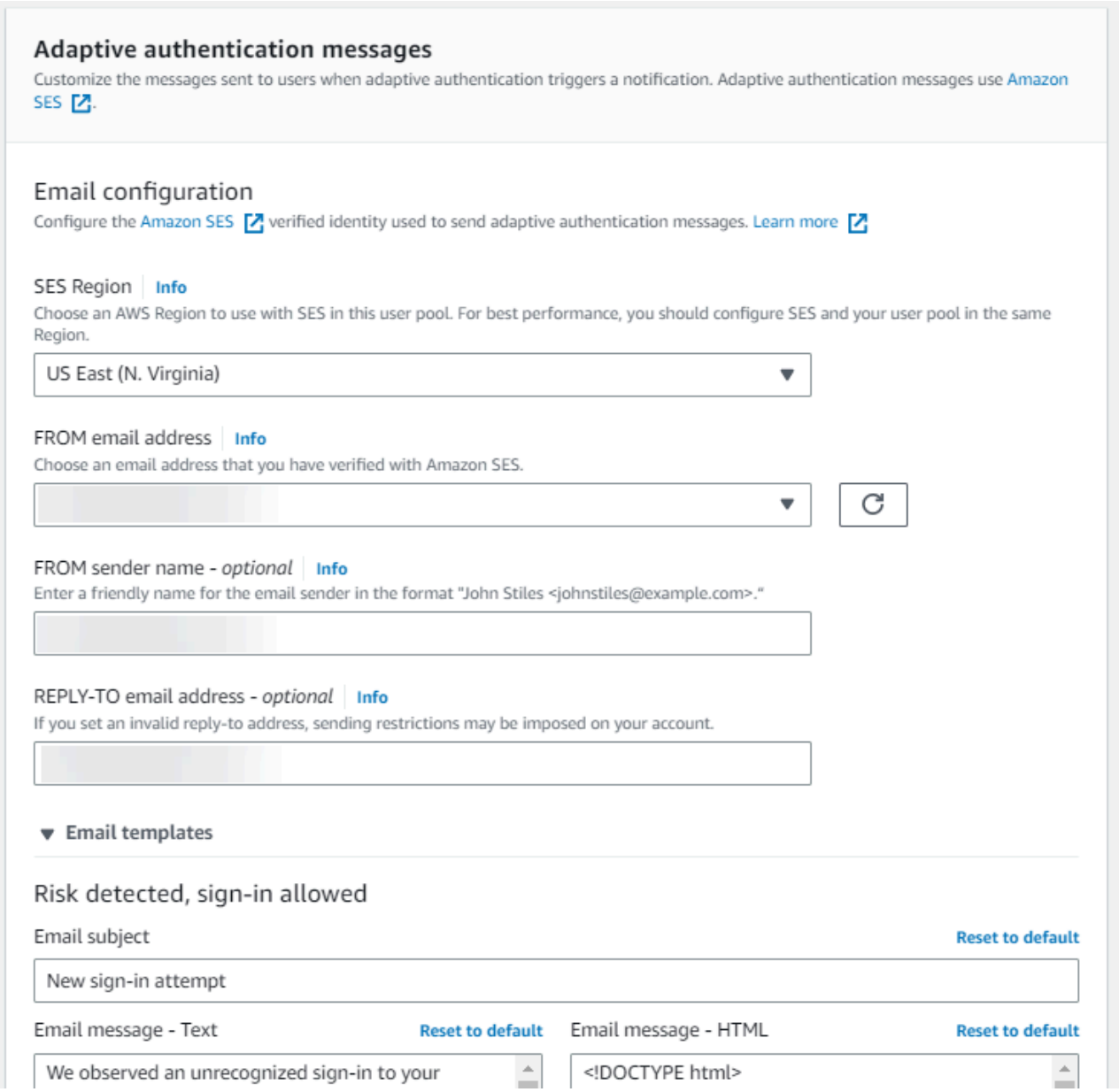
iii. Require MFA (Exigir MFA): se o usuário tiver a MFA configurada, o Amazon Cognito sempre vai exigir que o usuário forneça um fator adicional de SMS ou TOTP quando fizer o acesso. Se o usuário não tiver a MFA configurada, o Amazon Cognito solicitará que ele configure a MFA. Antes de exigir automaticamente a MFA de seus usuários, configure um mecanismo em sua aplicação para capturar números de telefone para MFA via SMS ou para registrar aplicações autenticadoras para MFA com TOTP.

iv. Block sign-in (Bloquear acesso): impedir que o usuário faça o acesso.

v. Notify user (Notificar o usuário): enviar uma mensagem de e-mail para o usuário com informações sobre o risco que o Amazon Cognito detectou e a resposta adotada. Você pode personalizar modelos de mensagem de e-mail para as mensagens enviadas.

9. Se tiver escolhido Notify user (Notificar o usuário) na etapa anterior, você pode personalizar suas configurações de entrega de e-mail e modelos de mensagem de e-mail para autenticação adaptativa.

   a. Em Configuração de e-mail, escolha os valores para Região SES, Endereço de e-mail do remetente, Nome do remetente e Endereço de e-mail do destinatário que você deseja usar com a autenticação adaptativa. Para obter mais informações sobre como integrar as mensagens de e-mail do grupo de usuários ao Amazon Simple Email Service, consulte Configurações de e-mail dos grupos de usuários do Amazon Cognito.



b. Expanda Email templates (Modelos de e-mail) para personalizar as notificações de autenticação adaptativa com as versões de mensagens de e-mail HTML e de texto simples. Para saber mais sobre modelos de mensagem de e-mail, consulte Modelos de mensagens.

10. Expanda as exceções de endereço IP para criar uma lista sempre permitida ou sempre bloqueada ou intervalos de IPv6 endereços que sempre serão permitidos IPv4 ou bloqueados, independentemente da avaliação de risco de proteção contra ameaças. Especifique os intervalos de endereços IP em CIDR notation (Notação CIDR) (por exemplo, 192.168.100.0/24).
11. Escolha Salvar alterações.

API (user pool)

Para definir a configuração de proteção contra ameaças para um grupo de usuários, envie uma solicitação de SetRiskConfigurationAPI que inclua um `UserPoolId` parâmetro, mas não um `ClientId` parâmetro. Veja a seguir um exemplo de corpo da solicitação para um grupo de usuários. Essa configuração de risco executa uma série crescente de ações com base na gravidade do risco e notifica os usuários em todos os níveis de risco. Ela aplica um bloco de credenciais comprometidas às operações de cadastro.

Para aplicar essa configuração, você deve `AdvancedSecurityMode` defini-la `ENFORCED` em uma solicitação separada CreateUserPoolou de UpdateUserPoolAPI. Para obter mais informações sobre os modelos de espaço reservado, como `{username}` neste exemplo, consulte Configurar mensagens de MFA, autenticação, verificação e convite.

```
{
   "AccountTakeoverRiskConfiguration": {
      "Actions": {
         "HighAction": {
            "EventAction": "MFA_REQUIRED",
            "Notify": true
         },
         "LowAction": {
            "EventAction": "NO_ACTION",
            "Notify": true
         },
         "MediumAction": {
            "EventAction": "MFA_IF_CONFIGURED",
            "Notify": true
         }
      },
      "NotifyConfiguration": {
         "BlockEmail": {
            "Subject": "You have been blocked for suspicious activity",
            "TextBody": "We blocked {username} at {login-time} from {ip-address}."
         },
         "From": "admin@example.com",
         "MfaEmail": {
            "Subject": "Suspicious activity detected, MFA required",
            "TextBody": "Unexpected sign-in from {username} on device {device-name}.
 You must use MFA."
         },
         "NoActionEmail": {
```

```
            "Subject": "Suspicious activity detected, secure your user account",
            "TextBody": "We noticed suspicious sign-in activity by {username} from
 {city}, {country} at {login-time}. If this was not you, reset your password."
         },
         "ReplyTo": "admin@example.com",
         "SourceArn": "arn:aws:ses:us-west-2:123456789012:identity/
admin@example.com"
      }
   },
   "CompromisedCredentialsRiskConfiguration": {
      "Actions": {
         "EventAction": "BLOCK"
      },
      "EventFilter": [ "SIGN_UP" ]
   },
   "RiskExceptionConfiguration": {
      "BlockedIPRangeList": [ "192.0.2.0/24","198.51.100.0/24" ],
      "SkippedIPRangeList": [ "203.0.113.0/24" ]
   },
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

API (app client)

Para definir a configuração de proteção contra ameaças para um cliente de aplicativo, envie uma solicitação de SetRiskConfigurationAPI que inclua um `UserPoolId` parâmetro e um `ClientId` parâmetro. Veja a seguir um exemplo de corpo da solicitação de um cliente de aplicação. Essa configuração de risco é mais severa do que a configuração do grupo de usuários, pois bloqueia entradas de alto risco. Também aplica blocos de credenciais comprometidas às operações de cadastro, login e redefinição de senha.

Para aplicar essa configuração, você deve `AdvancedSecurityMode` defini-la `ENFORCED` em uma solicitação separada CreateUserPoolou de UpdateUserPoolAPI. Para obter mais informações sobre os modelos de espaço reservado, como `{username}` neste exemplo, consulte Configurar mensagens de MFA, autenticação, verificação e convite.

```
{
   "AccountTakeoverRiskConfiguration": {
      "Actions": {
         "HighAction": {
            "EventAction": "BLOCK",
            "Notify": true
```

```
         },
         "LowAction": {
            "EventAction": "NO_ACTION",
            "Notify": true
         },
         "MediumAction": {
            "EventAction": "MFA_REQUIRED",
            "Notify": true
         }
      },
      "NotifyConfiguration": {
         "BlockEmail": {
            "Subject": "You have been blocked for suspicious activity",
            "TextBody": "We blocked {username} at {login-time} from {ip-address}."
         },
         "From": "admin@example.com",
         "MfaEmail": {
            "Subject": "Suspicious activity detected, MFA required",
            "TextBody": "Unexpected sign-in from {username} on device {device-name}.
 You must use MFA."
         },
         "NoActionEmail": {
            "Subject": "Suspicious activity detected, secure your user account",
            "TextBody": "We noticed suspicious sign-in activity by {username} from
 {city}, {country} at {login-time}. If this was not you, reset your password."
         },
         "ReplyTo": "admin@example.com",
         "SourceArn": "arn:aws:ses:us-west-2:123456789012:identity/
admin@example.com"
      }
   },
   "ClientId": "1example23456789",
   "CompromisedCredentialsRiskConfiguration": {
      "Actions": {
         "EventAction": "BLOCK"
      },
      "EventFilter": [ "SIGN_UP", "SIGN_IN", "PASSWORD_CHANGE" ]
   },
   "RiskExceptionConfiguration": {
      "BlockedIPRangeList": [ "192.0.2.1/32","192.0.2.2/32" ],
      "SkippedIPRangeList": [ "192.0.2.3/32","192.0.2.4/32" ]
   },
   "UserPoolId": "us-west-2_EXAMPLE"
```

```
}
```

Trabalhar com a detecção de credenciais comprometidas

O Amazon Cognito pode detectar se o nome de usuário e a senha de um usuário foram comprometidos em outro local. Isso pode ocorrer quando os usuários reutilizam credenciais em mais de um local ou quando usam senhas inseguras. O Amazon Cognito confere usuários locais que fazem login com nome de usuário e senha, no login gerenciado e com a API do Amazon Cognito.

No menu Proteção contra ameaças do console do Amazon Cognito, você pode configurar Credenciais comprometidas. Configure Event detection (Detecção de eventos) para escolher os eventos do usuário que você deseja monitorar em relação a credenciais comprometidas. Configure Compromised credentials responses (Respostas de credenciais comprometidas) para escolher se deseja permitir ou bloquear o usuário se forem detectadas credenciais comprometidas. O Amazon Cognito pode conferir a existência de credenciais comprometidas durante o login, o cadastro e as alterações de senha.

Ao escolher Permitir login, você pode revisar os Amazon CloudWatch Logs para monitorar as avaliações que o Amazon Cognito faz em eventos de usuários. Para obter mais informações, consulte Como exibir métricas de proteção contra ameaças. Ao escolher Block sign-in (Bloquear login), o Amazon Cognito impede o login dos usuários que usam credenciais comprometidas. Quando o Amazon Cognito bloqueia o login de um usuário, ele define o `UserStatus` do usuário como `RESET_REQUIRED`. Um usuário com o status `RESET_REQUIRED` precisa alterar a senha para poder fazer login novamente.

As credenciais comprometidas podem verificar as senhas da atividade do usuário a seguir.

Cadastrar-se

Seu grupo de usuários verifica as senhas que os usuários transmitem na SignUpoperação e na página de inscrição do login gerenciado em busca de indicadores de comprometimento.

Fazer login

Seu grupo de usuários verifica as senhas que os usuários enviam no login baseado em senha em busca de indicadores de comprometimento. O Amazon Cognito pode analisar o `ADMIN_USER_PASSWORD_AUTH` fluxo de AdminInitiateAuthentrada, o `USER_PASSWORD_AUTH` fluxo de InitiateAuthentrada e a `PASSWORD` opção do `USER_AUTH` fluxo em ambos.

No momento, o Amazon Cognito não confere credenciais comprometidas para operações de login com o fluxo de Secure Remote Password (SRP). O SRP envia uma prova de senha com hash durante o login. Com o Amazon Cognito não tem acesso às senhas internamente, ele só pode avaliar uma senha que seu cliente transmite para ele em texto simples.

Redefinição de senhas

Seu grupo de usuários verifica os indicadores de comprometimento nas operações que definem novas senhas de usuário com a operação de redefinição ConfirmForgotPasswordde senha de autoatendimento. O código necessário para essa operação é gerado por ForgotPasswordAdminResetUserPassworde.

As credenciais comprometidas não verificam as senhas temporárias ou permanentes definidas pelo administrador definidas com. AdminSetUserPassword No entanto, com senhas temporárias, seu grupo de usuários verifica as senhas a partir das respostas ao `NEW_PASSWORD_REQUIRED` desafio em RespondToAuthChallengeAdminRespondToAuthChallengee.

Para adicionar proteções contra credenciais comprometidas ao grupo de usuários, consulte Segurança avançada com proteção contra ameaças.

Trabalhar com autenticação adaptável

Com a autenticação adaptável, você pode configurar o grupo de usuários para bloquear logins suspeitos ou exigir a autenticação de segundo fator em resposta a um aumento no nível de risco. Para cada tentativa de login, o Amazon Cognito gera uma pontuação de risco para a probabilidade da solicitação de login ser de uma fonte comprometida. Essa pontuação de risco é baseada em fatores de dispositivo e usuário que sua aplicação fornece e outros que o Amazon Cognito extrai da solicitação. Alguns fatores que contribuem para a avaliação de risco pelo Amazon Cognito são o endereço IP, o agente do usuário e a distância geográfica de outras tentativas de login. A autenticação adaptativa pode ativar ou exigir a autenticação multifator (MFA) para um usuário em seu grupo de usuários quando o Amazon Cognito detecta riscos na sessão de um usuário e o usuário ainda não selecionou um método de MFA. Quando você ativa a MFA para um usuário, ele sempre recebe o desafio de fornecer ou configurar um segundo fator durante a autenticação, independentemente de como você configurou a autenticação adaptativa. Do ponto de vista do usuário, a aplicação oferece ajuda para configurar a MFA e, opcionalmente, o Amazon Cognito impede que ele faça login novamente até que tenha configurado um fator adicional.

O Amazon Cognito publica métricas sobre tentativas de login, seus níveis de risco e desafios fracassados para a Amazon. CloudWatch Para obter mais informações, consulte Como exibir métricas de proteção contra ameaças.

Para adicionar autenticação adaptável ao grupo de usuários, consulte Segurança avançada com proteção contra ameaças.

Tópicos

- Visão geral da autenticação adaptável
- Adicionar dados de sessão e dispositivo do usuário a solicitações de API
- Como exibir e exportar o histórico de eventos do usuário
- Como fornecer feedback sobre eventos
- Como enviar mensagens de notificação

Visão geral da autenticação adaptável

No menu Proteção contra ameaças do console do Amazon Cognito, você pode escolher as configurações de autenticação adaptável, incluindo as ações que serão executadas em diferentes níveis de risco e a personalização de mensagens de notificação que serão enviadas aos usuários. É possível atribuir uma configuração de proteção contra ameaças a todos os seus clientes de aplicações, mas aplicar uma configuração no nível de cliente a clientes de aplicações individuais.

A autenticação adaptativa do Amazon Cognito atribui um dos seguintes níveis de risco a cada sessão do usuário: Alto, Médio, Baixo ou Sem risco.

Considere suas opções com cuidado ao alterar seu Enforcement method (método de aplicação) de Audit-only (Somente auditoria) para Full-function (Função completa). As respostas automáticas que você aplica aos níveis de risco influenciam o nível de risco que o Amazon Cognito atribui às sessões de usuário subsequentes com as mesmas características. Por exemplo, depois de optar por não realizar nenhuma ação ou permitir (Allow) sessões de usuário que o Amazon Cognito inicialmente avalia como de alto risco, o Amazon Cognito considera que sessões semelhantes têm um risco menor.

Para cada nível de risco, você pode escolher as seguintes opções:

Opção	Ação
Permitir	Os usuários podem fazer login sem um fator adicional.
MFA opcional	Os usuários que tiverem um segundo fator configurado deverão concluir um segundo desafio de fator para fazer login. Um número de telefone para SMS e um token de software TOTP são o segundo fator disponível. Usuários sem um segundo fator configurado podem fazer login apenas com um conjunto de credenciais.
Solicitar MFA	Os usuários que tiverem um segundo fator configurado deverão concluir um segundo desafio de fator para fazer login. O Amazon Cognito bloqueia o login para usuários que não têm um segundo fator configurado.
Bloquear	O Amazon Cognito bloqueia todas as tentativas de login no nível de risco designado.

> **Note**
>
> Não é necessário confirmar os números de telefone para usá-los para SMS como segundo fator de autenticação.

Adicionar dados de sessão e dispositivo do usuário a solicitações de API

Você pode coletar e transmitir informações sobre a sessão do usuário à proteção contra ameaças do Amazon Cognito ao usar a API para inscrevê-lo, fazer seu login e redefinir sua senha. Essas informações incluem o endereço IP do usuário e um identificador de dispositivo exclusivo.

É possível ter um dispositivo de rede intermediário entre seus usuários e o Amazon Cognito, como um serviço proxy ou um servidor de aplicações. Você pode coletar dados de contexto dos usuários e transmiti-los ao Amazon Cognito para que a autenticação adaptativa calcule seu risco com base nas características do endpoint do usuário, em vez de seu servidor ou proxy. Se a aplicação do lado do cliente chamar as operações da API do Amazon Cognito diretamente, a autenticação adaptativa registrará automaticamente o endereço IP de origem. No entanto, outras informações sobre o dispositivo não serão registradas, como o `user-agent`, a menos que você também colha uma impressão digital do dispositivo.

Gere esses dados com a biblioteca de coleta de dados de contexto do Amazon Cognito e envie-os para a proteção contra ameaças do Amazon Cognito com ContextDataos parâmetros e. UserContextData A biblioteca de coleta de dados de contexto está incluída no AWS SDKs. Para obter mais informações, consulte Integração da autenticação e autorização do Amazon Cognito com aplicações móveis e da web. Você pode enviar `ContextData` se tiver o plano de recursos Plus. Para obter mais informações, consulte Configurar a proteção contra ameaças.

Ao chamar essas operações de API autenticadas do Amazon Cognito do seu servidor de aplicações, transmita o IP do dispositivo do usuário no parâmetro `ContextData`. Além disso, transmita o nome e o caminho do servidor, bem como os dados de impressão digital do dispositivo codificado.

- AdminInitiateAuth
- AdminRespondToAuthChallenge

Ao chamar as operações de API não autenticadas do Amazon Cognito, você pode enviar `UserContextData` à proteção contra ameaças do Amazon Cognito. Esses dados incluem uma impressão digital do dispositivo no parâmetro `EncodedData`. Você também pode enviar um parâmetro `IpAddress` em `UserContextData` se atender às seguintes condições:

- Seu grupo de usuários está no plano de recursos Plus. Para obter mais informações, consulte Planos de recursos de grupos de usuários.
- O cliente da aplicação tem um segredo do cliente. Para obter mais informações, consulte Configurações específicas da aplicação com clientes de aplicação.
- Você ativou a opção Accept additional user context data (Aceitar dados de contexto do usuário adicionais) no cliente da aplicação. Para obter mais informações, consulte Aceitar dados de contexto do usuário adicionais (Console de gerenciamento da AWS).

Sua aplicação pode preencher o parâmetro `UserContextData` com dados codificados de impressão digital e o endereço IP do dispositivo do usuário nestas operações de API não autenticadas do Amazon Cognito.

- InitiateAuth
- RespondToAuthChallenge
- SignUp
- ConfirmSignUp
- ForgotPassword
- ConfirmForgotPassword
- ResendConfirmationCode

Aceitar dados de contexto do usuário adicionais (Console de gerenciamento da AWS)

Seu grupo de usuários aceita um endereço IP em um parâmetro `UserContextData` depois que você ativa o recurso Accept additional user context data (Aceitar dados de contexto do usuário adicionais). Não será necessário ativar esse recurso se:

- Seus usuários só fazem login com operações de API autenticadas AdminInitiateAuth , como, e você usa o `ContextData` parâmetro.
- Você quiser que suas operações de API não autenticadas só enviem uma impressão digital do dispositivo, mas não um endereço IP, à proteção contra ameaças do Amazon Cognito.

Atualize o cliente da aplicação da maneira a seguir no console do Amazon Cognito para adicionar suporte para dados de contexto do usuário adicionais.

1. Faça login no console do Amazon Cognito.
2. No painel de navegação, selecione Manage your User Pools e escolha o grupo de usuários que você deseja editar.
3. Clique no menu Clientes da aplicação.
4. Escolha ou crie um cliente da aplicação. Para obter mais informações, consulte Configurar um cliente da aplicação do grupo de usuários.
5. Escolha Edit (Editar) no contêiner App client information (Informações do cliente da aplicação).

6. Em Advanced authentication settings (Configurações de autenticação avançada) do cliente da aplicação, escolha Accept additional user context data (Aceitar dados de contexto do usuário adicionais).
7. Escolha Salvar alterações.

Para configurar seu cliente de aplicativo para aceitar dados de contexto do usuário na API do Amazon Cognito, `EnablePropagateAdditionalUserContextData` defina como `true` em uma solicitação CreateUserPoolClientou UpdateUserPoolClient. Para obter informações sobre como trabalhar com a proteção contra ameaças na aplicação web ou móvel, consulte Coletar dados para proteção contra ameaças em aplicações. Quando a aplicação chamar o Amazon Cognito do servidor, colete dados de contexto do usuário no lado do cliente. Veja a seguir um exemplo que usa o método JavaScript `getData` SDK.

```
var EncodedData =
 AmazonCognitoAdvancedSecurityData.getData(username, userPoolId, clientId);
```

Quando você estiver criando sua aplicação para usar a autenticação adaptativa, é recomendável incorporar nela o SDK mais recente do Amazon Cognito. A versão mais recente do SDK coleta informações de impressão digital do dispositivo, como ID, modelo e fuso horário. Para obter mais informações sobre o Amazon Cognito SDKs, consulte Instalar um SDK de grupo de usuários.
A proteção contra ameaças do Amazon Cognito só salva e atribui uma pontuação de risco aos eventos enviados pela aplicação no formato correto. Se o Amazon Cognito retornar uma resposta de erro, verifique se sua solicitação inclui um hash secreto válido e se o `IPaddress` parâmetro é um endereço ou válido IPv4 . IPv6

Recursos de **`ContextData`** e **`UserContextData`**

- AWS Amplify SDK para Android: GetUserContextData
- AWS Amplify SDK para iOS: userContextData
- JavaScript: amazon-cognito-advanced-security-data.min.js

Como exibir e exportar o histórico de eventos do usuário

O Amazon Cognito gera um log para cada evento de autenticação de um usuário quando você habilita a proteção contra ameaças. Por padrão, você pode visualizar os registros do usuário no menu Usuários no console do Amazon Cognito ou com a operação da AdminListUserAuthEventsAPI.

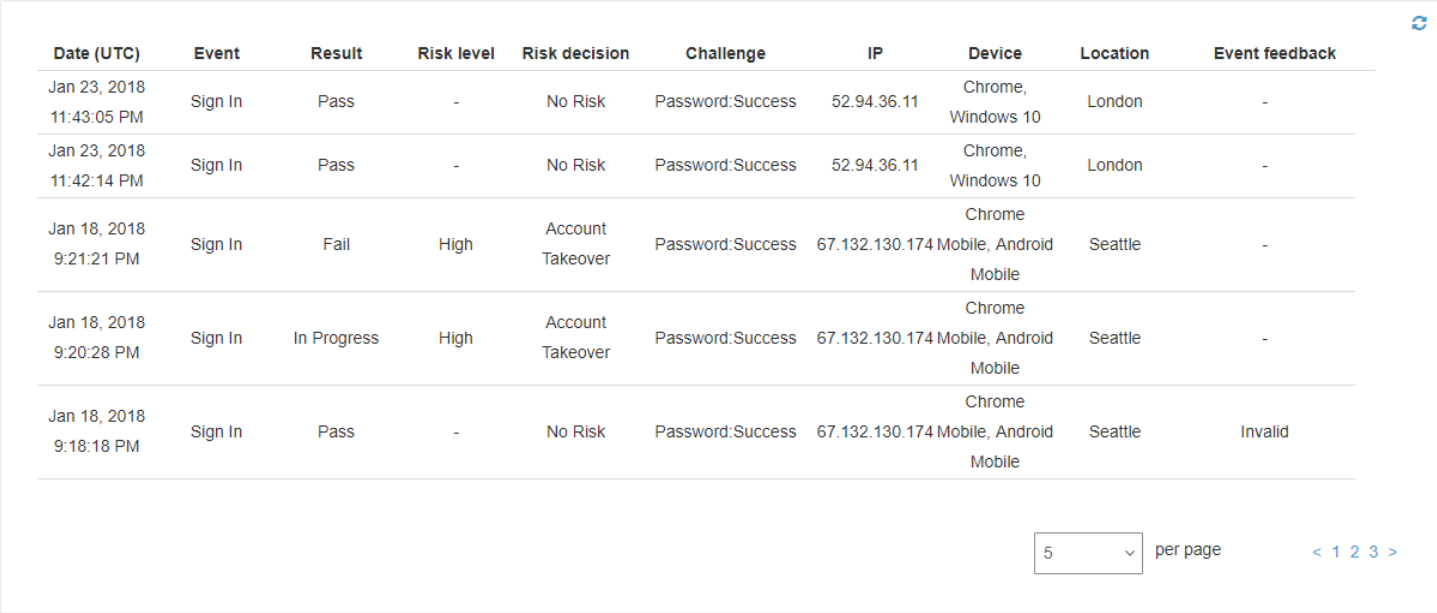
Você também pode exportar esses eventos para um sistema externo, como CloudWatch Logs, Amazon S3 ou Amazon Data Firehose. O recurso de exportação pode tornar as informações de segurança sobre a atividade do usuário em sua aplicação mais acessíveis aos seus próprios sistemas de análise de segurança.

Tópicos

- Como exibir o histórico de eventos do usuário (Console de gerenciamento da AWS)
- Como exibir o histórico de eventos do usuário (API/CLI)
- Como exportar eventos de autenticação de usuários

Como exibir o histórico de eventos do usuário (Console de gerenciamento da AWS)

Para ver o histórico de logins de um usuário, é possível selecionar o usuário no menu Usuários no console do Amazon Cognito. O Amazon Cognito mantém o histórico de eventos do usuário por dois anos.



Date (UTC)	Event	Result	Risk level	Risk decision	Challenge	IP	Device	Location	Event feedback
Jan 23, 2018 11:43:05 PM	Sign In	Pass	-	No Risk	Password:Success	52.94.36.11	Chrome, Windows 10	London	-
Jan 23, 2018 11:42:14 PM	Sign In	Pass	-	No Risk	Password:Success	52.94.36.11	Chrome, Windows 10	London	-
Jan 18, 2018 9:21:21 PM	Sign In	Fail	High	Account Takeover	Password:Success	67.132.130.174	Chrome Mobile, Android Mobile	Seattle	-
Jan 18, 2018 9:20:28 PM	Sign In	In Progress	High	Account Takeover	Password:Success	67.132.130.174	Chrome Mobile, Android Mobile	Seattle	-
Jan 18, 2018 9:18:18 PM	Sign In	Pass	-	No Risk	Password:Success	67.132.130.174	Chrome Mobile, Android Mobile	Seattle	Invalid

5 per page < 1 2 3 >

Cada evento de login tem um ID de evento. O evento também tem dados de contexto correspondentes, como localização, detalhes do dispositivo e resultados da detecção de risco.

Você também pode correlacionar o ID do evento com o token que o Amazon Cognito emitiu no momento em que gravou o evento. O ID e os tokens de acesso incluem esse ID de evento em sua carga útil. O Amazon Cognito também correlaciona o uso de token de atualização ao ID do evento original. É possível rastrear o ID do evento original de volta para o ID do evento de login que resultou na emissão de tokens do Amazon Cognito. Você pode rastrear o uso de um token em seu sistema

para determinado evento de autenticação. Para obter mais informações, consulte Compreendendo os tokens web JSON do grupo de usuários () JWTs.

Como exibir o histórico de eventos do usuário (API/CLI)

Você pode consultar o histórico de eventos do usuário com a operação da API do Amazon Cognito AdminListUserAuthEventsou com o AWS Command Line Interface (AWS CLI) com admin-list-user-auth -events.

AdminListUserAuthEvents request

O corpo da solicitação a seguir `AdminListUserAuthEvents` retorna o log de atividades mais recente de um usuário.

```
{
  "UserPoolId": "us-west-2_EXAMPLE",
  "Username": "myexampleuser",
  "MaxResults": 1
}
```

admin-list-user-auth-events request

A solicitação a seguir `admin-list-user-auth-events` retorna o log de atividades mais recente de um usuário.

```
aws cognito-idp admin-list-user-auth-events --max-results 1 --username myexampleuser
 --user-pool-id us-west-2_EXAMPLE
```

Response

O Amazon Cognito retorna o mesmo corpo de resposta JSON para as duas solicitações. Veja um exemplo de resposta para um evento de login de login gerenciado que não continha fatores de risco:

```
{
    "AuthEvents": [
        {
            "EventId": "[event ID]",
            "EventType": "SignIn",
            "CreationDate": "[Timestamp]",
```

```
            "EventResponse": "Pass",
            "EventRisk": {
                "RiskDecision": "NoRisk",
                "CompromisedCredentialsDetected": false
            },
            "ChallengeResponses": [
                {
                    "ChallengeName": "Password",
                    "ChallengeResponse": "Success"
                }
            ],
            "EventContextData": {
                "IpAddress": "192.168.2.1",
                "DeviceName": "Chrome 125, Windows 10",
                "Timezone": "-07:00",
                "City": "Bellevue",
                "Country": "United States"
            }
        }
    ],
    "NextToken": "[event ID]#[Timestamp]"
}
```

Como exportar eventos de autenticação de usuários

Configure seu grupo de usuários para exportar eventos de usuário da proteção contra ameaças para um sistema externo. Os sistemas externos compatíveis — Amazon S3, CloudWatch Logs e Amazon Data Firehose — podem adicionar custos à sua AWS fatura pelos dados que você envia ou recupera. Para obter mais informações, consulte Exportar logs de atividade de usuários de proteção contra ameaças.

Console de gerenciamento da AWS

1. Faça login no console do Amazon Cognito.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Fluxo de logs. Selecione Editar.
5. Em Status de registro em log, marque a caixa de seleção ao lado de Ativar exportação do log de atividades do usuário.

6. Em Logging destination, escolha o AWS service (Serviço da AWS) que você deseja manipular com seus registros: grupo de CloudWatch registros, stream do Amazon Data Firehose ou bucket do S3.
7. Sua seleção preencherá o seletor de recursos com o tipo de recurso correspondente. Selecione um grupo de logs, stream ou bucket na lista. Você também pode selecionar o botão Criar para ir ao Console de gerenciamento da AWS do serviço selecionado e criar um novo recurso.
8. Selecione Salvar alterações.

API

Escolha um tipo de destino para seus logs de atividades do usuário.

Veja a seguir um exemplo de corpo de solicitação `SetLogDeliveryConfiguration` que define um stream do Firehose como o destino do log.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "FirehoseConfiguration": {
            "StreamArn": "arn:aws:firehose:us-west-2:123456789012:deliverystream/
example-user-pool-activity-exported"
         },
         "LogLevel": "INFO"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Veja a seguir um exemplo de corpo de solicitação `SetLogDeliveryConfiguration` que define um bucket do Amazon S3 como o destino do log.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "S3Configuration": {
            "BucketArn": "arn:aws:s3:::amzn-s3-demo-logging-bucket"
         },
```

```
         "LogLevel": "INFO"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Veja a seguir um exemplo de corpo de `SetLogDeliveryConfiguration` solicitação que define um grupo de CloudWatch registros como o destino do registro.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "CloudWatchLogsConfiguration": {
            "LogGroupArn": "arn:aws:logs:us-west-2:123456789012:log-group:DOC-
EXAMPLE-LOG-GROUP"
         },
         "LogLevel": "INFO"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Como fornecer feedback sobre eventos

Os feedbacks sobre eventos não só afetam a avaliação de risco em tempo real, mas também aprimoram o algoritmo de avaliação de risco ao longo do tempo. Você pode fornecer feedback sobre a validade das tentativas de login por meio do console do Amazon Cognito e das operações de API.

> **Note**
>
> O feedback de seu evento influencia o nível de risco que o Amazon Cognito atribui às sessões de usuário subsequentes com as mesmas características.

No console do Amazon Cognito, selecione um usuário na guia Usuários e selecione Fornecer feedback de evento. É possível revisar os detalhes do evento e definir como válido (Set as valid ou definir como inválido (Set as invalid).

O console lista o histórico de login em detalhes do usuário no menu Usuários. Se você selecionar uma entrada, poderá marcar o evento como válido ou não válido. Você também pode fornecer feedback por meio da operação da API do grupo AdminUpdateAuthEventFeedbackde usuários e do AWS CLI comando admin-update-auth-event-feedback.

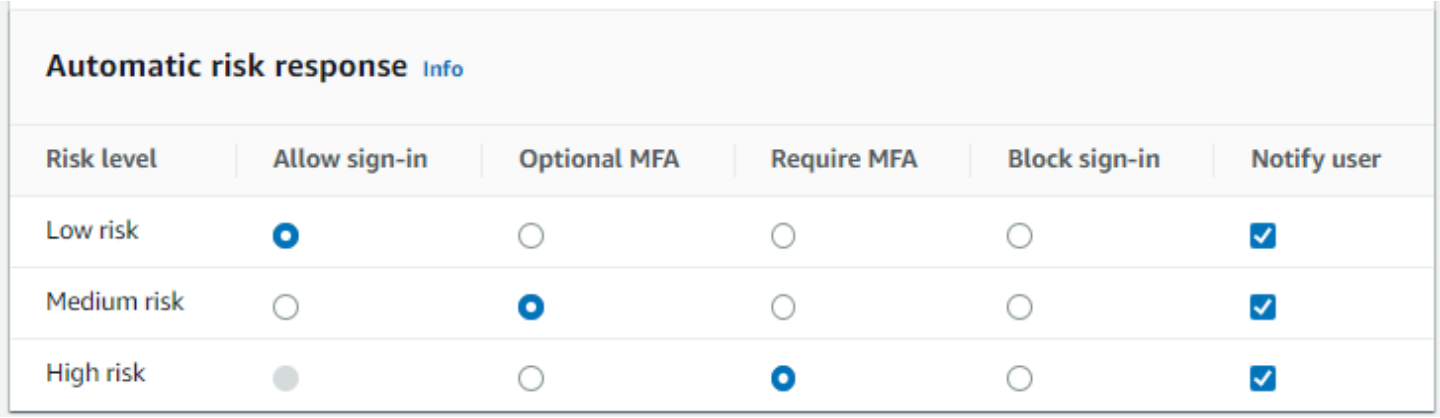
Ao selecionar Set as valid (Definir como válido) no console do Amazon Cognito ou fornecer um valor `FeedbackValue` de `valid` na API, você diz ao Amazon Cognito que confia em uma sessão de usuário em que o Amazon Cognito avaliou algum nível de risco. Ao selecionar Set as invalid (Definir como inválido) no console do Amazon Cognito ou fornecer um valor `FeedbackValue` de `invalid` na API, você diz ao Amazon Cognito que não confia em uma sessão de usuário ou não acredita que o Amazon Cognito avaliou um nível de risco alto o suficiente.

Como enviar mensagens de notificação

Com a proteção contra ameaças, o Amazon Cognito pode notificar seus usuários sobre tentativas de login arriscadas. O Amazon Cognito também pode solicitar que os usuários selecionem links para indicar se o login foi ou não válido. O Amazon Cognito usa esse feedback para melhorar a precisão da detecção de riscos para seu grupo de usuários.

> **Note**
>
> O Amazon Cognito só envia mensagens de notificação aos usuários quando a ação deles gera uma resposta automática ao risco: bloquear o login, permitir o login, definir a MFA como opcional ou exigir a MFA. Algumas solicitações podem ter um nível de risco atribuído, mas não geram respostas de risco automatizadas de autenticação adaptável. Para elas, seu grupo de usuários não envia notificações. Por exemplo, senhas incorretas podem ser registradas com uma classificação de risco, mas a resposta do Amazon Cognito é falha do login, não aplicar uma regra de autenticação adaptável.

Na seção Automatic risk response (Resposta automática a riscos), selecione Notify Users (Notificar usuários) para os casos de baixo, médio e alto risco.



O Amazon Cognito envia notificações por e-mail aos seus usuários, independentemente de eles terem verificado o endereço de e-mail.

Você pode personalizar mensagens de e-mail de notificação e disponibilizá-las em versões de texto simples e HTML. Para personalizar suas notificações por e-mail, abra Modelos de e-mail em Mensagens de autenticação adaptável em sua configuração de proteção contra ameaças. Para saber mais sobre modelos de e-mail, consulte Modelos de mensagens.

Coletar dados para proteção contra ameaças em aplicações

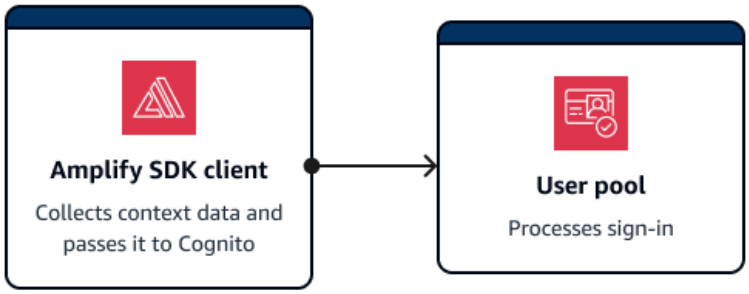
A autenticação adaptável do Amazon Cognito avalia os níveis de risco de tentativa de invasão da conta a partir de detalhes contextuais das tentativas de login dos usuários. A aplicação deve adicionar dados de contexto às solicitações de API para que a proteção contra ameaças do Amazon Cognito possa avaliar os riscos com mais precisão. Dados de contexto são informações como endereço IP, agente do navegador, informações do dispositivo e cabeçalhos de solicitação que fornecem informações contextuais sobre como o usuário se conectou ao grupo de usuários.

A responsabilidade central de uma aplicação que envia esse contexto ao Amazon Cognito é um parâmetro `EncodedData` nas solicitações de autenticação para grupos de usuários. Para adicionar esses dados às suas solicitações, você pode implementar o Amazon Cognito com um SDK que gera automaticamente essas informações para você, ou você pode implementar um módulo para JavaScript iOS ou Android que coleta esses dados. Aplicativos somente para clientes que fazem solicitações diretas ao Amazon Cognito devem ser implementados. AWS Amplify SDKs As aplicações cliente-servidor que têm um servidor intermediário ou componente de API devem implementar um módulo SDK separado.

Nos cenários a seguir, seu frontend de autenticação gerencia a coleta de dados de contexto do usuário sem qualquer configuração adicional:

- O login gerenciado coleta e envia automaticamente dados de contexto para a proteção contra ameaças.
- Todas as AWS Amplify bibliotecas têm coleta de dados contextuais incorporada em seus métodos de autenticação.

Envio de dados de contexto do usuário em aplicações somente para clientes com Amplify



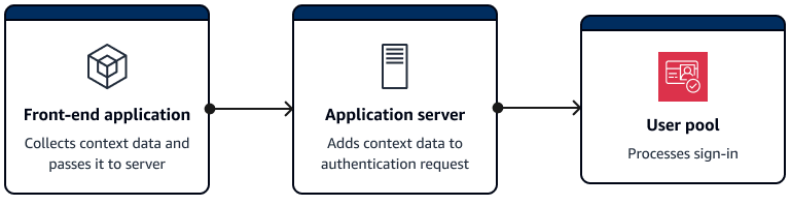
O Amplify SDKs oferece suporte a clientes móveis que se autenticam diretamente no Amazon Cognito. Clientes desse tipo fazem solicitações diretas de API às operações públicas de API do Amazon Cognito. Por padrão, os clientes do Amplify coletam automaticamente dados de contexto para a proteção contra ameaças.

Os aplicativos Amplify com JavaScript são uma exceção. Eles exigem a adição de um JavaScript módulo que coleta dados de contexto do usuário.

Normalmente, um aplicativo nessa configuração usa operações de API não autenticadas, como e. InitiateAuthRespondToAuthChallenge O UserContextDataobjeto ajuda a avaliar os riscos com mais precisão nessas operações. O Amplify SDKs adiciona informações do dispositivo e da sessão a um `EncodedData` parâmetro de. `UserContextData`

Coletar dados de contexto em aplicações cliente-servidor

Algumas aplicações têm um nível de frontend que coleta dados de autenticação do usuário e um nível de backend de aplicações que envia solicitações de autenticação para o Amazon Cognito. Essa é uma arquitetura comum em servidores web e aplicações auxiliadas por microsserviços. Nessas aplicações, você deve importar uma biblioteca pública de coleta de dados contextuais.



Normalmente, um servidor de aplicativos nessa configuração usa operações de API autenticadas, como AdminInitiateAuthe. AdminRespondToAuthChallenge O ContextDataobjeto ajuda o Amazon Cognito a avaliar os riscos dessas operações com mais precisão. O conteúdo de `ContextData` são os dados codificados que o frontend passou para o servidor e detalhes adicionais da solicitação HTTP do usuário para o servidor. Esses detalhes adicionais de contexto, como os cabeçalhos HTTP e o endereço IP, fornecem ao servidor de aplicações as características do ambiente do usuário.

Seu servidor de aplicativos também pode fazer login com operações de API não autenticadas, como e. InitiateAuthRespondToAuthChallenge O UserContextDataobjeto informa a análise de risco de proteção contra ameaças nessas operações. As operações nas bibliotecas de coleta de dados de contexto público disponíveis adicionam informações de segurança ao parâmetro `EncodedData` nas solicitações de autenticação. Além disso, configure seu grupo de usuários para aceitar dados de contexto adicionais e adicionar o IP de origem do usuário ao parâmetro `IpAddress` de `UserContextData`.

Para adicionar dados de contexto a aplicações cliente-servidor

1. Em seu aplicativo front-end, colete dados de contexto codificados do cliente com um iOS, Android ou módulo. JavaScript
2. Passe os dados codificados e os detalhes da solicitação de autenticação para seu servidor de aplicações.
3. No seu servidor de aplicações, extraia o endereço IP do usuário, os cabeçalhos HTTP relevantes, o nome do servidor solicitado e o caminho da solicitação HTTP. Preencha esses valores com o ContextDataparâmetro da sua solicitação de API para o Amazon Cognito.
4. Preencha o parâmetro `EncodedData` de `ContextData` na solicitação de API com os dados codificados do dispositivo que seu módulo SDK coletou. Adicione esses dados de contexto à solicitação de autenticação.

Bibliotecas de dados de contexto para aplicações cliente-servidor

JavaScript

O módulo `amazon-cognito-advanced-security-data.min.js` coleta `EncodedData` que você pode passar para o servidor de aplicações.

Adicione o `amazon-cognito-advanced-security-data.min.js` módulo à sua JavaScript configuração. `<region>`Substitua por um Região da AWS da lista a seguir: `us-east-1`us-east-2,`us-west-2`,`eu-west-1`,`eu-west-2`, ou`eu-central-1`.

```
<script src="https://amazon-cognito-assets.<region>.amazoncognito.com/amazon-cognito-
advanced-security-data.min.js"></script>
```

Para gerar um `encodedContextData` objeto que você possa usar no `EncodedData` parâmetro, adicione o seguinte à fonte do JavaScript aplicativo:

```
var encodedContextData = AmazonCognitoAdvancedSecurityData.getData(_username,
 _userpoolId, _userPoolClientId);
```

iOS/Swift

Para gerar dados de contexto, os aplicativos iOS podem integrar o AWSCognitoIdentityProvidermódulo ASF do Mobile SDK para iOS.

Para coletar dados de contexto codificados para proteção contra ameaças, adicione o seguinte trecho à aplicação:

```
import AWSCognitoIdentityProviderASF

let deviceId = getDeviceId()
let encodedContextData = AWSCognitoIdentityProviderASF.userContextData(
                            userPoolId,
                            username: username,
                            deviceId: deviceId,
                            userPoolClientId: userPoolClientId)

/**
 * Reuse DeviceId from keychain or generate one for the first time.
 */
func getDeviceId() -> String {
```

```
    let deviceIdKey = getKeyChainKey(namespace: userPoolId, key:
 "AWSCognitoAuthAsfDeviceId")

   if let existingDeviceId = self.keychain.string(forKey: deviceIdKey) {
        return existingDeviceId
    }

    let newDeviceId = UUID().uuidString
    self.keychain.setString(newDeviceId, forKey: deviceIdKey)
    return newDeviceId
}

/**
 * Get a namespaced keychain key given a namespace and key
 */
func getKeyChainKey(namespace: String, key: String) -> String {
    return "\(namespace).\(key)"
}
```

Android

Para gerar dados de contexto, os aplicativos Android podem integrar o aws-android-sdk-cognitoidentityprovidermódulo -asf do Mobile SDK for Android.

Para coletar dados de contexto codificados para proteção contra ameaças, adicione o seguinte trecho à aplicação:

```
UserContextDataProvider provider = UserContextDataProvider.getInstance();
// context here is android application context.
String encodedContextData = provider.getEncodedContextData(context, username,
 userPoolId, userPoolClientId);
```

Associar uma ACL AWS WAF da web a um grupo de usuários

AWS WAF é um firewall de aplicativos da web. Com uma lista de controle de acesso à AWS WAF web (web ACL), você pode proteger seu grupo de usuários contra solicitações indesejadas para sua interface de usuário hospedada clássica, login gerenciado e endpoints de serviço da API Amazon Cognito. A ACL da web oferece controle detalhado sobre todas as solicitações web HTTPS às quais o grupo de usuários responde. Para obter mais informações sobre a AWS WAF web ACLs, consulte Gerenciando e usando uma lista de controle de acesso à web (Web ACL) no Guia do AWS WAF desenvolvedor.

Quando você tem uma ACL AWS WAF da web associada a um grupo de usuários, o Amazon Cognito encaminha cabeçalhos e conteúdos não confidenciais selecionados das solicitações de seus usuários para. AWS WAF AWS WAF inspeciona o conteúdo da solicitação, compara com as regras que você especificou na sua ACL da web e retorna uma resposta ao Amazon Cognito.

Coisas que você deve saber sobre a AWS WAF web ACLs e o Amazon Cognito

- Você não pode configurar as regras de Web ACL para corresponder às informações de identificação pessoal (PII) nas solicitações do grupo de usuários, por exemplo, nomes de usuário, senhas, números de telefone ou endereços de e-mail. Esses dados não estarão disponíveis para AWS WAF. Em vez disso, configure suas regras de Web ACL para corresponder aos dados da sessão nos cabeçalhos, no caminho e no corpo, como endereços IP, agentes do navegador e operações de API solicitadas.
- As condições das regras da Web ACL só podem retornar respostas de bloco personalizadas à primeira solicitação dos usuários em uma página de login gerenciada interativa com o usuário. Quando as conexões subsequentes correspondem a uma condição de resposta de bloco personalizada, elas retornam seu código de status, cabeçalho e respostas de redirecionamento personalizados, mas uma mensagem de bloqueio padrão.
- Solicitações bloqueadas por AWS WAF não contam para a cota de taxa de solicitação de nenhum tipo de solicitação. O AWS WAF manipulador é chamado antes dos manipuladores de limitação no nível da API.
- Quando você cria uma ACL da web, há um pequeno tempo de espera até que a ACL da web seja totalmente propagada e esteja disponível para o Amazon Cognito. O tempo de propagação pode ser de alguns segundos a alguns minutos. AWS WAF retorna a `WAFUnavailableEntityException`quando você tenta associar uma ACL da web antes que ela seja totalmente propagada.
- É possível associar uma Web ACL a cada grupo de usuários.
- Sua solicitação pode ocasionar uma carga útil acima dos limites inspecionados pelo AWS WAF . Consulte Tratamento de componentes de solicitações de tamanho grande no Guia do AWS WAF desenvolvedor para saber como configurar como lidar AWS WAF com solicitações de grandes dimensões do Amazon Cognito.
- Você não pode associar uma ACL da web que usa a prevenção de aquisição de contas (ATP) do AWS WAF Fraud Control a um grupo de usuários do Amazon Cognito. O recurso ATP está no grupo de regras gerenciadas `AWS-AWSManagedRulesATPRuleSet`. Antes de associar uma Web ACL a um grupo de usuários, certifique-se de que ela não usa esse grupo de regras gerenciadas.

- Quando você tem uma ACL AWS WAF da web associada a um grupo de usuários e uma regra na sua ACL da web apresenta um CAPTCHA, isso pode causar um erro irrecuperável no registro do TOTP de login gerenciado. Para criar uma regra que tenha uma ação de CAPTCHA e não afete a TOTP do login gerenciado, consulte Configurando sua ACL AWS WAF da web para login gerenciado TOTP MFA.

AWS WAF inspeciona solicitações para os seguintes endpoints.

Login gerenciado e a IU hospedada clássica

Solicitações a todos os endpoints no Referência de login gerenciado e endpoints do grupo de usuários.

Operações públicas de API

Solicitações do seu aplicativo para a API do Amazon Cognito que não usam AWS credenciais para autorizar. Isso inclui operações de API como InitiateAuthRespondToAuthChallenge, GetUsere. As operações de API que estão no escopo de AWS WAF não exigem autenticação com AWS credenciais. Elas não são autenticadas nem autorizadas com uma string de sessão nem um token de acesso. Para obter mais informações, consulte Lista de operações de API agrupadas por modelo de autorização.

É possível configurar as regras na Web ACL com ações como Contar, Permitir, Bloquear ou apresentar um CAPTCHA em resposta a uma solicitação correspondente a uma regra. Para ter mais informações, consulte Regras do AWS WAF no Guia do desenvolvedor do AWS WAF . Dependendo da ação da regra, você pode personalizar a resposta que o Amazon Cognito retorna aos usuários.

Important

Suas opções para personalizar a resposta de erro dependem da forma como você faz uma solicitação de API.

- Você pode personalizar o código de erro e o corpo da resposta das solicitações do login gerenciado. Você só pode apresentar um CAPTCHA para o usuário resolver no login gerenciado.
- Para solicitações feitas com a API de grupos de usuários do Amazon Cognito, você pode personalizar o corpo da resposta de uma solicitação que recebe uma resposta Bloquear. Você também pode especificar um código de erro personalizado no intervalo de 400 a 499.

- O AWS Command Line Interface (AWS CLI) e o AWS SDKs retornam um `ForbiddenException` erro às solicitações que produzem uma resposta de bloco ou CAPTCHA.

Associar uma ACL da web ao grupo de usuários

Para trabalhar com uma ACL da web em seu grupo de usuários, seu diretor AWS Identity and Access Management (IAM) deve ter o Amazon Cognito AWS WAF e as seguintes permissões. Para obter informações sobre AWS WAF permissões, consulte Permissões de AWS WAF API no Guia do AWS WAF desenvolvedor.

JSON

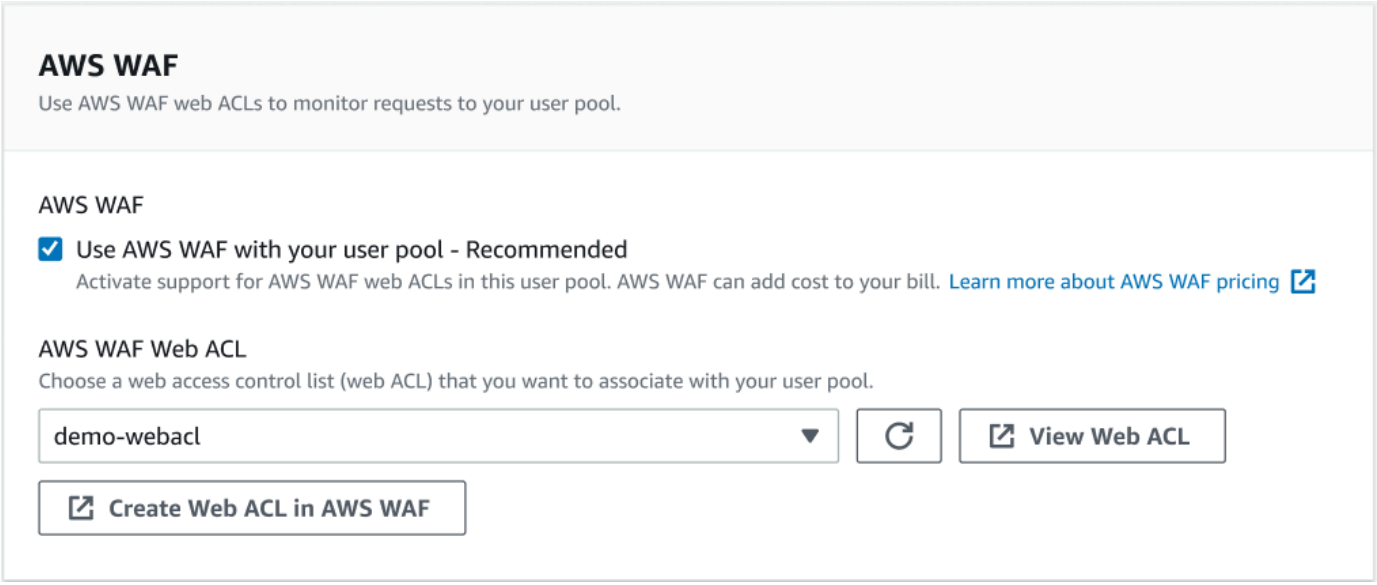
```
{
 "Version":"2012-10-17",
 "Statement": [
  {
   "Sid": "AllowWebACLUserPool",
   "Effect": "Allow",
   "Action": [
    "cognito-idp:ListResourcesForWebACL",
    "cognito-idp:GetWebACLForResource",
    "cognito-idp:AssociateWebACL"
   ],
   "Resource": [
    "arn:aws:cognito-idp:*:123456789012:userpool/*"
   ]
  },
  {
   "Sid": "AllowWebACLUserPoolWAFv2",
   "Effect": "Allow",
   "Action": [
    "wafv2:ListResourcesForWebACL",
    "wafv2:AssociateWebACL",
    "wafv2:DisassociateWebACL",
    "wafv2:GetWebACLForResource"
   ],
   "Resource": "arn:aws:wafv2:*:123456789012:*/webacl/*/*"
  },
  {
```

```
    "Sid": "DisassociateWebACL1",
    "Effect": "Allow",
    "Action": "wafv2:DisassociateWebACL",
    "Resource": "*"
   },
   {
    "Sid": "DisassociateWebACL2",
    "Effect": "Allow",
    "Action": [
     "cognito-idp:DisassociateWebACL"
    ],
    "Resource": [
     "arn:aws:cognito-idp:*:123456789012:userpool/*"
    ]
   }
  ]
}
```

Embora você deva conceder permissões do IAM, as ações listadas são somente com permissão e não correspondem a nenhuma operação de API.

AWS WAF Para ativar seu grupo de usuários e associar uma ACL da web

1. Faça login no console do Amazon Cognito.
2. No painel de navegação, escolha User Pools (Grupos de usuários) e escolha o grupo de usuários que deseja editar.
3. Clique na guia AWS WAF na seção Segurança.
4. Escolha Editar.
5. Selecione Usar AWS WAF com seu grupo de usuários.



6. Escolha uma AWS WAF Web ACL que você já criou ou escolha Criar ACL da Web em AWS WAF para criar uma em uma nova AWS WAF sessão no. Console de gerenciamento da AWS
7. Escolha Salvar alterações.

Para associar programaticamente uma ACL da web ao seu grupo de usuários no AWS Command Line Interface ou a um SDK, use a AssociateWebACL da API. AWS WAF O Amazon Cognito não tem uma operação de API separada que associe uma ACL da web.

Testando e registrando AWS WAF na web ACLs

Quando você define uma ação de regra como Count em sua ACL da web, AWS WAF adiciona a solicitação a uma contagem de solicitações que correspondem à regra. Para testar uma ACL da web com o grupo de usuários, defina as ações da regra como Count (Contar) e considere o volume de solicitações correspondentes a cada regra. Por exemplo, se uma regra que você deseja definir como uma ação Block (Bloquear) corresponder a um grande número de solicitações que você considera tráfego normal de usuários, talvez seja necessário reconfigurar sua regra. Para ter mais informações, consulte Testar e ajustar suas proteções do AWS WAF no Guia do desenvolvedor do AWS WAF .

Você também pode configurar AWS WAF para registrar cabeçalhos de solicitação em um grupo de CloudWatch logs do Amazon Logs, em um bucket do Amazon Simple Storage Service (Amazon S3) ou em um Amazon Data Firehose. Você pode identificar as solicitações do Amazon Cognito realizadas com a API de grupos de usuários pelo `x-amzn-cognito-client-id` e pelo `x-amzn-cognito-operation-name`. As solicitações de login gerenciado incluem somente o cabeçalho

`x-amzn-cognito-client-id`. Para obter mais informações, consulte Logging web ACL traffic (Registrar em log o tráfego da ACL da web) no Guia do desenvolvedor do AWS WAF .

AWS WAF A web ACLs está disponível em todos os planos de recursos do grupo de usuários. Os recursos de segurança do AWS WAF complementam a proteção contra ameaças do Amazon Cognito. É possível ativar os dois recursos em um grupo de usuários. O AWS WAF cobra separadamente pela inspeção das solicitações do grupo de usuários. Para obter mais informações, consulte AWS WAF Preço.

Os dados da AWS WAF solicitação de registro estão sujeitos à cobrança adicional do serviço ao qual você segmenta seus registros. Para obter mais informações, consulte Definição de preço para registrar informações de tráfego da ACL da Web no Guia do desenvolvedor do AWS WAF .

Sensibilidade entre maiúsculas e minúsculas do grupo de usuários

Os grupos de usuários do Amazon Cognito que você cria no não Console de gerenciamento da AWS diferenciam maiúsculas de minúsculas por padrão. Quando um grupo de usuários não faz distinção entre maiúsculas e minúsculas, User@example.com e user@example.com referem-se ao mesmo usuário. Quando nomes de usuário em um grupo de usuários não fazem distinção entre maiúsculas e minúsculas, os atributos `preferred_username` e `email` também não fazem essa distinção.

A não diferenciação de maiúsculas e minúsculas não se aplica somente às entradas de atributos, mas também às saídas. Valores de atributos com letras maiúsculas e minúsculas em grupos de usuários que não as diferenciam são reduzidos a minúsculas na saída de texto do grupo de usuários. Exemplos de saída de texto do grupo de usuários são respostas do UserInfo, respostas à consulta do usuário, como a saída de GetUser, e eventos de entrada para os gatilhos do Lambda.

Para explicar as configurações de distinção entre maiúsculas e minúsculas de grupo de usuários, identifique usuários no código da aplicação com base em um atributo de usuário alternativo. Como o uso de maiúsculas e minúsculas no nome de usuário, do nome de usuário preferido ou no atributo de endereço de e-mail pode variar em diferentes perfis de usuário, consulte o atributo `sub`. Você também pode criar um atributo personalizado imutável no seu grupo de usuários e designar ao atributo seu próprio valor de identificador exclusivo em cada novo perfil de usuário. Ao criar um usuário pela primeira vez, é possível gravar um valor em um atributo personalizado imutável que você criou.

> **Note**
>
> Independentemente das configurações de diferenciação entre maiúsculas e minúsculas do grupo de usuários, o Amazon Cognito exige que um usuário federado de um provedor de identidade (IdP) SAML ou OIDC passe uma declaração `NameId` ou `sub` exclusiva e que diferencie maiúsculas e minúsculas. Para obter mais informações sobre a distinção entre maiúsculas e minúsculas do identificador exclusivo e SAML IdPs, consulteImplementar o login SAML iniciado pelo SP.

Criar um grupo de usuários com distinção entre maiúsculas e minúsculas

Se você criar recursos com as operações AWS Command Line Interface (AWS CLI) e de API CreateUserPool, como, deverá definir o `CaseSensitive` parâmetro booleano como. `false` Essa configuração cria um grupo de usuários sem distinção entre maiúsculas e minúsculas. Se você não especificar um valor, a `CaseSensitive` definirá como padrão `true`. Os grupos de usuários que você cria no console do Amazon Cognito não diferenciam maiúsculas de minúsculas. Para produzir um grupo de usuários com distinção entre letras maiúsculas e minúsculas, você deve usar a operação `CreateUserPool`. Antes de 12 de fevereiro de 2020, grupos de usuários tinham distinção entre maiúsculas e minúsculas por padrão, independentemente da plataforma.

No menu de login do Console de gerenciamento da AWS e na `UsernameConfiguration` propriedade de DescribeUserPool, você pode revisar as configurações de distinção entre maiúsculas e minúsculas de cada grupo de usuários em sua conta.

Migração para um novo grupo de usuários

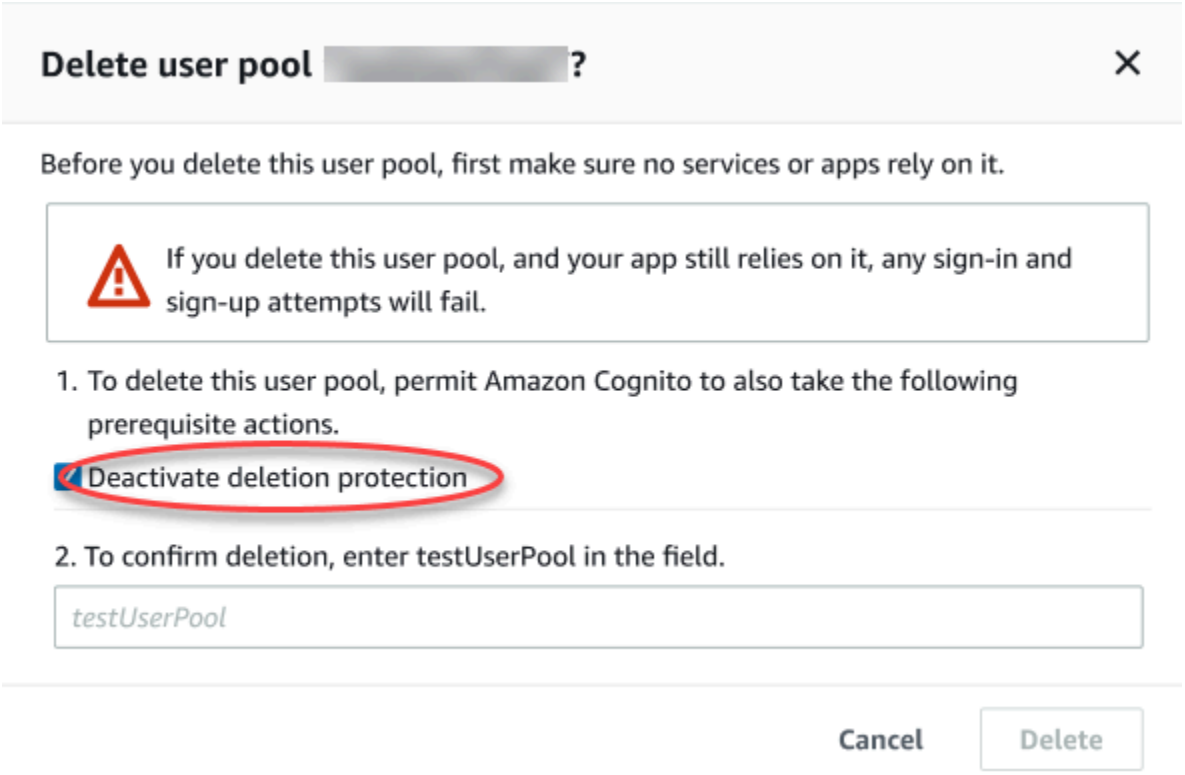
Devido a possíveis conflitos entre perfis de usuário, você não pode alterar um grupo de usuários do Amazon Cognito que faz distinção entre maiúsculas e minúsculas para um que não faça essa distinção. Em vez disso, faça a migração dos seus usuários para um novo grupo de usuários. Você deve criar um código de migração para resolver conflitos relacionados a maiúsculas e minúsculas. Esse código deve retornar um novo usuário exclusivo ou rejeitar a tentativa de login quando detectar um conflito. Em um novo grupo de usuários sem distinção entre maiúsculas e minúsculas, atribua um Migrar o acionador do Lambda do usuário. A AWS Lambda função pode criar usuários no novo grupo de usuários que não diferencia maiúsculas de minúsculas. Quando o usuário não conseguir fazer login com o grupo de usuários sem distinção entre maiúsculas e minúsculas, a função do Lambda localizará e duplicará o usuário desse grupo com distinção entre maiúsculas e minúsculas. Você também pode ativar um gatilho ForgotPasswordLambda

de usuário de migração em eventos. O Amazon Cognito transmite informações do usuário e metadados de eventos da ação de login ou recuperação de senha para a sua função do Lambda. Você pode usar dados de evento para gerenciar conflitos entre nomes de usuário e endereços de e-mail quando sua função cria o usuário em seu grupo de usuários sem distinção entre maiúsculas e minúsculas. Esses conflitos ocorrem entre nomes de usuário e endereços de e-mail que seriam exclusivos em um grupo de usuários sem distinção entre letras maiúsculas e minúsculas.

Para obter mais informações sobre como usar um gatilho Lambda de migração de usuários entre grupos de usuários do Amazon Cognito, consulte Migração de usuários para grupos de usuários do Amazon Cognito no blog. AWS

Proteção contra exclusão do grupo de usuários

Para que os administradores não excluam acidentalmente seu grupo de usuários, ative a proteção contra exclusão. Com a proteção contra exclusão ativa, você deve confirmar que deseja excluir o grupo de usuários antes de excluí-lo. Ao excluir um grupo de usuários no Console de gerenciamento da AWS, você pode desativar a proteção contra exclusão ao mesmo tempo. Quando você aceita a solicitação para desativar a proteção contra exclusão e confirma sua intenção de excluir, o Amazon Cognito exclui o grupo de usuários, conforme mostrado na imagem a seguir.



Quando quiser excluir um grupo de usuários com uma solicitação da API do Amazon Cognito, você deve primeiro `DeletionProtection` fazer a alteração `Inactive` em uma UpdateUserPoolsolicitação. Se você não desativar a proteção contra exclusão, o Amazon Cognito retornará um erro `InvalidParameterException`. Depois de desativar a proteção contra exclusão, você pode excluir o grupo de usuários em uma DeleteUserPoolsolicitação.

O Amazon Cognito ativa a Deletion protection (Proteção contra exclusão) por padrão quando você cria um grupo de usuários no Console de gerenciamento da AWS. Quando você cria um grupo de usuários com a API `CreateUserPool`, a proteção contra exclusão fica inativa por padrão. Para usar esse recurso nos grupos de usuários que você cria com o AWS CLI ou com um AWS SDK, defina o `DeletionProtection` parâmetro como. `True`

É possível ativar ou desativar o status da proteção contra exclusão no contêiner Proteção contra exclusão no menu Configurações no console do Amazon Cognito.

Como configurar a proteção contra exclusão

1. Acesse o console do Amazon Cognito. Você pode ser solicitado a fornecer suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Selecione o menu Configurações e navegue até a guia Proteção contra exclusão. Selecione Ativar ou Desativar.
5. Confirme sua escolha na próxima caixa de diálogo.

Gerenciar respostas de erro de existência do usuário

O Amazon Cognito permite personalizar respostas de erro retornadas por grupos de usuários. As respostas de erro personalizadas estão disponíveis para operações de criação e autenticação de usuários, recuperação de senha e confirmação.

Use o `PreventUserExistenceErrors` de um cliente da aplicação de grupo de usuários para habilitar ou desabilitar erros relacionados à existência do usuário. Ao criar um cliente de aplicação utilizando a API de grupo de usuários do Amazon Cognito `LEGACY` será `PreventUserExistenceErrors` ou então desativada, por padrão. No console do Amazon Cognito, a opção Habilitar a prevenção de erros de existência do usuário — uma configuração de

ENABLED para PreventUserExistenceErrors — é selecionada por padrão. Para atualizar a configuração PreventUserExistenceErrors, siga um destes procedimentos:

- Altere o valor de PreventUserExistenceErrors entre ENABLED e LEGACY em uma solicitação da API UpdateUserPoolClient.
- Edite seu cliente de aplicação no console do Amazon Cognito e altere o estado de Habilitar a prevenção de erros de existência do usuário entre selecionado (ENABLED) e desmarcado (LEGACY).

Quando essa propriedade tem um valor LEGACY, o cliente da aplicação retorna uma resposta de erro UserNotFoundException quando um usuário tenta fazer login com um nome de usuário que não existe no seu grupo de usuários.

Quando essa propriedade tem um valor ENABLED, o cliente da aplicação não divulga a inexistência de uma conta de usuário em seu grupo de usuários e exibe o erro UserNotFoundException. Uma configuração PreventUserExistenceErrors de ENABLED tem os seguintes efeitos quando você envia uma solicitação de um nome de usuário que não existe:

- O Amazon Cognito responde com informações não específicas às solicitações de API, em que sua resposta poderia revelar a existência de um usuário válido.
- O Amazon Cognito retorna uma resposta genérica de falha de autenticação para solicitações de esquecimento de senha e solicitações de autenticação com fluxos de autenticação, exceto para autenticação baseada em opções (USER_AUTH). Por exemplo, USER_SRP_AUTH ou CUSTOM_AUTH. A resposta de erro informa que o nome de usuário ou a senha está incorreta.
- O Amazon Cognito responde às solicitações de autenticação baseada em opções com uma seleção aleatória dos tipos de desafio permitidos para o grupo de usuários. Seu grupo de usuários pode retornar uma chave de acesso, senha de uso único ou desafio de senha.
- O comportamento da confirmação da conta do Amazon Cognito e da recuperação de senha APIs alterna entre retornar uma resposta indicando que um código foi enviado para uma mídia de entrega simulada e retornar um erro. InvalidParameterException

As informações a seguir detalham os comportamentos das operações do grupo de usuários quando PreventUserExistenceErrors está definido como ENABLED.

Operações de criação e autenticação de usuários

Você pode configurar respostas de erro na autenticação de nome de usuário-senha e Senha remota segura (SRP). Também é possível personalizar os erros retornados com a autenticação personalizada. A autenticação baseada em opções não é afetada pela sua configuração `PreventUserExistenceErrors`.

Detalhes da divulgação da existência do usuário nos fluxos de autenticação

Autenticação baseada em opções

No fluxo de autenticação baseado em opções `USER_AUTH`, o Amazon Cognito retorna um desafio dos principais fatores de autenticação que estão disponíveis, dependendo da configuração do seu grupo de usuários e dos atributos dos usuários. Esse fluxo de autenticação pode retornar desafios de senha, senha remota segura (SRP), WebAuthn (chave de acesso), senha de uso único por SMS (OTP) ou OTP por e-mail. Com a opção `PreventUserExistenceErrors` ativa, o Amazon Cognito desafia usuários inexistentes a realizarem uma ou mais das formas de autenticação disponíveis. Com `PreventUserExistenceErrors` inativo, o Amazon Cognito retorna uma exceção `UserNotFound`.

Autenticação com nome de usuário e senha

Os fluxos de autenticação `ADMIN_USER_PASSWORD_AUTH` e `USER_PASSWORD_AUTH`, além do fluxo `PASSWORD` de `USER_AUTH` retornam `NotAuthorizedException` com a mensagem `Incorrect username or password` quando `PreventUserExistenceErrors` está ativo. Quando `PreventUserExistenceErrors` está inativo, esses fluxos retornam `UserNotFoundException`.

Autenticação baseada em senha remota segura (SRP)

Como prática recomendada, implemente somente `PreventUserExistenceErrors` com `USER_SRP_AUTH` ou o fluxo `PASSWORD_SRP` de `USER_AUTH` grupos de usuários sem endereço de e-mail, número de telefone ou atributos preferenciais de alias de nome de usuário. Usuários com atributos de alias podem não estar sujeitos à supressão da existência do usuário no fluxo de autenticação SRP. Os fluxos de autenticação de nome de usuário e senha, `ADMIN_USER_PASSWORD_AUTH`, `USER_PASSWORD_AUTH` e o desafio `USER_AUTH` `PASSWORD`, suprimem totalmente a existência de usuários com base em atributos de alias.

Quando alguém tenta fazer login no SRP com um nome de usuário não conhecido pelo cliente de aplicação, o Amazon Cognito retorna uma resposta simulada na primeira etapa conforme descrito em RFC 5054. O Amazon Cognito retorna o mesmo salt e um ID de usuário interno no formato de

UUID para a mesma combinação de nome de usuário e grupo de usuários. Quando você envia uma solicitação de API `RespondToAuthChallenge` com prova de senha, o Amazon Cognito retorna um erro genérico `NotAuthorizedException` quando o nome de usuário ou a senha está incorreta. Para obter mais informações sobre como implementar uma autenticação SRP, consulte Fazer login com senhas persistentes e carga útil segura.

> **Note**
>
> Você pode simular uma resposta genérica com a autenticação de nome de usuário e senha se estiver usando atributos de alias baseados em verificação e se o nome de usuário imutável não estiver formatado como um UUID.

Acionador do Lambda do desafio de autenticação personalizada

O Amazon Cognito invoca os acionadores do Lambda do desafio de autenticação personalizada quando os usuários tentam fazer login com o fluxo de autenticação `CUSTOM_AUTH`, mas o nome de usuário não é encontrado. O evento de entrada inclui um parâmetro booleano nomeado `UserNotFound` com um valor de `true` para qualquer usuário inexistente. Esse parâmetro aparece nos eventos de solicitação que seu grupo de usuários envia às funções do Lambda do desafio de criação, definição e verificação de autenticação que compõem a arquitetura de autenticação personalizada. Ao examinar esse indicador na lógica da função do Lambda, você pode simular desafios de autenticação personalizados para usuários não existentes.

Acionador do Lambda de pré-autenticação

O Amazon Cognito invoca o acionador de pré-autenticação quando os usuários tentam fazer login, mas o nome de usuário não é encontrado. O evento de entrada inclui um parâmetro `UserNotFound` com um valor de `true` para qualquer usuário inexistente.

A lista a seguir descreve o efeito `PreventUserExistenceErrors` na criação de contas de usuário.

Detalhes da divulgação da existência do usuário nos fluxos de criação do usuário

SignUp

A operação `SignUp` sempre retorna `UsernameExistsException` quando um nome de usuário já está sendo usado. Se você não quiser que o Amazon Cognito retorne um erro

`UsernameExistsException` para endereços de e-mail e números de telefone ao inscrever usuários na aplicação, use atributos de alias baseados em verificação. Para obter mais informações sobre aliases, consulte Personalização dos atributos de login.

Para ver um exemplo de como o Amazon Cognito pode impedir o uso de solicitações da API `SignUp` para descobrir usuários no grupo de usuários, consulte Evitar erros `UsernameExistsException` de endereços de e-mail e números de telefone na inscrição.

Usuários importados

Se `PreventUserExistenceErrors` estiver habilitado durante a autenticação de usuários importados, será retornado um erro genérico `NotAuthorizedException`, que indica que o nome de usuário ou a senha estava incorreta, em vez de `PasswordResetRequiredException`. Consulte Solicitação de redefinição de senha aos usuários importados para obter mais informações.

Migrar o acionador do Lambda do usuário

O Amazon Cognito retornará uma resposta simulada para usuários não existentes quando uma resposta vazia tiver sido definida no contexto do evento original pelo acionador do Lambda. Para obter mais informações, consulte Como importar usuários com um acionador do Lambda de migração de usuários.

Evitar erros **`UsernameExistsException`** de endereços de e-mail e números de telefone na inscrição

O exemplo a seguir demonstra como, ao configurar atributos de alias no grupo de usuários, você pode impedir que endereços de e-mail e números de telefone duplicados gerem erros `UsernameExistsException` em resposta às solicitações da API `SignUp`. Você deve ter criado o grupo de usuários com o endereço de e-mail ou o número de telefone como atributos de alias. Para obter mais informações, consulte a seção Personalizar atributos de login de Atributos de grupos de usuários.

1. Jie se inscreve com um novo nome de usuário e também fornece o endereço de e-mail `jie@example.com`. O Amazon Cognito envia um código para o endereço de e-mail dele.

   Exemplo de AWS CLI comando

   ```
   aws cognito-idp sign-up --client-id 1234567890abcdef0 --username jie --password
    PASSWORD --user-attributes Name="email",Value="jie@example.com"
   ```

Exemplo de resposta

```
{
    "UserConfirmed": false,
    "UserSub": "<subId>",
    "CodeDeliveryDetails": {
        "AttributeName": "email",
        "Destination": "j****@e****",
        "DeliveryMedium": "EMAIL"
    }
}
```

2. Jie fornece o código enviado a ele para confirmar a propriedade do endereço de e-mail. Isso conclui seu registro como usuário.

   Exemplo de AWS CLI comando

   ```
   aws cognito-idp confirm-sign-up --client-id 1234567890abcdef0 --username=jie --
   confirmation-code xxxxxx
   ```

3. Shirley registra uma nova conta de usuário e fornece o endereço de e-mail `jie@example.com`. O Amazon Cognito não retorna um erro `UsernameExistsException` e envia um código de confirmação para o endereço de e-mail de Jie.

   Exemplo de AWS CLI comando

   ```
   aws cognito-idp sign-up --client-id 1234567890abcdef0 --username shirley --password
    PASSWORD --user-attributes Name="email",Value="jie@example.com"
   ```

   Exemplo de resposta

   ```
   {
       "UserConfirmed": false,
       "UserSub": "<new subId>",
       "CodeDeliveryDetails": {
           "AttributeName": "email",
           "Destination": "j****@e****",
           "DeliveryMedium": "EMAIL"
       }
   }
   ```

4. Em um cenário diferente, Shirley é proprietária de `jie@example.com`. Shirley recupera o código que o Amazon Cognito enviou para o endereço de e-mail de Jie e tenta confirmar a conta.

   Exemplo de AWS CLI comando

   ```
   aws cognito-idp confirm-sign-up --client-id 1234567890abcdef0 --username=shirley --
   confirmation-code xxxxxx
   ```

   Exemplo de resposta

   ```
   An error occurred (AliasExistsException) when calling the ConfirmSignUp operation: An
    account with the email already exists.
   ```

O Amazon Cognito não retorna um erro à solicitação `aws cognito-idp sign-up` de Shirley, apesar de `jie@example.com` ter sido atribuído a um usuário existente. Shirley deve demonstrar a propriedade do endereço de e-mail antes que o Amazon Cognito retorne uma resposta de erro. Em um grupo de usuários com atributos de alias, esse comportamento impede o uso da API `SignUp` pública para verificar se existe um usuário com um determinado endereço de e-mail ou número de telefone.

Esse comportamento é diferente da resposta que o Amazon Cognito retorna à solicitação `SignUp` com um nome de usuário existente, conforme mostrado no exemplo a seguir. Embora Shirley saiba, com base nessa resposta, que já existe um usuário com o nome `jie`, não é possível saber sobre nenhum endereço de e-mail ou número de telefone associado ao usuário.

Exemplo de comando da CLI

```
aws cognito-idp sign-up --client-id 1example23456789 --username jie --password PASSWORD
      --user-attributes Name="email",Value="shirley@example.com"
```

Exemplo de resposta

```
An error occurred (UsernameExistsException) when calling the SignUp operation: User
 already exists
```

Operações de redefinição de senha

O Amazon Cognito retorna as respostas a seguir às operações de redefinição de senha do usuário quando você evita erros de existência do usuário.

ForgotPassword

Quando um usuário não é encontrado, está desativado ou não tem um mecanismo de entrega verificado para recuperar a senha, o Amazon Cognito retorna `CodeDeliveryDetails` com um meio de entrega simulado para um usuário. O meio de entrega simulado é determinado pelo formato de entrada do nome de usuário e as configurações de verificação do grupo de usuários.

ConfirmForgotPassword

O Amazon Cognito retorna o erro `CodeMismatchException` para usuários que não existem ou estão desabilitados. Se um código não for solicitado ao ser usado o `ForgotPassword`, o Amazon Cognito retornará o erro `ExpiredCodeException`.

Operações de confirmação

O Amazon Cognito retorna as respostas a seguir às operações de confirmação e verificação do usuário quando você evita erros de existência do usuário.

ResendConfirmationCode

O Amazon Cognito retorna `CodeDeliveryDetails` para um usuário desabilitado ou um usuário que não existe. O Amazon Cognito envia um código de confirmação para o e-mail ou telefone do usuário existente.

ConfirmSignUp

Retorna `ExpiredCodeException` se um código tiver expirado. O Amazon Cognito retorna `NotAuthorizedException` quando um usuário não está autorizado. Se o código não corresponder ao que o servidor espera que o Amazon Cognito retorne `CodeMismatchException`.

Referência de login gerenciado e endpoints do grupo de usuários

O Amazon Cognito tem dois modelos de autenticação de grupos de usuários: com a API de grupos de usuários e com o servidor de autorização OAuth 2.0. Use a API quando quiser recuperar tokens do OpenID Connect (OIDC) AWS com um SDK no back-end do seu aplicativo. Use o servidor de autorização quando quiser implementar seu grupo de usuários como um provedor OIDC. O servidor de autorização adiciona recursos como login federado, autorização de API e M2M com escopos de token de acesso e login gerenciado. Você pode usar os modelos de API e OIDC, individualmente ou em conjunto, configurados no nível do grupo de usuários ou no nível do cliente de aplicação.

Esta seção é uma referência para a implementação do modelo OIDC. Para obter mais informações sobre os dois modos de autenticação, consulte Noções básicas sobre a autenticação de API, OIDC e páginas de login gerenciado.

O Amazon Cognito ativa as páginas da web públicas listadas aqui quando você atribui um domínio ao grupo de usuários. O domínio serve como um ponto de acesso central para todos os clientes da aplicação. Isso inclui o login gerenciado, no qual os usuários podem se cadastrar e fazer login (Endpoint de login) e logout (Endpoint de logout). Para obter mais informações sobre esses recursos, consulte Login gerenciado do grupo de usuários.

Essas páginas também incluem os recursos públicos da web que permitem que seu grupo de usuários se comunique com provedores de identidade SAML, OpenID Connect (OIDC OAuth) e 2.0 () de terceiros. IdPs Para conectar um usuário com um provedor de identidades federado, os usuários devem iniciar uma solicitação para o Endpoint de login de login gerenciado interativo ou o Autorizar endpoint do OIDC. O endpoint de autorização redireciona seus usuários para suas páginas de login gerenciado ou para a página de login do IdP.

Sua aplicação também pode fazer login de usuários locais com a API de grupos de usuários do Amazon Cognito. Um usuário local existe exclusivamente em seu diretório de grupo de usuários sem federação por meio de um IdP externo.

Além do login gerenciado, o Amazon Cognito se integra com SDKs Android JavaScript, iOS e muito mais. Eles SDKs fornecem ferramentas para realizar operações de API de grupos de usuários com endpoints de serviço da API Amazon Cognito. Para obter mais informações sobre endpoints de serviço, consulte Endpoints e cotas do Amazon Cognito Identity.

> **Warning**
>
> Não fixe os certificados TLS (Transport Layer Security) da entidade final ou intermediária para os domínios do Amazon Cognito. AWS gerencia todos os certificados de todos os endpoints e domínios de prefixo do seu grupo de usuários. As autoridades de certificação (CAs) na cadeia de confiança que dá suporte aos certificados do Amazon Cognito são alternadas e renovadas dinamicamente. Quando você fixa seu aplicativo em um certificado intermediário ou secundário, seu aplicativo pode falhar sem aviso prévio ao AWS alternar os certificados.
>
> Em vez disso, fixe sua aplicação em todos os certificados raiz da Amazon disponíveis. Para obter mais informações, consulte as práticas recomendadas e as recomendações em Fixação do certificado no Guia do usuário do AWS Certificate Manager .

Tópicos

Endpoints de login gerenciado interativo com o usuário e IU hospedada clássica

O Amazon Cognito ativa os endpoints de login gerenciado nesta seção quando você adiciona um domínio ao grupo de usuários. São páginas da web nas quais os usuários podem concluir as principais operações de autenticação de um grupo de usuários. Eles incluem páginas para gerenciamento de senhas, autenticação multifator (MFA) e verificação de atributos.

As páginas da web que compõem o login gerenciado são uma aplicação web de frontend para sessões interativas de usuários com os clientes. A aplicação deve invocar o login gerenciado nos navegadores dos usuários. O Amazon Cognito não permite o acesso programático às páginas da web deste capítulo. Esses endpoints de federação no Provedor de identidades e endpoints de terceiros confiáveis que retornam uma resposta JSON podem ser consultados diretamente no código da aplicação. O Autorizar endpoint redireciona para o login gerenciado ou para uma página de login do IdP e também deve ser aberto nos navegadores dos usuários.

Todos os endpoints do grupo de usuários aceitam tráfego IPv4 e endereços IP IPv6 de origem.

Os tópicos deste guia descrevem detalhadamente os endpoints de IU hospedada clássica e login gerenciado usados com frequência. A diferença entre o login gerenciado e a IU hospedada é visível, não funcional. Com exceção de `/passkeys/add`, todos os caminhos são compartilhados entre as duas versões da identidade visual do login gerenciado.

O Amazon Cognito disponibiliza as páginas da web a seguir quando você atribui um domínio ao grupo de usuários.

Endpoints de login gerenciado

URL do endpoint	Description	Como é acessado
https://*Your user pool domain*/login	Faz login no grupo de usuários locais e federados.	Redirecione de endpoints como Autorizar endpoint, `/logout` e `/confirmforgotPassword`. Consulte Endpoint de login.
https://*Your user pool domain*/logout	Desconecta os usuários do grupo de usuários.	Link direto. Consulte Endpoint de logout.
https://*Your user pool domain*/ConfirmUser	Confirma os usuários que selecionaram um link de e-mail para confirmar a respectiv a conta de usuário.	O usuário selecionou um link em uma mensagem de e-mail.
https://*Your user pool domain*/signup	Inscreve um novo usuário. A página `/login` direciona o usuário para `/signup` quando ele seleciona Sign up (Cadastrar-se).	Link direto com os mesmos parâmetros do `/oauth2/authorize`.
https://*Your user pool domain*/confirm	Depois que o grupo de usuários envia um código de confirmação a um usuário que se inscreveu, solicita o código ao usuário.	Redireciona somente de `/signup`.
https://*Your user pool domain*/Esqueci minha senha	Solicita que o usuário informe o nome de usuário e envia um código para redefinição da senha. A página `/login` direciona o usuário para `/forgotPassword` quando ele seleciona Forgot your	1. Do link Esqueci minha senha em `/login`. 2. Link direto com os mesmos parâmetros do `/oauth2/authorize`.

URL do endpoint	Description	Como é acessado
	password? (Esqueceu a senha?).	
https://*Your user pool domain*/confirmForgotPassword	Solicita ao usuário o código para redefinição da senha e uma nova senha. A página `/forgotPassword` direciona o usuário para `/confirmforgotPassword` quando ele seleciona Reset your password (Redefinir a senha).	Redireciona somente de `/forgotPassword` .
https://*Your user pool domain*/reenviar código	Envia um novo código de confirmação a um usuário que se inscreveu no grupo de usuários.	Redireciona somente do link Enviar um novo código em `/confirm`.
https://*Your user pool domain*/passkeys/add	Registra uma nova chave de acesso. Disponível somente no login gerenciado.	• No fluxo de cadastro após a confirmação em clientes da aplicação que oferecem suporte à autenticação por chave de acesso. • Link direto com os mesmos parâmetros do `/oauth2/authorize` .

Tópicos

- O endpoint de login do login gerenciado: /login
- O endpoint de logout do login gerenciado: /logout

O endpoint de login do login gerenciado: **`/login`**

O endpoint de login é um servidor de autenticação e um destino de redirecionamento do Autorizar endpoint. É o ponto de entrada para o login gerenciado quando você não especifica um provedor de

identidade. Ao gerar um redirecionamento para o endpoint de login, ele carrega a página de login e apresenta as opções de autenticação configuradas para o cliente ao usuário.

> **Note**
>
> O endpoint de login é um componente do login gerenciado. Na aplicação, invoque a federação e as páginas do login gerenciado que redirecionam para o endpoint de login. O acesso direto dos usuários ao endpoint de login não é uma prática recomendada.

GET/login

O endpoint de `/login` só é compatível com `HTTPS GET` para a solicitação inicial do usuário. A aplicação invoca a página em um navegador como o Chrome ou o Firefox. Quando você redireciona de Autorizar endpoint para `/login`, ele transmite todos os parâmetros que você forneceu na solicitação inicial. O endpoint de login é compatível com todos os parâmetros de solicitação do endpoint de autorização. Você também pode acessar o endpoint de login diretamente. Como prática recomendada, origine todas as sessões dos usuários em `/oauth2/authorize`.

Exemplo — solicitar que o usuário faça login

Este exemplo exibe a tela de login.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/login?
                response_type=code&
                client_id=ad398u21ijw3s9w3939&
                redirect_uri=https://YOUR_APP/redirect_uri&
                state=STATE&
                scope=openid+profile+aws.cognito.signin.user.admin
```

Exemplo - resposta

O servidor de autenticação redireciona o estado e o código de autorização à aplicação. O servidor deve retornar o código e o estado nos parâmetros de string de consulta e não no fragmento.

```
HTTP/1.1 302 Found
                    Location: https://YOUR_APP/redirect_uri?
code=AUTHORIZATION_CODE&state=STATE
```

Solicitação de login iniciada pelo usuário

Depois que o usuário carrega o endpoint de `/login`, ele pode inserir um nome de usuário e uma senha e selecionar Fazer login. Ao fazer isso, eles geram uma solicitação `HTTPS POST` com os mesmos parâmetros de solicitação de cabeçalho da solicitação `GET` e um corpo de solicitação com seu nome de usuário, senha e impressão digital do dispositivo.

O endpoint de logout do login gerenciado: **`/logout`**

O `/logout` é um endpoint de redirecionamento. Ele desconecta o usuário e o redireciona para um URL de saída autorizado para o cliente da aplicação ou o endpoint do `/login`. Os parâmetros disponíveis em uma solicitação GET para o endpoint `/logout` são personalizados para casos de uso de login gerenciado do Amazon Cognito.

O endpoint de desconexão é uma aplicação web de front-end para sessões interativas de usuários com os clientes. A aplicação deve invocar esse e outros endpoints de login gerenciado nos navegadores dos usuários.

Para redirecionar o usuário para o login gerenciado para que ele possa fazer login novamente, adicione um parâmetro `redirect_uri` à solicitação. Uma solicitação `logout` com um parâmetro `redirect_uri` também deve incluir parâmetros para sua solicitação subsequente ao Endpoint de login, como `client_id`, `response_type` e `scope`.

Para redirecionar seu usuário para uma página que você escolher, adicione Sessão permitida URLs ao seu cliente de aplicativo. Nas solicitações dos usuários para o endpoint `logout`, adicione parâmetros `logout_uri` e `client_id`. Se o valor de `logout_uri` for uma das saídas permitidas URLs para seu cliente de aplicativo, o Amazon Cognito redirecionará os usuários para essa URL.

Com o logout único (SLO) para SAML 2.0, o Amazon IdPs Cognito primeiro redireciona seu usuário para o endpoint de SLO que você definiu na sua configuração de IdP. Depois que seu IdP redireciona o usuário de volta para `saml2/logout`, o Amazon Cognito responde com mais um redirecionamento para `redirect_uri` ou `logout_uri` de sua solicitação. Para obter mais informações, consulte Como desconectar usuários do SAML com logout único.

O endpoint de logout não desconecta os usuários do OIDC ou dos provedores de identidade social (). IdPs Para desconectar os usuários da sessão com um IdP externo, direcione-os para a página de desconexão desse provedor.

GET /logout

O endpoint `/logout` só é compatível com `HTTPS GET`. O cliente do grupo de usuários normalmente faz essa solicitação por meio do navegador do sistema. O navegador geralmente é a guia do Chrome personalizada no Android ou no Safari View Control no iOS.

Parâmetros de solicitação

client_id

O ID do cliente dp aplicativo para o aplicativo. Para obter um ID do cliente da aplicação, é preciso registrar a aplicação no grupo de usuários. Para obter mais informações, consulte Configurações específicas da aplicação com clientes de aplicação.

Obrigatório.

logout_uri

Redirecione seu usuário para uma página de logout personalizada com um parâmetro logout_uri. Defina seu valor como o sign-out URL (URL de saída) do cliente da aplicação para o qual você deseja redirecionar o usuário depois que ele sair. Use logout_uri somente com um parâmetro client_id. Para obter mais informações, consulte Configurações específicas da aplicação com clientes de aplicação.

Você também pode usar o parâmetro logout_uri para redirecionar o usuário para a página de login de outro cliente de aplicação. Defina a página de login para o outro cliente de aplicação como um Allowed callback URL (URL de retorno de chamada permitido) em seu cliente de aplicação. Em sua solicitação para o endpoint `/logout`, defina o valor do parâmetro logout_uri para a página de login codificada em URL.

O Amazon Cognito requer um parâmetro logout_uri ou redirect_uri em sua solicitação para o endpoint `/logout`. O parâmetro logout_uri redireciona o usuário para outro site. Se os parâmetros logout_uri e redirect_uri forem incluídos em sua solicitação para o endpoint `/logout`, o Amazon Cognito utilizará exclusivamente o parâmetro logout_uri, substituindo o parâmetro redirect_uri.

nonce

(Opcional) Um valor aleatório que você pode adicionar à solicitação. O valor nonce fornecido está incluído no token de ID que o Amazon Cognito emite. Para se proteger contra ataques de repetição, a aplicação pode inspecionar a reivindicação `nonce` no token de ID e compará-la

com o que você gerou. Para obter mais informações sobre a solicitação `nonce`, consulte “ID Token Validation” (Validação de tokens de ID) no OpenID Connect Standard (Padrão do OpenID Connect).

redirect_uri

Redirecione seu usuário para sua página de login a fim de realizar a autenticação com um parâmetro redirect_uri. Defina seu valor como o Allowed callback URL (URL de retorno de chamada permitido) do cliente da aplicação para o qual você deseja redirecionar o usuário depois que ele fizer login novamente. Adicione os parâmetros client_id, scope, state e response_type que você deseja transmitir ao endpoint `/login`.

O Amazon Cognito requer um parâmetro logout_uri ou redirect_uri em sua solicitação para o endpoint `/logout`. Para redirecionar o usuário para o endpoint `/login`, a fim de reautenticar e passar tokens à sua aplicação, adicione um parâmetro redirect_uri. Se os parâmetros logout_uri e redirect_uri forem incluídos em sua solicitação para o endpoint `/logout`, o Amazon Cognito substituirá o parâmetro redirect_uri e processará o parâmetro logout_uri exclusivamente.

response_type

A resposta OAuth 2.0 que você deseja receber do Amazon Cognito após o login do usuário. `code` e `token` são os valores válidos para o parâmetro response_type.

Obrigatório quando você usa um parâmetro redirect_uri.

estado

Quando sua aplicação adiciona um parâmetro state a uma solicitação, o Amazon Cognito retorna o valor para a aplicação quando o endpoint `/oauth2/logout` redireciona o usuário.

Adicione esse valor às suas solicitações para se proteger contra ataques CSRF.

Não é possível definir o valor de um parâmetro `state` como uma string JSON codificada por URL. Para transmitir uma string que corresponda a esse formato em um parâmetro `state`, codifique-a como base64 e, depois, decodifique-a em sua aplicação.

Altamente recomendado se você usar um parâmetro redirect_uri.

scope

Os escopos OAuth 2.0 que você deseja solicitar do Amazon Cognito depois de desconectá-los com um parâmetro redirect_uri. O Amazon Cognito redireciona o usuário para o endpoint `/login` com o parâmetro scope em sua solicitação ao endpoint `/logout`.

Opcional se você usar um parâmetro redirect_uri. Se você não incluir um parâmetro scope, o Amazon Cognito redirecionará o usuário para o endpoint `/login` com um parâmetro scope. Quando o Amazon Cognito redireciona o usuário e preenche automaticamente `scope`, o parâmetro inclui todos os escopos autorizados para seu cliente de aplicação.

Exemplo de solicitações

Exemplo - fazer logout e redirecionar o usuário ao cliente

O Amazon Cognito redireciona as sessões do usuário para o URL no valor de `logout_uri`, ignorando todos os outros parâmetros da solicitação, quando as solicitações incluem e `logout_uri` e `client_id`. Esse URL deve ser um URL de logoff autorizado para o cliente da aplicação.

Veja a seguir um exemplo de solicitação de logoff e de redirecionamento para `https://www.example.com/welcome`.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/logout?
  client_id=1example23456789&
  logout_uri=https%3A%2F%2Fwww.example.com%2Fwelcome
```

Exemplo - fazer logout e solicitar que o usuário faça login como outro usuário

Quando as solicitações omitem `logout_uri`, mas fornecem os parâmetros que compõem uma solicitação bem formada para o endpoint de autorização, o Amazon Cognito redireciona os usuários para o login do login gerenciado. O endpoint de logout anexa os parâmetros em sua solicitação original ao destino do redirecionamento.

Os parâmetros adicionais que você adiciona à solicitação de logout devem estar na lista em Parâmetros de solicitação. Por exemplo, o endpoint de logout não é compatível com o redirecionamento automático do IdP com parâmetros `identity_provider` ou `idp_identifier`. O parâmetro `redirect_uri` em uma solicitação para o endpoint de logout não é um URL de logout, mas um post-sign-in URL que você deseja passar para o endpoint de autorização.

Veja a seguir um exemplo de solicitação que faz logout de um usuário, redireciona para a página de login e fornece um código de autorização para `https://www.example.com` depois do login.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/logout?
  response_type=code&
  client_id=1example23456789&
  redirect_uri=https%3A%2F%2Fwww.example.com&
```

```
  state=example-state-value&
  nonce=example-nonce-value&
  scope=openid+profile+aws.cognito.signin.user.admin
```

Provedor de identidades e endpoints de terceiros confiáveis

Os endpoints de federação são endpoints de grupos de usuários que servem ao propósito de um dos padrões de autenticação usados pelos grupos de usuários. Eles incluem SAML ACS URLs, endpoints de descoberta OIDC e endpoints de serviço para funções de grupos de usuários, tanto como provedor de identidade quanto como parte confiável. Os endpoints da federação iniciam fluxos de autenticação, recebem comprovantes de IdPs autenticação e emitem tokens para os clientes. Eles interagem com IdPs aplicativos e administradores, mas não com usuários.

Os tópicos de página inteira após esta página têm detalhes sobre os endpoints do provedor OAuth 2.0 e OIDC que ficam disponíveis quando você adiciona um domínio ao seu grupo de usuários. O gráfico a seguir é uma lista de todos os endpoints de federação.

Exemplos de domínios de grupos de usuários são:

1. Domínio de prefixo: `mydomain.auth.us-east-1.amazoncognito.com`
2. Domínio personalizado: `auth.example.com`

Endpoints de federação do grupo de usuários

URL do endpoint	Description	Como é acessado
https://*Your user pool domain*/oauth2/authorize	Redireciona um usuário para o login gerenciado ou para fazer login com seu IdP.	Invocado no navegador do cliente para iniciar a autentica ção do usuário. Consulte Autorizar endpoint.
*Your user pool domain*https://oauth2/token	Retorna tokens com base em um código de autorização ou solicitação de credenciais do cliente.	Solicitado pela aplicação para recuperar tokens. Consulte Endpoint de token.
https://*Your user pool domain*/oauth2/UserInfo	Retorna atributos do usuário com base nos escopos OAuth	Solicitado pela aplicação para recuperar o perfil do usuário. Consulte endpoint userinfo.

URL do endpoint	Description	Como é acessado
	2.0 e na identidade do usuário em um token de acesso.	
*Your user pool domain*https://oauth2/revoke	Revoga um token de atualização e os tokens de acesso associados.	Solicitado pela aplicação para revogar um token. Consulte Revogar endpoint.
https://cognito-idp.*Region*.amazonaws.com/*your user pool ID* /.well-known/openid-configuration	Um diretório da arquitetura OIDC do seu grupo de usuários. [1]	Solicitado pela aplicação para localizar metadados do emissor do grupo de usuários.
https://cognito-idp.*Region*.amazonaws.com/ /.well-known/jwks.json*your user pool ID*	Chaves públicas que você pode usar para validar os tokens do Amazon Cognito. [2]	Solicitado pelo aplicativo para verificação JWTs.
*Your user pool domain*https://oauth2/idresponse	Os provedores de identidades sociais precisam redirecionar seus usuários para esse endpoint com um código de autorização. O Amazon Cognito resgata o código para um token quando autentica seu usuário federado.	Redirecionado do login do IdP OIDC como URL de retorno de chamada do cliente IdP.
*Your user pool domain*https://saml2/idresponse	O URL do Serviço do Consumidor de Declaração (ACS) para integração com provedores de identidades SAML 2.0.	Redirecionado do IdP SAML 2.0 como URL do ACS ou o ponto de origem para login iniciado pelo IdP[3].
*Your user pool domain*https://saml2/logout	O URL de Logout único (SLO) para integração com provedores de identidades SAML 2.0.	Redirecionado do IdP SAML 2.0 como URL de logout único (SLO). Aceita somente a vinculação POST.

[1] O `openid-configuration` documento pode ser atualizado a qualquer momento com informações adicionais que mantenham o endpoint em conformidade com o OIDC e as especificações. OAuth2

[2] O arquivo JSON `jwks.json` pode ser atualizado a qualquer momento com novas chaves públicas de assinatura de token.

[3] Para obter mais informações sobre o login SAML iniciado pelo IdP, consulte. Implementar o login SAML iniciado pelo IdP

Para obter mais informações sobre o OpenID Connect e OAuth os padrões, consulte OpenID Connect 1.0 e 2.0. OAuth

Tópicos

- O endpoint de redirecionamento e autorização
- O endpoint do emissor de tokens
- O endpoint de atributos do usuário
- O endpoint de revogação do token
- O endpoint de declaração SAML do IdP

O endpoint de redirecionamento e autorização

O endpoint `/oauth2/authorize` é um endpoint de redirecionamento compatível com dois destinos de redirecionamento. Se você incluir um `identity_provider` ou `idp_identifier` no URL, ele redirecionará silenciosamente o usuário para a página de login desse provedor de identidades (IdP). Do contrário, ele redirecionará para o Endpoint de login com os mesmos parâmetros de URL que você incluiu em sua solicitação.

O endpoint de autorização redireciona para o login gerenciado ou para a página de login do IdP. O destino de uma sessão de usuário nesse endpoint é uma página da web com a qual o usuário deve interagir diretamente no navegador.

Para usar o endpoint de autorização, invoque o navegador do usuário em `/oauth2/authorize` com parâmetros que forneçam ao seu grupo de usuários os detalhes a seguir do grupo de usuários.

- O cliente da aplicação no qual você deseja fazer login.
- O URL de retorno de chamada ao qual você deseja chegar.

- Os escopos OAuth 2.0 que você deseja solicitar no token de acesso do seu usuário.
- Opcionalmente, o IdP de terceiros que você deseja usar para fazer login.

Você também pode fornecer os parâmetros `state` e `nonce` que o Amazon Cognito usa para validar as solicitações recebidas.

GET **`/oauth2/authorize`**

O endpoint `/oauth2/authorize` só é compatível com `HTTPS GET`. Sua aplicação normalmente inicia essa solicitação no navegador do usuário. Você só pode fazer solicitações ao endpoint `/oauth2/authorize` por HTTPS.

Você pode saber mais sobre a definição de endpoint de autorização no padrão do OpenID Connect (OIDC) em Authorization Endpoint (Endpoint de autorização).

Parâmetros de solicitação

`response_type`

Obrigatório.

O tipo de resposta. Precisa ser `code` ou `token`.

Uma solicitação bem-sucedida com um `response_type` do `code` retorna uma concessão de código de autorização. Uma concessão de código de autorização é um parâmetro `code` que o Amazon Cognito anexa ao URL de redirecionamento. Sua aplicação pode trocar o código por Endpoint de token para acesso, ID e tokens de atualização. Como prática recomendada de segurança e para receber tokens de atualização para os usuários, use uma concessão de código de autorização na aplicação.

Uma solicitação bem-sucedida com um `response_type` do `token` retorna uma concessão implícita. Uma concessão implícita é um ID e um token de acesso que o Amazon Cognito anexa ao URL de redirecionamento. A concessão implícita é menos segura porque expõe tokens e possíveis informações de identificação aos usuários. Você pode desativar o suporte para concessões implícitas na configuração do cliente da aplicação.

`client_id`

Obrigatório.

O ID do cliente do aplicativo

O valor de `client_id` deve ser o ID de um cliente da aplicação no grupo de usuários em que você faz a solicitação. O cliente da aplicação deve ser compatível com o login de usuários locais do Amazon Cognito ou pelo menos um IdP de terceiros.

`redirect_uri`

Obrigatório.

O URL para o qual o servidor de autenticação redireciona o navegador depois que o Amazon Cognito autoriza o usuário.

Um identificador de recurso uniforme (URI) de redirecionamento deve ter os seguintes atributos:

- Deve ser um URI absoluto.
- É necessário pré-registrar o URI em um cliente.
- Não pode incluir um componente de fragmento.

Consulte OAuth 2.0 - Endpoint de redirecionamento.

O Amazon Cognito exige que seu URI de redirecionamento use HTTPS, exceto para `http://localhost`, que você pode definir como um URL de retorno de chamada para fins de teste.

O Amazon Cognito também oferece suporte ao retorno de chamadas URLs de aplicativos, como. `myapp://example`

`state`

Opcional, recomendado.

Quando sua aplicação adiciona um parâmetro state a uma solicitação, o Amazon Cognito retorna o valor para a aplicação quando o endpoint `/oauth2/authorize` redireciona o usuário.

Adicione esse valor às suas solicitações para se proteger contra ataques CSRF.

Não é possível definir o valor de um parâmetro `state` como uma string JSON codificada por URL. Para transmitir uma string que corresponda a esse formato em um parâmetro `state`, codifique-a como Base64 e, depois, decodifique-a em sua aplicação.

`identity_provider`

Opcional.

Adicione esse parâmetro para ignorar o login gerenciado e redirecionar seu usuário para uma página de login do provedor. O valor do parâmetro identity_provider é o nome do provedor de identidade (IdP) da forma como ele aparece no grupo de usuários.

- Para provedores sociais, você pode usar os valores identity_provider`Facebook`, `Google` e `LoginWithAmazon` e `SignInWithApple`.
- Para grupos de usuários do Amazon Cognito, use o valor `COGNITO`.
- Para provedores de identidade SAML 2.0 e OpenID Connect (OIDC) (IdPs), use o nome que você atribuiu ao IdP em seu grupo de usuários.

`idp_identifier`

Opcional.

Adicione esse parâmetro para redirecionar para um provedor com um nome alternativo para o nome de identity_provider. Você pode inserir identificadores para seu SAML 2.0 e OIDC no menu de provedores sociais e externos IdPs do console do Amazon Cognito.

`scope`

Opcional.

Pode ser uma combinação de quaisquer escopos reservados ao sistema ou de escopos personalizados associados a um cliente. Os escopos devem ser separados por espaços. Os escopos reservados ao sistema são `openid`, `email`, `phone`, `profile` e `aws.cognito.signin.user.admin`. Qualquer escopo usado deve ser associado ao cliente ou ele será ignorado durante o tempo de execução.

Se o cliente não solicita qualquer escopo, o servidor de autenticação usa todos os escopos associados ao cliente.

Um token de ID só é retornado se o escopo `openid` é solicitado. O token de acesso só pode ser usado com relação a grupos de usuários do Amazon Cognito se o escopo `aws.cognito.signin.user.admin` é solicitado. Os escopos `phone`, `email` e `profile` só podem ser solicitados se o escopo `openid` também é solicitado. Esses escopos ditam as solicitações que entram no token de ID.

`code_challenge_method`

Opcional.

O protocolo de hash que você usa para gerar o desafio. O PKCE RFC define dois métodos, S256 e simples; no entanto, o servidor de autenticação do Amazon Cognito só é compatível com o S256.

`code_challenge`

Opcional.

O desafio da chave de prova para troca de código (PKCE) que você gerou por meio de `code_verifier`. Para obter mais informações, consulte Como usar PKCE em concessões de código de autorização.

Obrigatório somente quando você especifica um parâmetro `code_challenge_method`.

`nonce`

Opcional.

Um valor aleatório que você pode adicionar à solicitação. O valor nonce fornecido está incluído no token de ID que o Amazon Cognito emite. Para se proteger contra ataques de repetição, a aplicação pode inspecionar a reivindicação `nonce` no token de ID e compará-la com o que você gerou. Para obter mais informações sobre a solicitação `nonce`, consulte "ID Token Validation" (Validação de tokens de ID) no OpenID Connect Standard (Padrão do OpenID Connect).

`lang`

Opcional.

O idioma no qual você deseja exibir as páginas interativas. As páginas de login gerenciado podem ser localizadas, mas as páginas de IU hospedada (clássica) não. Para obter mais informações, consulte Managed login localization.

`login_hint`

Opcional.

Um prompt de nome de usuário que você deseja enviar ao servidor de autorização. Você pode coletar um nome de usuário, endereço de e-mail ou número de telefone do seu usuário e permitir que o provedor de destino preencha previamente o nome de login do usuário. Quando você envia um parâmetro `login_hint` e nenhum parâmetro `idp_identifier` ou `identity_provider` para o endpoint `oauth2/authorize`, o login gerenciado preenche o campo do nome de usuário com o valor da dica. Você também pode passar esse parâmetro para o Endpoint de login e preencher automaticamente o valor do nome de usuário.

Quando sua solicitação de autorização invoca um redirecionamento para o OIDC, o IdPs Amazon Cognito adiciona `login_hint` um parâmetro à solicitação para esse autorizador terceirizado.

Você não pode encaminhar dicas de login para SAML, Apple, Login With Amazon, Google ou Facebook (Meta). IdPs

`prompt`

Opcional.

Um parâmetro OIDC que controla o comportamento de autenticação para sessões existentes. Disponível somente na versão de identidade visual de login gerenciado, não na IU hospedada clássica. Para obter mais informações da especificação do OIDC, consulte Authentication request. Os valores `none` e `login` têm um efeito no comportamento de autenticação do grupo de usuários.

O Amazon Cognito encaminha todos os valores de, `prompt` exceto `none` para o seu, IdPs quando os usuários selecionam a autenticação com provedores terceirizados. Isso ocorre quando o URL que os usuários acessam inclui um parâmetro `identity_provider` ou `idp_identifier`, ou quando o servidor de autorização os redireciona para o Endpoint de login e eles selecionam um IdP nos botões disponíveis.

Valores de parâmetros de prompt

`prompt=none`

O Amazon Cognito continua silenciosamente a autenticação para usuários que têm uma sessão autenticada válida. Com esse prompt, os usuários podem se autenticar silenciosamente entre diferentes clientes da aplicação no grupo de usuários. Se o usuário ainda não estiver autenticado, o servidor de autorização retornará um erro `login_required`.

`prompt=login`

O Amazon Cognito exige que os usuários se autentiquem novamente, mesmo que já tenham uma sessão ativa. Envie esse valor quando quiser verificar a identidade do usuário novamente. Usuários autenticados que tenham uma sessão existente podem retornar ao login sem invalidar essa sessão. Quando um usuário com uma sessão ativa faz login novamente, o Amazon Cognito atribui a ele um novo cookie de sessão. Esse parâmetro também pode ser encaminhado para o seu IdPs. IdPsque aceitam esse parâmetro também solicitam uma nova tentativa de autenticação do usuário.

`prompt=select_account`

Esse valor não tem efeito no login local e deve ser enviado em solicitações que redirecionam para o. IdPs Quando incluído na solicitação de autorização, esse parâmetro adiciona `prompt=select_account` ao caminho do URL para o destino de redirecionamento do IdP.

Quando IdPs oferecem suporte a esse parâmetro, eles solicitam que os usuários selecionem a conta com a qual desejam fazer login.

`prompt=consent`

Esse valor não tem efeito no login local e deve ser enviado em solicitações que redirecionam para o. IdPs Quando incluído na solicitação de autorização, esse parâmetro adiciona `prompt=consent` ao caminho do URL para o destino de redirecionamento do IdP. Quando IdPs oferecem suporte a esse parâmetro, eles solicitam o consentimento do usuário antes de serem redirecionados de volta para seu grupo de usuários.

Quando você omite o parâmetro `prompt` da solicitação, o login gerenciado segue o comportamento padrão: os usuários devem fazer login, a menos que o navegador tenha um cookie de sessão de login gerenciado válido. Você pode combinar vários valores para `prompt` com um delimitador de espaço, por exemplo, `prompt=login consent`.

`resource`

Opcional.

O identificador de um recurso que você deseja vincular ao token de acesso na declaração `aud`. Quando você inclui esse parâmetro, o Amazon Cognito valida se o valor é um URL e define o público do token de acesso resultante para o recurso solicitado. Você pode solicitar um servidor de recursos do grupo de usuários com um identificador em formato de URL ou um URL de sua escolha. Os valores desse parâmetro devem começar com`https://`, `http://localhost` ou com um esquema de URL personalizado, como `myapp://`.

A vinculação de recursos é definida no RFC 8707. Para obter mais informações sobre servidores de recursos e vinculação de recursos, consulte Vinculação de recursos.

Exemplo: concessão de código de autorização

Este é um exemplo de solicitação de concessão de código de autorização.

A solicitação a seguir inicia uma sessão para recuperar um código de autorização que seu usuário passa para a aplicação de destino `redirect_uri`. Essa sessão solicita escopos para atributos de usuário e acesso às operações da API de autoatendimento do Amazon Cognito.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
client_id=1example23456789&
redirect_uri=https://www.example.com&
```

```
state=abcdefg&
scope=openid+profile+aws.cognito.signin.user.admin
```

O servidor de autenticação do Amazon Cognito faz o redirecionamento de volta à aplicação com o estado e o código de autorização. O código de autorização é válido por cinco minutos.

```
HTTP/1.1 302 Found
Location: https://www.example.com?code=a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111&state=abcdefg
```

Exemplo: concessão de código de autorização com PKCE

Este fluxo de exemplo realiza uma concessão de código de autorização com PKCE.

Esta solicitação adiciona um parâmetro `code_challenge`. Para concluir a troca de um código por um token, você deve incluir o parâmetro `code_verifier` em sua solicitação para o endpoint `/oauth2/token`.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=aws.cognito.signin.user.admin&
code_challenge_method=S256&
code_challenge=a1b2c3d4...
```

O servidor de autorização redireciona de volta à aplicação com o estado e o código de autorização. Sua aplicação processa o código de autorização e o troca por tokens.

```
HTTP/1.1 302 Found
Location: https://www.example.com?code=a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111&state=abcdefg
```

Exemplo: exigência de reautenticação com **`prompt=login`**

A solicitação a seguir adiciona um parâmetro `prompt=login` que exige que o usuário se autentique novamente, mesmo que tenha uma sessão ativa.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
```

```
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=openid+profile+aws.cognito.signin.user.admin&
prompt=login
```

O servidor de autorização redireciona para o endpoint de login, exigindo uma nova autenticação.

```
HTTP/1.1 302 Found Location: https://mydomain.auth.us-east-1.amazoncognito.com/
login?response_type=code&client_id=1example23456789&redirect_uri=https://
www.example.com&state=abcdefg&scope=openid+profile
+aws.cognito.signin.user.admin&prompt=login
```

Exemplo: autenticação silenciosa com **`prompt=none`**

A solicitação a seguir adiciona um parâmetro `prompt=none` que verifica silenciosamente se o usuário tem uma sessão válida.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=openid+profile+aws.cognito.signin.user.admin&
prompt=none
```

Quando não há uma sessão válida, o servidor de autorização retorna um erro ao URI de redirecionamento.

```
HTTP/1.1 302 Found Location: https://www.example.com?error=login_required&state=abcdefg
```

Quando há uma sessão válida, o servidor de autorização retorna um código de autorização.

```
HTTP/1.1 302 Found Location: https://www.example.com?
code=AUTHORIZATION_CODE&state=abcdefg
```

Exemplo: concessão de código de autorização com vinculação de recursos

A solicitação a seguir adiciona um parâmetro `resource` para vincular o token de acesso a um servidor de recursos específico. O token de acesso resultante cria as condições para que a API de destino valide que é o público-alvo da solicitação do usuário autenticado.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=solar-system-data-api.example.com/asteroids.add&
resource=https://solar-system-data-api.example.com
```

O servidor de autorização retorna um código de autorização que resulta em um token de acesso com uma declaração aud de `https://solar-system-data-api.example.com`.

```
HTTP/1.1 302 Found Location: https://www.example.com?
code=AUTHORIZATION_CODE&state=abcdefg
```

Exemplo: concessão de token (implícita) sem escopo **`openid`**

Esse exemplo de fluxo gera uma concessão implícita e retorna JWTs diretamente para a sessão do usuário.

A solicitação é para uma concessão implícita do seu servidor de autorização. Ela solicita escopos no token de acesso que autorizam as operações de autoatendimento do perfil do usuário.

```
GET https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=token&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=aws.cognito.signin.user.admin
```

O servidor de autorização redireciona de volta à aplicação somente com um token de acesso. Como o escopo `openid` não foi solicitado, o Amazon Cognito não retorna um token de ID. Além disso, o Amazon Cognito não retorna um token de atualização nesse fluxo.

```
HTTP/1.1 302 Found
Location: https://example.com/
callback#access_token=eyJra456defEXAMPLE&token_type=bearer&expires_in=3600&state=STATE
```

Exemplo: concessão de token (implícita) com escopo **`openid`**

Este fluxo de exemplo gera uma concessão implícita e retorna tokens para o navegador do usuário.

A solicitação é para uma concessão implícita do seu servidor de autorização. Ela solicita escopos no token de acesso que autorizam o acesso a atributos do usuário e operações de autoatendimento.

```
GET
https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=token&
client_id=1example23456789&
redirect_uri=https://www.example.com&
state=abcdefg&
scope=aws.cognito.signin.user.admin+openid+profile
```

O servidor de autorização redireciona de volta à aplicação com token de acesso e token de ID (porque o escopo `openid` foi incluído):

```
HTTP/1.1 302 Found
Location: https://
www.example.com#id_token=eyJra67890EXAMPLE&access_token=eyJra12345EXAMPLE&token_type=bearer&exp
```

Exemplos de respostas negativas

O Amazon Cognito pode negar sua solicitação. As solicitações negativas vêm com um código de erro HTTP e uma descrição que você pode usar para corrigir os parâmetros da solicitação. Veja a seguir exemplos de respostas negativas.

- Se `client_id` e `redirect_uri` forem válidos, mas os parâmetros da solicitação não estiverem formatados corretamente, o servidor de autenticação redirecionará o erro para o `redirect_uri` do cliente e anexará uma mensagem de erro em um parâmetro de URL. Veja a seguir exemplos de formatos incorretos.
  - A solicitação não inclui um parâmetro `response_type`.
  - A solicitação de autorização forneceu um parâmetro `code_challenge`, mas não um parâmetro `code_challenge_method`.
  - O valor do parâmetro `code_challenge_method` não é `S256`.

  Veja a seguir um exemplo de resposta para a solicitação com formato incorreto.

  ```
  HTTP 1.1 302 Found Location: https://client_redirect_uri?error=invalid_request
  ```

- Se o cliente solicitar `code` ou `token` em `response_type`, mas não tiver permissão para essas solicitações, o servidor de autorização do Amazon Cognito retornará `unauthorized_client` ao `redirect_uri` do cliente da seguinte forma:

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?error=unauthorized_client
```

- Se o cliente solicitar um escopo inválido, desconhecido ou malformado, o servidor de autorização do Amazon Cognito deverá retornar o `invalid_scope` ao `redirect_uri` do cliente da seguinte forma:

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?error=invalid_scope
```

- Se acontece um erro inesperado no servidor, o servidor de autenticação retorna `server_error` ao `redirect_uri` do cliente. Como o erro HTTP 500 não é enviado ao cliente, ele não aparece no navegador do usuário. O servidor de autorização retorna o erro a seguir.

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?error=server_error
```

- Quando o Amazon Cognito se autentica por meio de federação para terceiros, IdPs o Amazon Cognito pode enfrentar problemas de conexão, como os seguintes:
  - Se ocorrer um tempo limite de conexão ao solicitar o token do IdP, o servidor de autenticação redirecionará o erro para o `redirect_uri` do cliente da seguinte maneira:

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?
error=invalid_request&error_description=Timeout+occurred+in+calling+IdP+token
+endpoint
```

  - Se ocorrer um tempo limite de conexão na chamada do endpoint `jwks_uri` para validação do token de ID, o servidor de autenticação redirecionará o erro para o `redirect_uri` do cliente da seguinte maneira:

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?
error=invalid_request&error_description=error_description=Timeout+in+calling+jwks
+uri
```

- Ao se autenticar por meio de federação com terceiros IdPs, os provedores podem retornar respostas de erro. Isso pode acontecer em razão de erros de configuração ou outros motivos, como os seguintes:

- Se uma resposta de erro for recebida de outros provedores, o servidor de autenticação redirecionará o erro para o `redirect_uri` do cliente da seguinte maneira:

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?
error=invalid_request&error_description=[IdP name]+Error+-+[status code]+error
 getting token
```

- Se uma resposta de erro for recebida do Google, o servidor de autenticação redirecionará o erro para o `redirect_uri` do cliente da seguinte maneira:

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?
error=invalid_request&error_description=Google+Error+-+[status code]+[Google-
provided error code]
```

- Quando o Amazon Cognito encontra uma exceção de comunicação com um IdP externo, o servidor de autenticação redireciona o erro para o `redirect_uri` do cliente com uma das seguintes mensagens:
  - ```
    HTTP 1.1 302 Found Location: https://client_redirect_uri?
    error=invalid_request&error_description=Connection+reset
    ```
  - ```
    HTTP 1.1 302 Found Location: https://client_redirect_uri?
    error=invalid_request&error_description=Read+timed+out
    ```

O endpoint do emissor de tokens

O endpoint do token OAuth 2.0 `/oauth2/token` emite tokens web JSON (JWTs) para aplicativos que desejam concluir fluxos de concessão de código de autorização e credenciais de cliente. Esses tokens são o resultado da autenticação com um grupo de usuários. Eles contêm informações sobre o usuário (token de ID), o nível de acesso do usuário (token de acesso) e o direito do usuário de persistir na sessão conectada (token de atualização). As bibliotecas independentes do OpenID Connect (OIDC) gerenciam cargas úteis de solicitações e respostas desse endpoint. Os tokens fornecem prova verificável de autenticação, informações de perfil e um mecanismo para acesso a sistemas de backend.

Seu servidor de autorização do grupo de usuários OAuth 2.0 emite tokens web JSON (JWTs) do endpoint do token para os seguintes tipos de sessões:

1. Usuários que concluíram uma solicitação de concessão de código de autorização. O resgate bem-sucedido de um código retorna tokens de ID, acesso e atualização.
2. Machine-to-machine Sessões (M2M) que concluíram uma concessão de credenciais de cliente. A autorização bem-sucedida com o segredo do cliente retorna um token de acesso.
3. Usuários que já fizeram login e receberam tokens de atualização. A autenticação de token de atualização retorna novos tokens de ID e acesso.

> **Note**
>
> Os usuários que fazem login com uma concessão de código de autorização no login gerenciado ou por meio da federação sempre podem atualizar seus tokens por meio do endpoint de token. Usuários que fazem login com as operações da API `InitiateAuth` e `AdminInitiateAuth` podem atualizar seus tokens com o endpoint do token quando os dispositivos memorizados não estão ativos em seu grupo de usuários. Se os dispositivos memorizados estiverem ativos, atualize os tokens com a API relevante ou a operação de atualização de token do SDK para seu cliente de aplicação.

O endpoint do token fica disponível ao público quando você adiciona um domínio ao grupo de usuários. Ele aceita solicitações HTTP POST. Para fins de segurança da aplicação, use o PKCE com eventos de login com código de autorização. O PKCE verifica se o usuário que está transmitindo um código de autorização é o mesmo usuário que se autenticou. Para obter mais informações sobre PKCE, consulte IETF RFC 7636.

Você pode aprender mais sobre os clientes do aplicativo do grupo de usuários e seus tipos de concessão, segredos do cliente, escopos permitidos e clientes IDs emConfigurações específicas da aplicação com clientes de aplicação. Você pode aprender mais sobre autorização M2M, concessões de credenciais de clientes e autorização com escopos de token de acesso em Escopos, M2M e servidores de recursos.

Para recuperar informações sobre um usuário por meio do token de acesso, transmita-o para endpoint userinfo ou para uma solicitação de API GetUser. O token de acesso deve conter os escopos apropriados para essas solicitações.

Formatar uma solicitação POST para o endpoint de token

O endpoint `/oauth2/token` só é compatível com `HTTPS POST`. Esse endpoint não é interativo com o usuário. Gerencie solicitações de token com uma biblioteca OpenID Connect (OIDC) em sua aplicação.

O endpoint de token é compatível com a autenticação de `client_secret_basic` e `client_secret_post`. Para obter mais informações sobre a especificação do OIDC, consulte Client Authentication. Para obter mais informações sobre o endpoint de token na especificação do OpenID Connect, consulte Endpoint de token.

Parâmetros de solicitação no cabeçalho

Você pode transmitir os parâmetros a seguir no cabeçalho da sua solicitação para o endpoint de token.

`Authorization`

Se um segredo foi emitido para o cliente, ele precisa passar o `client_id` e o `client_secret` no cabeçalho de autorização como autorização HTTP `client_secret_basic`. Você também pode incluir o `client_id` e `client_secret` no corpo da solicitação como autorização de `client_secret_post`.

A string do cabeçalho de autorização é Basic `Base64Encode(client_id:client_secret)`. O exemplo a seguir é um cabeçalho de autorização para o cliente da aplicação `djc98u3jiedmi283eu928` com segredo do cliente `abcdef01234567890` usando a versão codificada em Base64 da string `djc98u3jiedmi283eu928:abcdef01234567890`:

```
Authorization: Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw
```

`Content-Type`

Defina o valor desse parâmetro como `'application/x-www-form-urlencoded'`.

Parâmetros de solicitação no corpo

A seguir estão os parâmetros que você pode solicitar em formato `x-www-form-urlencoded` no corpo da solicitação para o endpoint de token.

`grant_type`

Obrigatório.

O tipo de concessão do OIDC que você deseja solicitar.

Deve ser `authorization_code` ou `refresh_token` ou `client_credentials`. Você pode solicitar um token de acesso para um escopo personalizado a partir do endpoint do token sob as seguintes condições:

- Você habilitou o escopo solicitado na configuração do cliente da aplicação.
- Você configurou o cliente da aplicação com um segredo do cliente.
- Você ativa a concessão de credenciais do cliente em seu cliente de aplicação.

> **Note**
>
> O endpoint de token retorna um token de atualização somente quando o `grant_type` é `authorization_code`.

`client_id`

Opcional. Não é obrigatório quando você fornece o ID do cliente de aplicação no cabeçalho *`Authorization`*.

O ID de um cliente de aplicação no grupo de usuários. Especifique o mesmo cliente de aplicação que autenticou o usuário.

Você deve fornecer esse parâmetro se o cliente for público e não tiver um segredo ou com `client_secret` na autorização `client_secret_post`.

`client_secret`

Opcional. Não é obrigatório quando você fornece o segredo do cliente no cabeçalho *`Authorization`* e quando o cliente de aplicação não tem um segredo.

O segredo do cliente de aplicação, caso o cliente de aplicação tenha um, para autorização `client_secret_post`.

`scope`

Opcional.

Pode ser uma combinação de quaisquer escopos associados ao seu cliente de aplicação. O Amazon Cognito ignora escopos na solicitação que não são permitidos para o cliente de aplicação solicitado. Se você não fornecer esse parâmetro de solicitação, o servidor de autorização retornará uma declaração `scope` de token de acesso com todos os escopos de autorização que você habilitou na configuração do cliente de aplicação. É possível solicitar qualquer um dos escopos permitidos para o cliente de aplicação solicitado: escopos padrão, escopos personalizados de servidores de recursos e o escopo de autoatendimento do usuário `aws.cognito.signin.user.admin`.

`redirect_uri`

Opcional. Não é obrigatório para concessões de credenciais de clientes.

Precisa ser o mesmo `redirect_uri` usado para obter o `authorization_code` em `/oauth2/authorize`.

Você deve fornecer esse parâmetro se `grant_type` for `authorization_code`.

`refresh_token`

Opcional. Usado somente quando o usuário já tem um token de atualização e deseja obter um novo ID e tokens de acesso.

Para gerar novos tokens de acesso e ID para a sessão de um usuário, defina o valor de `refresh_token` para um token de atualização válido emitido pelo cliente de aplicação solicitado.

Retorna um novo token de atualização com um novo token de ID e acesso quando a alternância de tokens de atualização está ativa, caso contrário, retorna somente tokens de ID e acesso. Se o token de acesso original estiver vinculado a um recurso da API, o novo token de acesso manterá o URL da API solicitado na declaração `aud`.

`code`

Opcional. Obrigatório somente em concessões de código de autorização.

O código de autorização de uma concessão de código de autorização. Você deve fornecer esse parâmetro se sua solicitação de autorização incluir `grant_type` de `authorization_code`.

`aws_client_metadata`

Opcional.

Informações que você deseja passar para os fluxos de autorização Acionador do Lambda antes da geração do token in machine-to-machine (M2M). Sua aplicação pode coletar informações de contexto sobre a sessão e transmiti-las neste parâmetro. Quando você transmite `aws_client_metadata` no formato JSON codificado por URL, o Amazon Cognito o inclui no evento de entrada para sua função do Lambda do acionador. Sua versão do evento de acionador de pré-geração de tokens ou a versão global de acionador do Lambda deve ser configurada para a versão três ou posterior. Embora o Amazon Cognito aceite solicitações para este endpoint em fluxos M2M de código de autorização e credenciais do cliente, seu grupo de usuários só transmite `aws_client_metadata` para o acionador de pré-geração de tokens por meio de solicitações de credenciais do cliente.

`code_verifier`

Opcional. Obrigatório somente se você tiver fornecido os parâmetros `code_challenge_method` e `code_challenge` em sua solicitação de autorização inicial.

O verificador de código gerado que sua aplicação usou para calcular `code_challenge` em uma solicitação de concessão de código de autorização com PKCE.

Como trocar um código de autorização por tokens

A solicitação a seguir gera tokens de ID, acesso e atualização com sucesso após a autenticação com uma concessão de código de autorização. A solicitação transmite o segredo do cliente no formato `client_secret_basic` no cabeçalho `Authorization`.

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token&
Content-Type='application/x-www-form-urlencoded'&
Authorization=Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

grant_type=authorization_code&
client_id=1example23456789&
code=AUTHORIZATION_CODE&
redirect_uri=com.myclientapp://myclient/redirect
```

A resposta emite novos tokens de ID, acesso e atualização para o usuário, com metadados adicionais.

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "refresh_token": "eyJj3example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Credenciais do cliente com autorização básica

A solicitação a seguir de uma aplicação M2M solicita a concessão de credenciais do cliente. Como as credenciais do cliente exigem um segredo do cliente, a solicitação é autorizada com um cabeçalho `Authorization` derivado do ID e do segredo do cliente de aplicação. A solicitação resulta em um token de acesso com os dois escopos solicitados. A solicitação também inclui metadados do cliente que fornecem informações de endereço IP e um token emitido para o usuário atribuído a essa concessão. O Amazon Cognito transmite os metadados do cliente para o acionador do Lambda de pré-geração de tokens.

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token >
Content-Type='application/x-www-form-urlencoded'&
Authorization=Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

grant_type=client_credentials&
client_id=1example23456789&
scope=resourceServerIdentifier1%2Fscope1%20resourceServerIdentifier2%2Fscope2&
&aws_client_metadata=%7B%22onBehalfOfToken%22%3A%22eyJra789ghiEXAMPLE%22,
%20%22ClientIpAddress%22%3A%22192.0.2.252%22%7D
```

O Amazon Cognito transmite o evento de entrada a seguir para o acionador do Lambda de pré-geração de tokens.

```
{
    version: '3',
    triggerSource: 'TokenGeneration_ClientCredentials',
    region: 'us-east-1',
    userPoolId: 'us-east-1_EXAMPLE',
    userName: 'ClientCredentials',
    callerContext: {
        awsSdkVersion: 'aws-sdk-unknown-unknown',
        clientId: '1example23456789'
    },
    request: {
```

```
        userAttributes: {},
        groupConfiguration: null,
        scopes: [
           'resourceServerIdentifier1/scope1',
           'resourceServerIdentifier2/scope2'
        ],
        clientMetadata: {
            'onBehalfOfToken': 'eyJra789ghiEXAMPLE',
            'ClientIpAddress': '192.0.2.252'
        }
    },
    response: { claimsAndScopeOverrideDetails: null }
}
```

A resposta retorna um token de acesso. As concessões de credenciais do cliente são para autorização machine-to-machine (M2M) e retornam apenas tokens de acesso.

```
HTTP/1.1 200 OK
Content-Type: application/json
{
    "access_token": "eyJra1example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Credenciais do cliente com autorização do corpo POST

A solicitação de concessão de credenciais do cliente a seguir inclui o parâmetro `client_secret` no corpo da solicitação e não inclui um cabeçalho `Authorization`. Essa solicitação usa a sintaxe de autorização `client_secret_post`. A solicitação resulta em um token de acesso com o escopo solicitado. A solicitação também inclui metadados do cliente que fornecem informações de endereço IP e um token emitido para o usuário atribuído a essa concessão. O Amazon Cognito transmite os metadados do cliente para o acionador do Lambda de pré-geração de tokens.

```
POST /oauth2/token HTTP/1.1
Content-Type: application/x-www-form-urlencoded
X-Amz-Target: AWSCognitoIdentityProviderService.Client credentials request
User-Agent: USER_AGENT
Accept: /
Accept-Encoding: gzip, deflate, br
Content-Length: 177
```

```
Referer: http://auth.example.com/oauth2/token
Host: auth.example.com
Connection: keep-alive

grant_type=client_credentials&
client_id=1example23456789&
scope=my_resource_server_identifier%2Fmy_custom_scope&
client_secret=9example87654321&
aws_client_metadata=%7B%22onBehalfOfToken%22%3A%22eyJra789ghiEXAMPLE%22,
%20%22ClientIpAddress%22%3A%22192.0.2.252%22%7D
```

O Amazon Cognito transmite o evento de entrada a seguir para o acionador do Lambda de pré-geração de tokens.

```
{
    version: '3',
    triggerSource: 'TokenGeneration_ClientCredentials',
    region: 'us-east-1',
    userPoolId: 'us-east-1_EXAMPLE',
    userName: 'ClientCredentials',
    callerContext: {
        awsSdkVersion: 'aws-sdk-unknown-unknown',
        clientId: '1example23456789'
    },
    request: {
        userAttributes: {},
        groupConfiguration: null,
        scopes: [
           'resourceServerIdentifier1/my_custom_scope'
        ],
        clientMetadata: {
            'onBehalfOfToken': 'eyJra789ghiEXAMPLE',
            'ClientIpAddress': '192.0.2.252'
        }
    },
    response: { claimsAndScopeOverrideDetails: null }
}
```

A resposta retorna um token de acesso. As concessões de credenciais do cliente são para autorização machine-to-machine (M2M) e retornam apenas tokens de acesso.

```
HTTP/1.1 200 OK
```

```
Content-Type: application/json;charset=UTF-8
Date: Tue, 05 Dec 2023 16:11:11 GMT
x-amz-cognito-request-id: 829f4fe2-a1ee-476e-b834-5cd85c03373b

{
    "access_token": "eyJra12345EXAMPLE",
    "expires_in": 3600,
    "token_type": "Bearer"
}
```

Concessão de código de autorização com PKCE

O exemplo a seguir conclui uma solicitação de autorização que incluiu os parâmetros `code_challenge_method` e `code_challenge` em uma solicitação de concessão de código de autorização com PKCE.

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token
Content-Type='application/x-www-form-urlencoded'&
Authorization=Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

grant_type=authorization_code&
client_id=1example23456789&
code=AUTHORIZATION_CODE&
code_verifier=CODE_VERIFIER&
redirect_uri=com.myclientapp://myclient/redirect
```

A resposta retorna tokens de ID, acesso e atualização da verificação bem-sucedida do PKCE pela aplicação.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "refresh_token": "eyJj3example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Atualização de token sem alternância de tokens de atualização

O exemplo de solicitações a seguir fornece um token de atualização para um cliente de aplicação no qual a alternância de tokens de atualização está inativa. Como o cliente de aplicação tem um segredo do cliente, a solicitação fornece um cabeçalho `Authorization`.

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token >
Content-Type='application/x-www-form-urlencoded'&
Authorization=Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

grant_type=refresh_token&
client_id=1example23456789&
refresh_token=eyJj3example
```

A resposta retorna novos tokens de ID e acesso.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Atualização de token com alternância de tokens de atualização

O exemplo de solicitações a seguir fornece um token de atualização para um cliente de aplicação no qual a alternância de tokens de atualização está ativa. Como o cliente de aplicação tem um segredo do cliente, a solicitação fornece um cabeçalho `Authorization`.

```
POST https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/token >
Content-Type='application/x-www-form-urlencoded'&
Authorization=Basic ZGpjOTh1M2ppZWRtaTI4M2V1OTI4OmFiY2RlZjAxMjM0NTY3ODkw

grant_type=refresh_token&
client_id=1example23456789&
refresh_token=eyJj3example
```

A resposta retorna novos tokens de ID, acesso e atualização.

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "access_token": "eyJra1example",
    "id_token": "eyJra2example",
    "refresh_token": "eyJj4example",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

Exemplos de respostas negativas

Solicitações malformadas geram erros no endpoint de token. Veja a seguir um mapa geral do corpo da resposta quando as solicitações de token geram um erro.

```
HTTP/1.1 400 Bad Request
Content-Type: application/json;charset=UTF-8

{
"error":"invalid_request|invalid_client|invalid_grant|unauthorized_client|
unsupported_grant_type"
}
```

`invalid_request`

A solicitação não tem um parâmetro obrigatório, inclui um valor de parâmetro não compatível (diferente de `unsupported_grant_type`) ou está malformado. Por exemplo, `grant_type` é `refresh_token` , mas `refresh_token` não está incluído.

`invalid_client`

Falha na autenticação do cliente. Por exemplo, quando o cliente inclui `client_id` e `client_secret` no cabeçalho de autorização, mas não há tal cliente com esse `client_id` e `client_secret`.

`invalid_grant`

O token de atualização foi revogado.

O código de autorização já foi consumido ou não existe.

O cliente da aplicação não tem acesso de leitura a todos os atributos no escopo solicitado. Por exemplo, a aplicação solicita o escopo `email` e o cliente da aplicação consegue ler o atributo `email`, mas não `email_verified`.

`unauthorized_client`

O cliente não tem permissão para fluxo de concessão de código ou para tokens de atualização.

`unsupported_grant_type`

Retornado se `grant_type` for diferente de `authorization_code`, `refresh_token` ou `client_credentials`.

O endpoint de atributos do usuário

Quando o OIDC emite tokens de ID que contêm atributos do usuário, o OAuth 2.0 implementa o endpoint. `/oauth2/userInfo` Um usuário ou cliente autenticado recebe um token de acesso com uma reivindicação `scopes`. Essa reivindicação determina os atributos que o servidor de autorização deve retornar. Quando uma aplicação apresenta um token de acesso ao endpoint `userInfo`, o servidor de autorização retorna um corpo de resposta que contém os atributos do usuário que estão dentro dos limites definidos pelos escopos do token de acesso. Essa aplicação pode recuperar informações sobre um usuário a partir do endpoint `userInfo`, desde que tenha um token de acesso válido com pelo menos uma reivindicação de escopo `openid`.

O endpoint `userInfo` é um endpoint userInfo do OpenID Connect (OIDC). Ele responde com atributos do usuário quando os provedores de serviço apresentam os tokens de acesso que seu endpoint do emitiu. Os escopos no token de acesso do usuário definem os atributos do usuário que o endpoint userInfo retorna em sua resposta. O escopo `openid` deve ser uma das reivindicações do token de acesso.

O Amazon Cognito emite tokens de acesso em resposta a solicitações de API dos grupos de usuários, como InitiateAuth. Como elas não contêm escopos, o endpoint userInfo não aceita esses tokens de acesso. Em vez disso, você deve apresentar os tokens de acesso do endpoint de token.

Seu provedor de identidade terceirizado (IdP) OAuth 2.0 também hospeda um userInfo endpoint. Quando o usuário faz a autenticação com esse IdP, o Amazon Cognito troca silenciosamente um código de autorização com o endpoint `token` do IdP. Seu grupo de usuários passa o token de acesso do IdP para autorizar a recuperação das informações do usuário do endpoint `userInfo` do IdP.

Os escopos no token de acesso de um usuário são determinados pelo parâmetro de solicitação `scopes` nas solicitações de autenticação ou pelos escopos que o acionador do Lambda de pré-geração de tokens adiciona. Você pode decodificar tokens de acesso e examinar as declarações `scope` para ver os escopos de controle de acesso que elas contêm. A seguir estão algumas combinações de escopo que influenciam os dados retornados do endpoint `userInfo`. O escopo reservado do Amazon Cognito `aws.cognito.signin.user.admin` não afeta os dados retornados desse endpoint.

Exemplos de escopos no token de acesso e seus efeitos na resposta **`userInfo`**

`openid`

Retorna uma resposta com todos os atributos do usuário que o cliente de aplicação pode ler.

`openid profile`

Retorna os atributos do usuário `name`, `family_name`, `given_name`, `middle_name`, `nickname`, `preferred_username`, `profile`, `picture`, `website`, `gender`, `birthdate`, `zoneinfo`, `locale` e `updated_at`. Também retorna atributos personalizados. Em clientes da aplicação que não têm acesso de leitura a cada atributo, a resposta a esse escopo inclui todos os atributos da especificação aos quais o cliente de aplicação tem acesso de leitura.

`openid email`

Retorna informações básicas do perfil e os atributos `email` e `email_verified`.

`openid phone`

Retorna informações básicas do perfil e os atributos `phone_number` e `phone_number_verified`.

GET /oauth2/userInfo

A aplicação gera solicitações para este endpoint diretamente, não por meio de um navegador.

Para ter mais informações, consulte Endpoint UserInfo na especificação do OpenID Connect (OIDC).

Tópicos

- Parâmetros de solicitação no cabeçalho
- Exemplo - solicitação

- Exemplo - resposta positiva
- Exemplo - respostas negativas

Parâmetros de solicitação no cabeçalho

`Authorization: Bearer <access_token>`

Repasse o token de acesso no campo do cabeçalho da autorização.

Obrigatório.

Exemplo - solicitação

```
GET /oauth2/userInfo HTTP/1.1
Content-Type: application/x-amz-json-1.1
Authorization: Bearer eyJra12345EXAMPLE
User-Agent: [User agent]
Accept: */*
Host: auth.example.com
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

Exemplo - resposta positiva

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Content-Length: [Integer]
Date: [Timestamp]
x-amz-cognito-request-id: [UUID]
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Pragma: no-cache
Expires: 0
Strict-Transport-Security: max-age=31536000 ; includeSubDomains
X-Frame-Options: DENY
Server: Server
Connection: keep-alive
{
    "sub": "[UUID]",
    "email_verified": "true",
```

```
    "custom:mycustom1": "CustomValue",
    "phone_number_verified": "true",
    "phone_number": "+12065551212",
    "email": "bob@example.com",
    "username": "bob"
}
```

Para obter uma lista de solicitações OIDC, consulte Solicitações padrão. No momento, o Amazon Cognito retorna os valores para `email_verified` e `phone_number_verified` como strings.

Exemplo - respostas negativas

Exemplo - solicitação inválida

```
HTTP/1.1 400 Bad Request
WWW-Authenticate: error="invalid_request",
error_description="Bad OAuth2 request at UserInfo Endpoint"
```

`invalid_request`

A solicitação não possui um parâmetro obrigatório, inclui um valor de parâmetro não compatível ou contém informações incorretas.

Exemplo - token inválido

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: error="invalid_token",
error_description="Access token is expired, disabled, or deleted, or the user has
 globally signed out."
```

`invalid_token`

O token de acesso está expirado, revogado, informado incorretamente ou inválido.

O endpoint de revogação do token

Os usuários que têm um token de atualização em sua sessão têm algo semelhante a um cookie de navegador. Eles podem renovar a sessão existente, desde que o token de atualização seja válido. Em vez de solicitar que o usuário faça login após a expiração do ID ou do token de acesso,

a aplicação pode usar o token de atualização para obter tokens novos e válidos. No entanto, você pode determinar externamente que a sessão de um usuário seja encerrada, ou o usuário pode optar por esquecer a sessão atual. Nesse ponto, você pode revogar esse token de atualização para que eles não possam mais persistir na sessão.

O endpoint `/oauth2/revoke` revoga o token de acesso de um usuário que o Amazon Cognito emitiu inicialmente com o token de atualização fornecido por você. Esse endpoint também revoga o próprio token de atualização e todos os tokens de acesso e identidade subsequentes do mesmo token de atualização. Depois que o endpoint revogar os tokens, você não poderá usar os tokens de acesso revogados para acessar a autenticação dos tokens do Amazon APIs Cognito.

POST /oauth2/revoke

O endpoint `/oauth2/revoke` só é compatível com `HTTPS POST`. O cliente do grupo de usuários faz solicitações para esse endpoint diretamente e não por meio do navegador do sistema.

Parâmetros de solicitação no cabeçalho

`Authorization`

Se o cliente da aplicação tiver recebido um segredo, a aplicação precisará passar o `client_id` e o `client_secret` no cabeçalho da autorização por meio da autorização HTTP básica. O segredo é https://en.wikipedia.org/wiki/Basic_access_authentication#Client_sidebásico`Base64Encode(client_id:client_secret)`.

`Content-Type`

Precisa ser sempre `'application/x-www-form-urlencoded'`.

Parâmetros de solicitação no corpo

`token`

(Obrigatório) O token de atualização que o cliente quer revogar. A solicitação também revoga todos os tokens de acesso que o Amazon Cognito emitiu com esse token de atualização.

Obrigatório.

`client_id`

(Opcional) O ID do cliente da aplicação para o token que você deseja revogar.

Obrigatório se o cliente for público e não tiver um segredo.

Exemplos de solicitação de revogação

Esta solicitação revoga um token de atualização para um cliente de aplicação que não tem segredo de cliente. O parâmetro `client_id` contém o corpo da solicitação.

```
POST /oauth2/revoke HTTP/1.1
Host: mydomain.auth.us-east-1.amazoncognito.com
Accept: application/json
Content-Type: application/x-www-form-urlencoded
token=2YotnFZFEjr1zCsicMWpAA&
client_id=1example23456789
```

Esta solicitação revoga um token de atualização para um cliente de aplicação que tem um segredo de cliente. Observe que o cabeçalho `Authorization` contém um ID de cliente e um segredo de cliente codificados, mas nenhum `client_id` no corpo da solicitação.

```
POST /oauth2/revoke HTTP/1.1
Host: mydomain.auth.us-east-1.amazoncognito.com
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW
token=2YotnFZFEjr1zCsicMWpAA
```

Resposta de erro de revogação

Uma resposta bem-sucedida contém um corpo vazio. A resposta de erro é um objeto JSON com um campo `error` e, em alguns casos, um campo `error_description`.

Erros de endpoint

- Se o token não estiver presente na solicitação ou se o recurso estiver desabilitado para o cliente da aplicação, você receberá HTTP 400 e o erro `invalid_request`.
- Se o token que o Amazon Cognito enviou na solicitação de revogação não for um token de atualização, você receberá um HTTP 400 e um erro `unsupported_token_type`.
- Se as credenciais do cliente não forem válidas, você receberá um HTTP 401 e um erro `invalid_client`.

- Se o token tiver sido revogado ou se o cliente tiver enviado um token que não é válido, você receberá um HTTP 200 OK.

O endpoint de declaração SAML do IdP

O `/saml2/idpresponse` recebe declarações de SAML. No login service-provider-initiated (iniciado pelo SP), seu aplicativo não interage diretamente com esse endpoint — seu provedor de identidade (IdP) do SAML 2.0 redireciona seu usuário aqui com a resposta do SAML. Para login iniciado pelo SP, configure seu IdP com o caminho para `saml2/idpresponse` como URL do serviço de consumidor de declaração (ACS). Para obter mais informações sobre o início da sessão, consulte Iniciação de sessão SAML em grupos de usuários do Amazon Cognito.

No login iniciado pelo IdP, invoque solicitações para esse endpoint em sua aplicação depois de fazer login como usuário com seu provedor SAML 2.0. Seus usuários fazem login com seu IdP no navegador e, em seguida, a aplicação coleta a declaração SAML e a envia para esse endpoint. Você deve enviar declarações SAML no corpo de uma solicitação `HTTP POST` por HTTPS. O corpo da sua solicitação `POST` deve ser um parâmetro `SAMLResponse` e um parâmetro `Relaystate`. Para obter mais informações, consulte Implementar o login SAML iniciado pelo IdP.

O endpoint `saml2/idpresponse` pode aceitar declarações SAML de até 100.000 caracteres.

POST **`/saml2/idpresponse`**

Para usar o endpoint `/saml2/idpresponse` em um login iniciado por IdP, gere uma solicitação POST com parâmetros que forneçam ao seu grupo de usuários os detalhes da sessão do usuário.

- O cliente da aplicação no qual ele deseja fazer login.
- O URL de retorno de chamada ao qual ele deseja chegar.
- Os escopos OAuth 2.0 que eles desejam solicitar no token de acesso do seu usuário.
- O IdP que iniciou a solicitação de login.

Parâmetros do corpo da solicitação iniciados pelo IdP

SAMLResponse

Uma declaração SAML codificada em Base64 de um IdP associado a um cliente de aplicação válido e a uma configuração de IdP em seu grupo de usuários.

RelayState

Um parâmetro `RelayState` contém os parâmetros de solicitação que, de outra forma, você passaria para o endpoint `oauth2/authorize`. Para mais informações sobre esses parâmetros, consulte Autorizar endpoint.

response_type

O tipo de subsídio OAuth 2.0.

client_id

O ID do cliente do aplicativo

redirect_uri

O URL para o qual o servidor de autenticação redireciona o navegador depois que o Amazon Cognito autoriza o usuário.

identity_provider

O nome do provedor de identidades para o qual você deseja redirecionar o usuário.

idp_identifier

O identificador do provedor de identidades para o qual você deseja redirecionar o usuário.

scope

Os escopos OAuth 2.0 que você deseja que seu usuário solicite do servidor de autorização.

Exemplos de solicitações com respostas positivas

Exemplo - solicitação POST

A solicitação a seguir é para uma concessão de código de autorização para um usuário do IdP `MySAMLIdP` no cliente de aplicação `1example23456789`. O usuário redireciona para `https://www.example.com` com seu código de autorização, que pode ser trocado por tokens que incluem um token de acesso com os escopos OAuth `openid` 2.0, e. `email` phone

```
POST /saml2/idpresponse HTTP/1.1
User-Agent: USER_AGENT
Accept: */*
Host: example.auth.us-east-1.amazoncognito.com
```

```
Content-Type: application/x-www-form-urlencoded

SAMLResponse=[Base64-encoded SAML assertion]&RelayState=identity_provider
%3DMySAMLIdP%26client_id%3D1example23456789%26redirect_uri%3Dhttps%3A%2F
%2Fwww.example.com%26response_type%3Dcode%26scope%3Demail%2Bopenid%2Bphone
```

Exemplo - resposta

Veja a seguir um exemplo de resposta para a solicitação anterior.

```
HTTP/1.1 302 Found
Date: Wed, 06 Dec 2023 00:15:29 GMT
Content-Length: 0
x-amz-cognito-request-id: 8aba6eb5-fb54-4bc6-9368-c3878434f0fb
Location: https://www.example.com?code=[Authorization code]
```

OAuth 2.0 subsídios

O servidor de autorização OAuth 2.0 do grupo de usuários do Amazon Cognito emite tokens em resposta a três tipos de concessões de autorização OAuth 2.0. Você pode definir os tipos de concessão compatíveis para cada cliente da aplicação no grupo de usuários. Não é possível habilitar concessões de credenciais do cliente no mesmo cliente de aplicação como concessões implícitas ou de código de autorização. As solicitações de concessões implícitas e de código de autorização começam em Autorizar endpoint, ao passo que as solicitações de concessões de credenciais de clientes começam em Endpoint de token.

Concessão de código de autorização

Em resposta a uma solicitação de autenticação bem-sucedida, o servidor de autorização anexa um código de autorização em um parâmetro `code` ao URL de retorno de chamada. Depois é necessário trocar o código para ID, acesso e tokens de atualização com o Endpoint de token. Para solicitar uma concessão de código de autorização, defina `response_type` como `code` na solicitação. Para ver um exemplo de solicitação, consulte Exemplo: concessão de código de autorização. O Amazon Cognito é compatível com a chave de prova para troca de código (PKCE) em concessões de códigos de autorização.

A concessão de código de autorização é a forma mais segura de concessão de autorização. Ela não mostra o conteúdo do token diretamente aos usuários. Em vez disso, a aplicação é responsável por recuperar e armazenar com segurança os tokens do usuário. No Amazon Cognito, a concessão de código de autorização é a única maneira de obter todos os três

tipos de token (ID, acesso e atualização) do servidor de autorização. Você também pode obter todos os três tipos de token da autenticação por meio da API de grupos de usuários do Amazon Cognito, mas a API não emite tokens de acesso com escopos diferentes de `aws.cognito.signin.user.admin`.

Concessão implícita

Em resposta a uma solicitação de autenticação bem-sucedida, o servidor de autorização anexa um token de acesso em um parâmetro `access_token` e um token de ID em um parâmetro `id_token` ao URL de retorno de chamada. Uma concessão implícita não requer nenhuma interação adicional com o Endpoint de token. Para solicitar uma concessão implícita, defina `response_type` como `token` na solicitação. A concessão implícita gera apenas um ID e um token de acesso. Para ver um exemplo de solicitação, consulte Exemplo: concessão de token (implícita) sem escopo `openid`.

A concessão implícita é uma concessão de autorização herdada. Diferentemente da concessão do código de autorização, os usuários podem interceptar e inspecionar seus tokens. Para evitar a entrega de tokens por meio de concessão implícita, configure o cliente da aplicação para aceitar somente a concessão de código de autorização.

Credenciais do cliente

As credenciais do cliente são uma concessão de acesso somente para autorização. machine-to-machine Para receber uma concessão de credenciais do cliente, ignore o Autorizar endpoint e gere uma solicitação diretamente para o Endpoint de token. O cliente da aplicação deve ter um segredo e aceitar apenas concessões de credenciais de cliente. Em resposta a uma solicitação bem-sucedida, o servidor de autorização retorna um token de acesso.

O token de acesso de uma concessão de credenciais do cliente é um mecanismo de autorização que contém OAuth escopos 2.0. Normalmente, o token contém declarações de escopo personalizado que autorizam operações HTTP a serem protegidas por acesso APIs. Para obter mais informações, consulte Escopos, M2M e servidores de recursos.

As concessões de credenciais do cliente adicionam custos à sua AWS fatura. Para mais informações, consulte Preços do Amazon Cognito.

Token de atualização

Você pode solicitar uma concessão de token de atualização diretamente do Endpoint de token. Essa concessão retorna novos tokens de ID e acesso em troca de um token de atualização válido.

Para obter mais perspectivas sobre essas concessões e sua implementação, consulte Como usar a OAuth versão 2.0 no Amazon Cognito: saiba mais sobre as diferentes concessões da OAuth versão 2.0 no blog de AWS segurança.

Como usar PKCE em concessões de código de autorização

O Amazon Cognito oferece suporte à autenticação PKCE (Proof Key for Code Exchange, chave de prova para troca de código) em concessões de códigos de autorização. O PKCE é uma extensão da concessão do código de autorização do OAuth 2.0 para clientes públicos. O PKCE oferece proteção contra o resgate de códigos de autorização interceptados.

Como o Amazon Cognito usa o PKCE

Para iniciar a autenticação com o PKCE, sua aplicação deve gerar um valor de string exclusivo. Essa string é o verificador de código, um valor secreto que o Amazon Cognito usa para comparar o cliente que está solicitando a concessão de autorização inicial com o cliente que está trocando o código de autorização por tokens.

Seu aplicativo deve aplicar um SHA256 hash à string do verificador de código e codificar o resultado em base64. Passe a string com hash para o Autorizar endpoint como um parâmetro `code_challenge` no corpo da solicitação. Quando a aplicação troca o código de autorização por tokens, deve incluir a string do verificador de código em texto simples como um parâmetro `code_verifier` no corpo da solicitação para o Endpoint de token. O Amazon Cognito executa a mesma hash-and-encode operação no verificador de código. O Amazon Cognito só retornará tokens de ID, acesso e atualização se determinar que o verificador de código resulta no mesmo desafio de código que recebeu na solicitação de autorização.

Para implementar o fluxo de concessão de autorização com o PKCE

1. Abra o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários. Se você criar um grupo de usuários, receberá uma solicitação para configurar um cliente de aplicação e o login gerenciado durante o assistente.

    a. Se você criar um novo grupo de usuários, configure um cliente de aplicação e o login gerenciado durante a configuração guiada.

    b. Se você configurar um grupo de usuários existente, adicione um domínio e um cliente de aplicação público, caso ainda não tenha feito isso.

4. Gere uma sequência alfanumérica aleatória, normalmente um identificador exclusivo universal (UUID), para criar um desafio de código para o PKCE. Essa string é o valor do parâmetro `code_verifier` que você enviará em sua solicitação para o Endpoint de token.

5. Faça o hash da `code_verifier` string com o SHA256 algoritmo. Codifique o resultado da operação de hashing para base64. Essa string é o valor do parâmetro `code_challenge` que você enviará em sua solicitação para o Autorizar endpoint.

   O exemplo de Python a seguir gera um `code_verifier` e calcula o `code_challenge`:

   ```
   #!/usr/bin/env python3

   import secrets
   from base64 import urlsafe_b64encode
   from hashlib import sha256
   from string import ascii_letters
   from string import digits

   # use the secrets module for cryptographically strong random values
   alphabet = ascii_letters + digits
   code_verifier = ''.join(secrets.choice(alphabet) for _ in range(128))
   code_verifier_hash = sha256(code_verifier.encode()).digest()
   code_challenge = urlsafe_b64encode(code_verifier_hash).decode().rstrip('=')

   print(f"code challenge: {code_challenge}")
   print(f"code verifier: {code_verifier}")
   ```

   Veja um exemplo de saída do script Python:

   ```
   code challenge: Eh0mg-OZv7BAyo-tdv_vYamx1boOYDulDklyXoMDtLg
   code verifier: 9D-aW_iygXrgQcWJd0y0tNVMPSXSChIc2xceDhvYVdGLCBk-
   JWFTmBNjvKSdOrjTTYazOFbUmrFERrjWx6oKtK2b6z_x4_gHBDlr4K1mRFGyE8yA-05-_v7Dxf3EIYJH
   ```

6. Conclua o login no login gerenciado com uma solicitação de concessão de código de autorização com PKCE. O seguinte é um exemplo de URL:

   ```
   https://mydomain.auth.us-east-1.amazoncognito.com/oauth2/authorize?
   response_type=code&client_id=1example23456789&redirect_uri=https://
   www.example.com&code_challenge=Eh0mg-OZv7BAyo-
   tdv_vYamx1boOYDulDklyXoMDtLg&code_challenge_method=S256
   ```

7. Colete a autorização `code` e troque-a por tokens com o endpoint de token. Veja a seguir uma solicitação de exemplo:

```
POST /oauth2/token HTTP/1.1
Host: mydomain.auth.us-east-1.amazoncognito.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 296

redirect_uri=https%3A%2F%2Fwww.example.com&
client_id=1example23456789&
code=7378f445-c87f-400c-855e-0297d072ff03&
grant_type=authorization_code&
code_verifier=9D-aW_iygXrgQcWJd0y0tNVMPSXSChIc2xceDhvYVdGLCBk-
JWFTmBNjvKSdOrjTTYazOFbUmrFERrjWx6oKtK2b6z_x4_gHBDlr4K1mRFGyE8yA-05-_v7Dxf3EIYJH
```

8. Revise a resposta. Ela conterá tokens de ID, acesso e atualização. Para obter mais informações sobre como usar tokens de grupos de usuários do Amazon Cognito, consulte Compreendendo os tokens web JSON do grupo de usuários () JWTs.

Respostas de erro de federação e login gerenciado

Um processo de login no login gerenciado ou no login federado pode retornar um erro. Veja a seguir algumas condições que podem fazer a autenticação terminar com um erro.

- Um usuário realiza uma operação que o grupo de usuários não pode realizar.
- Um acionador do Lambda não responde com a sintaxe esperada.
- O provedor de identidades (IdP) retorna um erro.
- O Amazon Cognito não conseguiu validar as informações de atributos fornecidas pelo usuário.
- O IdP não enviou declarações que correspondem aos atributos necessários.

Quando o Amazon Cognito encontra um erro, ele o comunica de uma das formas a seguir.

1. O Amazon Cognito envia um URL de redirecionamento com o erro nos parâmetros da solicitação.
2. O Amazon Cognito exibe um erro no login gerenciado.

Os erros que o Amazon Cognito acrescenta aos parâmetros de solicitação têm o formato a seguir.

```
https://<Callback URL>/?error_description=error+description&error=error+name
```

Ao ajudar os usuários a enviar informações de erro quando eles não conseguem realizar uma operação, solicite que eles capturem o URL e o texto ou façam uma captura da página.

> **Note**
>
> As descrições de erro do Amazon Cognito não são strings fixas, e você não deve usar uma lógica que dependa de um padrão ou formato fixo.

Mensagens de erro do OIDC e do provedor de identidades social

O provedor de identidades retorna um erro. Quando um OIDC ou OAuth IdP 2.0 retorna um erro que está em conformidade com os padrões, o Amazon Cognito redireciona seu usuário para a URL de retorno de chamada e adiciona a resposta de erro do provedor aos parâmetros da solicitação de erro. O Amazon Cognito adiciona o nome do provedor e o código de erro HTTP às strings de erro existentes.

O URL a seguir é um exemplo de redirecionamento de um IdP que retornou um erro para o Amazon Cognito.

```
https://www.amazon.com/?error_description=LoginWithAmazon+Error+-+400+invalid_request
+The+request+is+missing+a+required+parameter+%3A+client_secret&error=invalid_request
```

Como o Amazon Cognito só retorna o que recebe de um provedor, o usuário pode ver um subconjunto dessas informações.

Quando o usuário encontra um problema com o login inicial por meio de seu IdP, o IdP envia qualquer mensagem de erro diretamente ao usuário. O Amazon Cognito retransmite uma mensagem de erro ao usuário quando gera uma solicitação ao seu IdP para validar a sessão do usuário. O Amazon Cognito retransmite e mensagens de erro do OAuth OIDC IdP dos seguintes endpoints.

`/token`

O Amazon Cognito troca o código de autorização do IdP por um token de acesso.

`/.well-known/openid-configuration`

O Amazon Cognito descobre o caminho para os endpoints do emissor.

`/.well-known/jwks.json`

Para verificar os JSON Web Tokens (JWTs) do seu usuário, o Amazon Cognito descobre as JSON Web Keys (JWKs) que seu IdP usa para assinar tokens.

Como o Amazon Cognito não inicia sessões de saída para provedores de SAML 2.0 que possam retornar erros de HTTP, os erros dos usuários durante uma sessão com um IdP SAML 2.0 não incluem essa forma de mensagem de erro do provedor.

Banco de identidades do Amazon Cognito

Um banco de identidades do Amazon Cognito é um diretório de identidades federadas que você pode trocar por credenciais da AWS . Os grupos de identidades geram AWS credenciais temporárias para os usuários do seu aplicativo, independentemente de eles terem feito login ou se você ainda não os tiver identificado. Com as funções e políticas AWS Identity and Access Management (IAM), você pode escolher o nível de permissão que deseja conceder aos seus usuários. Os usuários podem começar como convidados e recuperar os ativos mantidos em Serviços da AWS. Depois, podem fazer login com um provedor de identidades de terceiros para desbloquear o acesso aos ativos disponibilizados aos membros registrados. O provedor de identidade terceirizado pode ser um provedor de consumo (social) OAuth 2.0, como Apple ou Google, um provedor de identidade SAML ou OIDC personalizado, ou um esquema de autenticação personalizado, também chamado de provedor de desenvolvedor, criado por você mesmo.

Recursos de bancos de identidades do Amazon Cognito

Assine solicitações para Serviços da AWS

> Assine solicitações de API Serviços da AWS como Amazon Simple Storage Service (Amazon S3) e Amazon DynamoDB. Analise a atividade do usuário com serviços como Amazon Pinpoint e Amazon. CloudWatch

Filtrar solicitações com políticas baseadas em recurso

> Exerça um controle detalhado sobre o acesso dos usuários aos seus recursos. Transforme declarações de usuários em tags de sessão do IAM e crie políticas do IAM que concedam acesso de recursos a subconjuntos distintos de usuários.

Atribuir acesso de convidado

> Para os usuários que ainda não fizeram login, configure o banco de identidades para gerar credenciais da AWS com um escopo de acesso restrito. Autentique usuários por meio de um provedor de autenticação única para aumentar o acesso deles.

Atribuir perfis do IAM com base nas características do usuário

> Atribua um único perfil do IAM a todos os usuários autenticados ou selecione o perfil com base nas declarações de cada um.

Aceitar uma variedade de provedores de identidades

Troque um ID ou token de acesso, um token de grupo de usuários, uma declaração SAML ou um token de provedor social por credenciais OAuth . AWS

Validar suas próprias identidades

Faça sua própria validação de usuário e use suas AWS credenciais de desenvolvedor para emitir credenciais para seus usuários.

Talvez você já tenha um grupo de usuários do Amazon Cognito que forneça serviços de autenticação e autorização para sua aplicação. Você pode configurar o grupo de usuários como um provedor de identidades (IdP) para o banco de identidades. Ao fazer isso, seus usuários podem se autenticar por meio de seu grupo de usuários IdPs, consolidar suas reivindicações em um token de identidade OIDC comum e trocar esse token por credenciais. AWS O usuário pode então apresentar as credenciais dele em uma solicitação assinada para os Serviços da AWS.

Você também pode apresentar declarações autenticadas de qualquer um dos provedores de identidades diretamente em seu banco de identidades. O Amazon Cognito personaliza declarações de usuários de provedores de SAML e OIDC em uma AssumeRoleWithWebIdentitysolicitação de API para credenciais de curto prazo. OAuth

Os grupos de usuários do Amazon Cognito são como provedores de identidades OIDC para suas aplicações habilitadas para SSO. Os bancos de identidades funcionam como um provedor de identidades da AWS para qualquer aplicação com dependências de recursos que funcionam melhor com a autorização do IAM.

Os grupos de identidades do Amazon Cognito oferecem suporte aos seguintes provedores de identidade:

- Provedores públicos: Configurar o Login with Amazon como um IdP de bancos de identidades, Configurar o Facebook como um IdP de bancos de identidades, Configurar o Google como um IdP do banco de identidades, Configurar o Login com a Apple como um IdP do banco de identidades, Twitter.
- Grupos de usuários do Amazon Cognito
- Configurar um provedor OIDC como um IdP do banco de identidades
- Configurar um provedor SAML como um IdP do banco de identidades
- Identidades autenticadas pelo desenvolvedor

Para obter informações sobre a disponibilidade de regiões dos grupos de identidades do Amazon Cognito, consulte Disponibilidade de regiões de serviço da AWS.

Para obter mais informações sobre os grupos de identidades do Amazon Cognito, consulte os tópicos a seguir.

Tópicos

- Visão geral do console de bancos de identidades
- Fluxo de autenticação dos bancos de identidades
- Perfis do IAM
- Práticas recomendadas de segurança para bancos de identidades do Amazon Cognito
- Usar atributos para controle de acesso
- Controle de acesso com base em perfil
- Como obter credenciais
- Acessando Serviços da AWS com credenciais temporárias
- Bancos de identidades: provedores de identidade de terceiros
- Identidades autenticadas pelo desenvolvedor
- Alternar usuários não autenticados para usuários autenticados

Visão geral do console de bancos de identidades

Os grupos de identidade do Amazon Cognito fornecem AWS credenciais temporárias para usuários convidados (não autenticados) e para usuários que foram autenticados e receberam um token. O banco de identidades é um repositório de identificadores de usuário vinculados aos seus provedores de identidade externos.

Uma forma de entender os recursos e as opções dos bancos de identidades é criar um no console do Amazon Cognito. Você pode explorar o efeito de diferentes configurações nos fluxos de autenticação, no controle de acesso baseado em funções e atributos e no acesso de convidados. A partir daí, você pode avançar até os capítulos posteriores deste guia e adicionar os componentes apropriados à sua aplicação para poder implementar a autenticação do banco de identidades.

Tópicos

- Criar um banco de identidades do

- Funções do IAM do usuário
- Identidades autenticadas e não autenticadas
- Ativar ou desativar o acesso de convidados
- Alteração da função associada a um tipo de identidade
- Editar provedores de identidades
- Excluir um grupo de identidades
- Excluir uma identidade de um grupo de identidades
- Usar o Amazon Cognito Sync com grupos de identidades

Criar um banco de identidades do

Para criar um novo grupo de identidades no console

1. Faça login no console do Amazon Cognito e selecione Bancos de identidades.
2. Selecione Criar banco de identidades.
3. Em Configurar confiança do banco de identidades, opte por configurar seu banco de identidades para Acesso autenticado, Acesso de convidado ou ambos.

    - Se você selecionou Acesso autenticado, escolha um ou mais Tipos de identidade que você deseja definir como origem de identidades autenticadas no banco de identidades. Se você configurar um Provedor de desenvolvedor personalizado, não poderá modificá-lo nem o excluir depois de criar o banco de identidades.
4. Em Configurar permissões, selecione um perfil padrão do IAM para usuários autenticados ou convidados em seu banco de identidades.

    a. Selecione Criar um perfil do IAM se quiser que o Amazon Cognito crie um perfil para você com permissões básicas e uma relação de confiança com seu banco de identidades. Insira um Nome de perfil do IAM para identificar seu novo perfil; por exemplo, `myidentitypool_authenticatedrole`. Selecione Visualizar documento de política para examinar as permissões que o Amazon Cognito atribuirá ao novo perfil do IAM.

    b. Você pode optar por usar uma função do IAM existente se já tiver uma função na sua Conta da AWS que queira usar. Você deve configurar sua política de confiança de perfis do IAM para incluir `cognito-identity.amazonaws.com`. Configure sua política de confiança de perfil para permitir que o Amazon Cognito assuma o perfil somente quando apresentar evidências de que a solicitação se originou de um usuário autenticado em seu banco de

identidades específico. Para obter mais informações, consulte Permissões e confiança de função.

5. Em Connect identity providers, insira os detalhes dos provedores de identidade (IdPs) que você escolheu em Configurar a confiança do grupo de identidades. Você pode ser solicitado a fornecer informações do cliente do OAuth aplicativo, escolher um grupo de usuários do Amazon Cognito, escolher um IdP do IAM ou inserir um identificador personalizado para um provedor de desenvolvedores.

   a. Selecione Configurações de perfil para cada IdP. Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras. Com um IdP de grupo de usuários do Amazon Cognito, você também pode Escolher perfil com preferred_role em tokens. Para ter mais informações sobre a declaração `cognito:preferred_role`, consulte Como atribuir valores de precedência a grupos.

      i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

      ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.

   b. Configure Atributos para controle de acesso para cada IdP. Os atributos para controle de acesso correlacionam as declarações do usuário com as tags de entidade principal que o Amazon Cognito aplica à sua sessão temporária. Você pode criar políticas do IAM para filtrar o acesso do usuário com base nas tags aplicadas à sessão.

      i. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

      ii. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

      iii. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.

6. Em Configurar propriedades, insira um Nome em Nome do banco de identidades.

7. Em Autenticação básica (clássica), escolha se você deseja Ativar fluxo básico. Com o fluxo básico ativo, você pode ignorar as seleções de função que você fez para você IdPs e ligar diretamente. AssumeRoleWithWebIdentity Para obter mais informações, consulte Fluxo de autenticação dos bancos de identidades.
8. Em Tags, selecione Adicionar tag se quiser aplicar tags ao banco de identidades.
9. Em Revisar e criar, confirme as seleções que você fez para o novo banco de identidades. Selecione Editar para retornar ao assistente e alterar as configurações. Quando terminar, selecione Criar banco de identidades.

Funções do IAM do usuário

Uma função do IAM define as permissões para seus usuários acessarem AWS recursos, por exemploAmazon Cognito Sync. Os usuários do aplicativo assumirão as funções que você criar. Você pode especificar funções diferentes para usuários autenticados e não autenticados. Para saber mais sobre as funções do IAM, consulte Perfis do IAM.

Identidades autenticadas e não autenticadas

Os grupos de identidades do Amazon Cognito oferecem suporte a identidades autenticadas e não autenticadas. As identidades autenticadas pertencem a usuários que são autenticados por qualquer provedor de identidades. As identidades não autenticadas normalmente pertencem a usuários convidados.

- Para configurar as identidades autenticadas com um provedor de login público, consulte Bancos de identidades: provedores de identidade de terceiros.
- Para configurar seu próprio processo de autenticação de backend, consulte Identidades autenticadas pelo desenvolvedor.

Ativar ou desativar o acesso de convidados

O acesso de convidados aos grupos de identidade do Amazon Cognito (identidades não autenticadas) fornece um identificador e AWS credenciais exclusivos para usuários que não se autenticam com um provedor de identidade. Se a aplicação permitir usuários que não fazem login, você poderá ativar o acesso para identidades não autenticadas. Para saber mais, consulte Conceitos básicos dos bancos de identidades do Amazon Cognito.

Como atualizar o acesso de convidados em um banco de identidades

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Localize Acesso de convidado. Em um banco de identidades não compatível no momento com o acesso de convidados, o Status é Inativo.

    a. Se Acesso de convidado estiver Ativo e você quiser desativá-lo, selecione Desativar.

    b. Se Acesso de convidado estiver Inativo e você quiser ativá-lo, selecione Editar.

    - Selecione um perfil padrão do IAM para usuários convidados em seu banco de identidades.

        A. Selecione Criar um perfil do IAM se quiser que o Amazon Cognito crie um perfil para você com permissões básicas e uma relação de confiança com seu banco de identidades. Insira um Nome de perfil do IAM para identificar seu novo perfil; por exemplo, `myidentitypool_authenticatedrole`. Selecione Visualizar documento de política para examinar as permissões que o Amazon Cognito atribuirá ao novo perfil do IAM.

        B. Você pode optar por usar uma função do IAM existente se já tiver uma função na sua Conta da AWS que queira usar. Você deve configurar sua política de confiança de perfis do IAM para incluir `cognito-identity.amazonaws.com`. Configure sua política de confiança de perfil para permitir que o Amazon Cognito assuma o perfil somente quando apresentar evidências de que a solicitação se originou de um usuário autenticado em seu banco de identidades específico. Para obter mais informações, consulte Permissões e confiança de função.

        C. Selecione Salvar alterações.

        D. Para ativar o acesso de convidados, selecione Ativar na guia Acesso do usuário.

Alteração da função associada a um tipo de identidade

Cada identidade em seu grupo de identidades é autenticada ou não autenticada. As identidades autenticadas pertencem aos usuários que são autenticados por um provedor de login público (grupos de usuários do Amazon Cognito, Login with Amazon, Fazer login com a Apple, Facebook, Google, SAML ou qualquer provedor do OpenID Connect) ou um provedor de desenvolvedor (seu próprio

processo de autenticação de backend). As identidades não autenticadas normalmente pertencem a usuários convidados.

Para cada tipo de identidade, há uma função atribuída. Essa função tem uma política anexada que determina qual função Serviços da AWS essa função pode acessar. Quando o Amazon Cognito receber uma solicitação, o serviço determina o tipo de identidade, a função atribuída a esse tipo de identidade e usa a política anexada a essa função para responder. Ao modificar uma política ou atribuir uma função diferente a um tipo de identidade, você pode controlar qual tipo Serviços da AWS de identidade pode acessar. Para exibir ou modificar as políticas associadas às funções no grupo de identidades, consulte o Console do AWS IAM.

Como alterar o perfil padrão autenticado ou não autenticado do banco de identidades

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Localize Acesso de convidado ou Acesso autenticado. Em um banco de identidades não configurado no momento para esse tipo de acesso, o Status é Inativo. Selecione Editar.
4. Selecione um perfil padrão do IAM para convidados ou usuários autenticados em seu banco de identidades.

    a. Selecione Criar um perfil do IAM se quiser que o Amazon Cognito crie um perfil para você com permissões básicas e uma relação de confiança com seu banco de identidades. Insira um Nome de perfil do IAM para identificar seu novo perfil; por exemplo, `myidentitypool_authenticatedrole`. Selecione Visualizar documento de política para examinar as permissões que o Amazon Cognito atribuirá ao novo perfil do IAM.

    b. Você pode optar por usar uma função do IAM existente se já tiver uma função na sua Conta da AWS que queira usar. Você deve configurar sua política de confiança de perfis do IAM para incluir `cognito-identity.amazonaws.com`. Configure sua política de confiança de perfil para permitir que o Amazon Cognito assuma o perfil somente quando apresentar evidências de que a solicitação se originou de um usuário autenticado em seu banco de identidades específico. Para obter mais informações, consulte Permissões e confiança de função.
5. Selecione Salvar alterações.

Editar provedores de identidades

Se você permitir que os usuários realizem a autenticação por meio de provedores de identidades públicos (por exemplo, grupos de usuários do Amazon Cognito, Login with Amazon, Login with Apple, Facebook ou Google), poderá especificar os identificadores da aplicação no console de bancos de identidades (identidades federadas) do Amazon Cognito. Isso associa o ID do aplicativo (fornecido pelo provedor de login público) ao seu grupo de identidades.

Você também pode configurar regras de autenticação para cada provedor desta página. Cada provedor permite até 25 regras. As regras são aplicadas na ordem salva para cada provedor. Para obter mais informações, consulte Controle de acesso com base em perfil.

> **Warning**
>
> A alteração do ID da aplicação do IdP vinculado em seu banco de identidades impede que os usuários existentes se autentiquem no banco de identidades em questão. Para obter mais informações, consulte Bancos de identidades: provedores de identidade de terceiros.

Como atualizar um provedor de identidades (IdP) de banco de identidades

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Localize Provedores de identidade. Selecione o provedor de identidades a ser editado. Se você quiser adicionar um novo IdP, selecione Adicionar provedor de identidade.
   - Se você escolheu Adicionar provedor de identidade, selecione um dos Tipos de identidade que você deseja adicionar.
4. Para alterar o ID da aplicação, selecione Editar em Informações do provedor de identidade.
5. Para alterar o perfil que o Amazon Cognito solicita ao emitir credenciais para usuários que se autenticaram com esse provedor, selecione Editar em Configurações do perfil.
   - Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras. Com um IdP de grupo de usuários do Amazon Cognito, você também pode Escolher perfil com preferred_role em tokens. Para ter mais informações sobre a declaração `cognito:preferred_role`, consulte Como atribuir valores de precedência a grupos.

i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.

6. Para alterar as tags de entidade principal que o Amazon Cognito atribui quando emite credenciais para usuários que se autenticaram com esse provedor, selecione Editar em Atributos para controle de acesso.

   a. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

   b. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

   c. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.

7. Selecione Salvar alterações.

Excluir um grupo de identidades

Não é possível desfazer a exclusão do banco de identidades. Após a exclusão de um banco de identidades, todas as aplicações e usuários que dependem dele param de funcionar.

Para excluir um grupo de identidades

1. Selecione Bancos de identidades no console do Amazon Cognito. Marque a caixa de opção ao lado do banco de identidades a ser excluído.
2. Selecione Excluir.
3. Insira ou cole o nome do banco de identidades e selecione Excluir.

> **Warning**
>
> Ao selecionar o botão Delete (Excluir), você excluirá permanentemente seu grupo de identidades e todos os dados de usuários nele contidos. A exclusão de um banco de identidades fará com que as aplicações e outros serviços que utilizam o banco parem de funcionar.

Excluir uma identidade de um grupo de identidades

Ao excluir uma identidade de um banco de identidades, você remove as informações de identificação que o Amazon Cognito armazenou para esse usuário federado. Quando o usuário solicitar credenciais novamente, ele receberá um novo ID de identidade se o banco de identidades ainda confiar em seu provedor de identidades. Você não pode desfazer esta operação.

Como excluir uma identidade

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Navegador de identidade.
3. Marque a caixa de seleção ao lado das identidades a serem excluídas e selecione Excluir. Confirme que você deseja excluir as identidades e selecione Excluir.

Usar o Amazon Cognito Sync com grupos de identidades

O Amazon Cognito Sync é uma AWS service (Serviço da AWS) biblioteca de clientes que possibilita a sincronização de dados de usuários relacionados a aplicativos em vários dispositivos. O Amazon Cognito pode sincronizar dados de perfil do usuário entre dispositivos móveis e a Web sem precisar usar seu próprio backend. As bibliotecas de cliente armazenam dados em cache localmente para que a aplicação possa ler e gravar dados, independentemente do status de conectividade do dispositivo. Quando o dispositivo estiver online, você poderá sincronizar dados. Se você configurar a sincronização por push, poderá notificar outros dispositivos imediatamente de que uma atualização está disponível.

Como gerenciar conjuntos de dados

Se você tiver implementado a funcionalidade do Amazon Cognito na sua aplicação, o console dos grupos de identidades do Amazon Cognito permitirá que você crie e exclua manualmente conjuntos

de dados e registros para identidades individuais. Qualquer alteração feita no conjunto de dados ou nos registros de uma identidade no console de grupos de identidades do Amazon Cognito só será salva depois que você selecionar Synchronize (Sincronizar) no console. A alteração não fica visível para o usuário final até que a identidade chame Synchronize (Sincronizar). Os dados que estão sendo sincronizados de outros dispositivos para identidades individuais ficam visíveis depois que você atualizar a página de conjuntos de dados de lista de uma identidade específica.

Criar um conjunto de dados para uma identidade

O Amazon Cognito Sync associa um conjunto de dados a uma identidade. Você pode preencher o conjunto de dados com informações de identificação sobre o usuário que a identidade representa e sincronizar essas informações com todos os dispositivos do usuário.

Como adicionar um conjunto de dados e registros de conjunto de dados a uma identidade

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Navegador de identidade.
3. Selecione a identidade a ser editada.
4. Em Conjuntos de dados, selecione Criar conjunto de dados.
5. Insira um Nome de conjunto de dados e selecione Criar conjunto de dados.
6. Se você quiser adicionar registros ao conjunto de dados, selecione o conjunto de dados nos detalhes de identidade. Em Registros, selecione Criar registro.
7. Insira uma Chave e um Valor para o registro. Escolha Confirmar. Repita para adicionar mais registros.

Excluir um conjunto de dados associado a uma identidade

Como excluir um conjunto de dados de uma identidade e os respectivos registros

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Navegador de identidade.
3. Selecione a identidade que contém o conjunto de dados a ser excluído.
4. Em Conjuntos de dados, selecione o botão de opção ao lado do conjunto de dados que você deseja excluir.

5. Selecione Excluir. Revise sua escolha e selecione Excluir novamente.

Publicação de dados em massa

A publicação em massa pode ser usada para exportar dados já armazenados no repositório do Amazon Cognito Sync para um fluxo do Amazon Kinesis. Para obter instruções sobre como publicar em massa todos os seus fluxos, consulte Como implementar o Amazon Cognito Sync Streams.

Ativar a sincronização por push

O Amazon Cognito rastreia automaticamente a associação entre identidade e dispositivos. Usando o recurso de sincronização por push, você pode garantir que cada instância de determinada identidade seja notificada quando os dados da identidade forem alterados. A sincronização por push faz isso de maneira que, sempre que o conjunto de dados de uma identidade for alterado, todos os dispositivos associados a essa identidade recebam uma notificação push silenciosa informando-os da alteração.

Você pode ativar a sincronização por push no console do Amazon Cognito.

Como ativar a sincronização por push

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Propriedades do grupo de identidades.
3. Em Sincronização por push, selecione Editar.
4. Selecione Ativar sincronização por push com seu banco de identidades.
5. Selecione uma das Aplicações de plataforma do Amazon Simple Notification Service (Amazon SNS) que você criou na Região da AWS atual. O Amazon Cognito publica notificações push em sua aplicação de plataforma. Selecione Criar aplicação de plataforma para navegar até o console do Amazon SNS e crie outra.
6. Para publicar em sua aplicação de plataforma, o Amazon Cognito assume um perfil do IAM em sua Conta da AWS. Selecione Criar um perfil do IAM se quiser que o Amazon Cognito crie um perfil para você com permissões básicas e uma relação de confiança com seu banco de identidades. Insira um Nome de perfil do IAM para identificar seu novo perfil; por exemplo, `myidentitypool_authenticatedrole`. Selecione Visualizar documento de política para examinar as permissões que o Amazon Cognito atribuirá ao novo perfil do IAM.
7. Você pode optar por usar uma função do IAM existente se já tiver uma função na sua Conta da AWS que queira usar. Você deve configurar sua política de confiança de perfis do IAM para

incluir `cognito-identity.amazonaws.com`. Configure sua política de confiança de perfil para permitir que o Amazon Cognito assuma o perfil somente quando apresentar evidências de que a solicitação se originou de um usuário autenticado em seu banco de identidades específico. Para obter mais informações, consulte Permissões e confiança de função.

8. Selecione Salvar alterações.

Configurar o Amazon Cognito Streams

O Amazon Cognito Streams oferece aos desenvolvedores controle e insight sobre os dados armazenados no Amazon Cognito Sync. Agora, os desenvolvedores podem configurar um fluxo do Kinesis para receber eventos como dados. O Amazon Cognito pode enviar cada alteração de conjunto de dados a um fluxo do Kinesis de sua propriedade em tempo real. Para obter instruções sobre como configurar o Amazon Cognito Streams no console do Amazon Cognito, consulte Como implementar o Amazon Cognito Sync Streams.

Configurar o Amazon Cognito Events

O Amazon Cognito Events permite que você execute uma AWS Lambda função em resposta a eventos importantes no Amazon Cognito Sync. O Amazon Cognito Sync gera o evento Sync Trigger quando um conjunto de dados é sincronizado. Você pode usar o evento Sync Trigger para executar uma ação quando um usuário atualizar dados. Para obter instruções sobre como configurar eventos do Amazon Cognito no console, consulte Como personalizar fluxos de trabalho com o Amazon Cognito Events.

Para saber mais sobre AWS Lambda, consulte AWS Lambda.

Fluxo de autenticação dos bancos de identidades

O Amazon Cognito ajuda você a criar identificadores exclusivos para seus usuários finais que são mantidos consistentes em diversos dispositivos e plataformas. O Amazon Cognito também fornece credenciais temporárias com privilégios limitados ao seu aplicativo para acessar recursos. AWS Esta página aborda as noções básicas de como funciona a autenticação no Amazon Cognito e explica o ciclo de vida de uma identidade no grupo de identidades.

Fluxo de autenticação de provedor externo

Uma autenticação de usuário com o Amazon Cognito passará por um processo de várias etapas para fazer bootstrap das respectivas credenciais. O Amazon Cognito tem dois fluxos diferentes para autenticação com provedores públicos: aprimorado e básico.

Depois de concluir um desses fluxos, você pode acessar outros Serviços da AWS conforme definido pelas políticas de acesso da sua função. Por padrão, o console do Amazon Cognito cria funções com acesso ao armazenamento do Amazon Cognito Sync e ao Amazon Mobile Analytics. Para obter mais informações sobre como conceder acesso adicional, consulte Perfis do IAM.

Os bancos de identidades aceitam os seguintes artefatos dos provedores:

Fornecedor	Artefato de autenticação
Conjunto de usuários do Amazon Cognito	Token de ID
OpenID Connect (OIDC)	Token de ID
SAML 2.0	Declaração SAML
Provedor social	Token de acesso

O fluxo de autenticação aprimorado (simplificado)

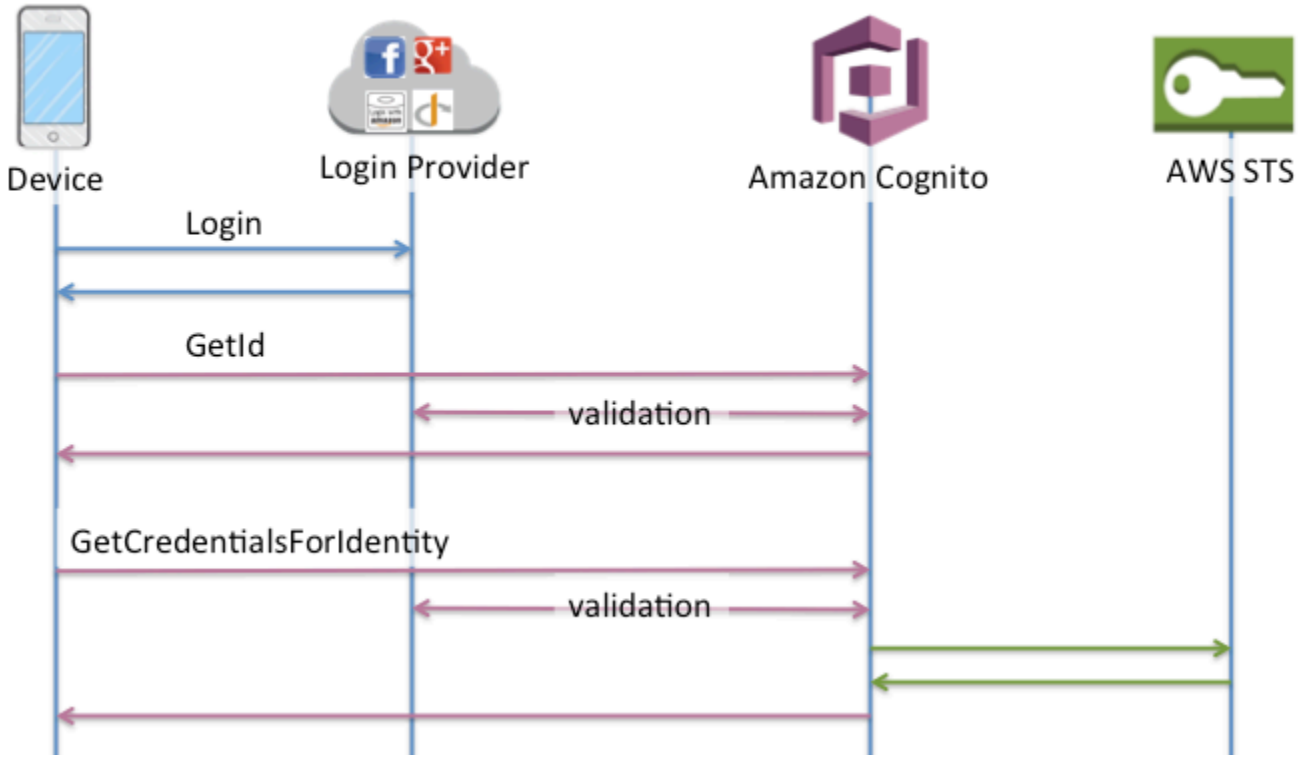
Quando você usa o fluxo de autenticação aprimorado, seu aplicativo primeiro apresenta uma prova de autenticação de um grupo de usuários autorizado do Amazon Cognito ou de um provedor de identidade terceirizado em GetIduma solicitação.

1. Sua aplicação apresenta uma prova de autenticação: um token JSON da web ou uma declaração SAML de um grupo de usuários autorizado do Amazon Cognito ou provedor de identidades de terceiros em uma solicitação GetID.
2. Seu banco de identidades retorna um ID de identidade.
3. Seu aplicativo combina o ID de identidade com o mesmo comprovante de autenticação em uma GetCredentialsForIdentitysolicitação.
4. Seu pool de identidade retorna AWS credenciais.
5. Seu aplicativo assina solicitações de AWS API com as credenciais temporárias.

A autenticação aprimorada gerencia a lógica da seleção de perfil do IAM e da recuperação de credenciais na configuração do seu banco de identidades. Você pode configurar o banco de identidades para selecionar uma função padrão, para aplicar os princípios de controle de acesso por atributo (ABAC) ou controle de acesso por função (RBAC) à seleção de função. As AWS credenciais da autenticação avançada são válidas por uma hora.

Ordem das operações na autenticação aprimorada

1. `GetId`
2. `GetCredentialsForIdentity`



O fluxo de autenticação básica (clássica)

Ao implementar o fluxo de autenticação básica, a aplicação seleciona o perfil do IAM que você deseja que os usuários assumam.

1. Sua aplicação apresenta uma prova de autenticação: um token JSON da web ou uma declaração SAML de um grupo de usuários autorizado do Amazon Cognito ou provedor de identidades de terceiros em uma solicitação GetID.
2. Seu banco de identidades retorna um ID de identidade.
3. Seu aplicativo combina o ID de identidade com o mesmo comprovante de autenticação em uma GetOpenIdTokensolicitação.
4. `GetOpenIdToken`retorna um novo token OAuth 2.0 emitido pelo seu grupo de identidades.
5. Seu aplicativo apresenta o novo token em uma AssumeRoleWithWebIdentitysolicitação.

6. AWS Security Token Service (AWS STS) retorna AWS as credenciais.
7. Seu aplicativo assina solicitações de AWS API com as credenciais temporárias.

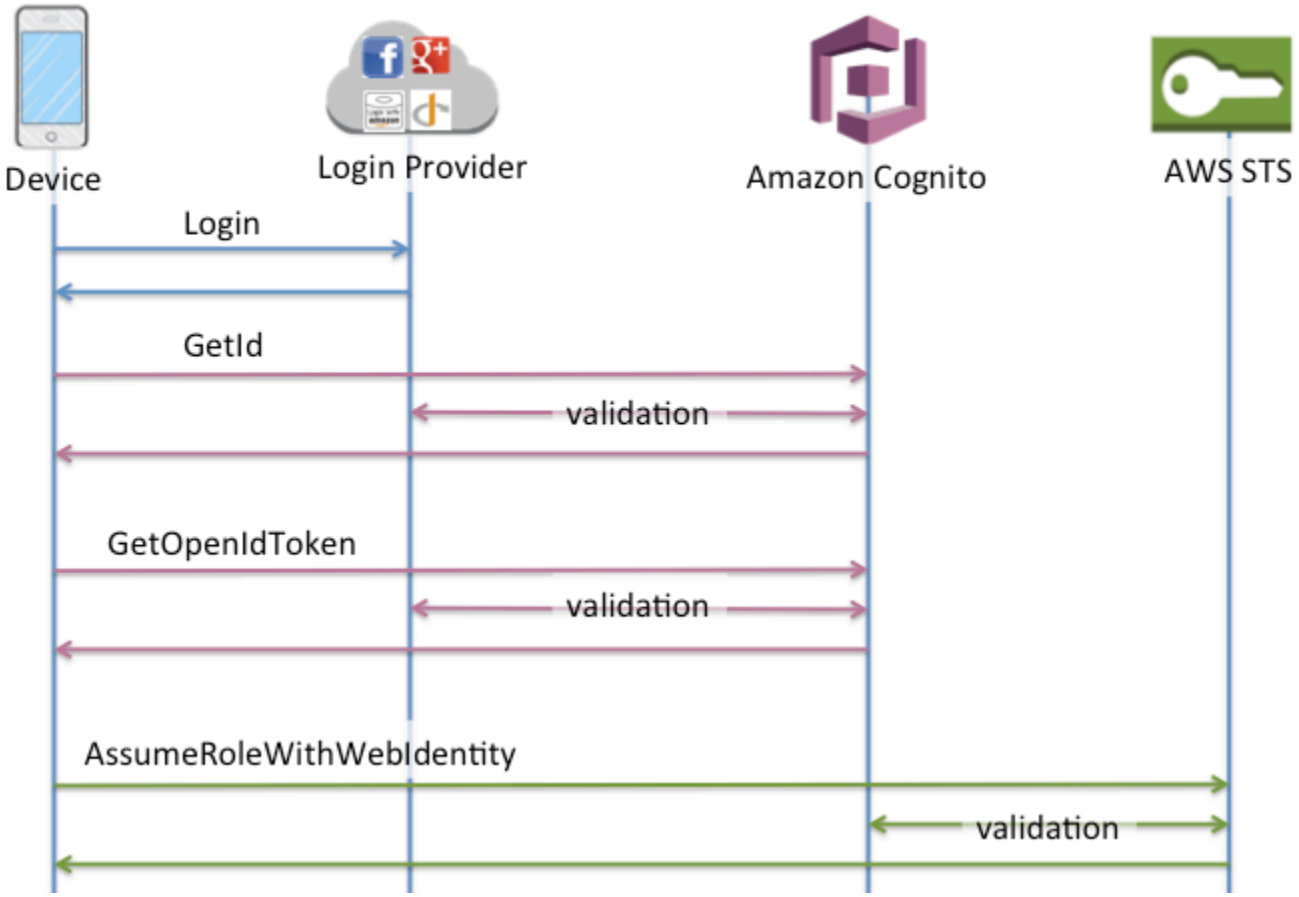
O fluxo de trabalho básico oferece um controle mais detalhado sobre as credenciais que você distribui aos seus usuários. A solicitação `GetCredentialsForIdentity` do fluxo de autenticação aprimorado solicita uma função com base no conteúdo de um token de acesso. A `AssumeRoleWithWebIdentity` solicitação no fluxo de trabalho clássico concede ao seu aplicativo uma maior capacidade de solicitar credenciais para qualquer AWS Identity and Access Management função que você tenha configurado com uma política de confiança suficiente. Você também pode solicitar uma duração de sessão de perfil personalizado.

Você pode fazer login com o fluxo de autenticação básico em grupos de usuários que não têm mapeamentos de perfil. Esse tipo de banco de identidades não tem um perfil padrão autenticado ou não autenticado e não tem controle de acesso por função ou atributo configurado. Ao tentar `GetOpenIdToken` em um banco de identidades com mapeamentos de perfil, você recebe o seguinte erro.

```
Basic (classic) flow is not supported with RoleMappings, please use enhanced flow.
```

Ordem das operações na autenticação básica

1. `GetId`
2. `GetOpenIdToken`
3. `AssumeRoleWithWebIdentity`

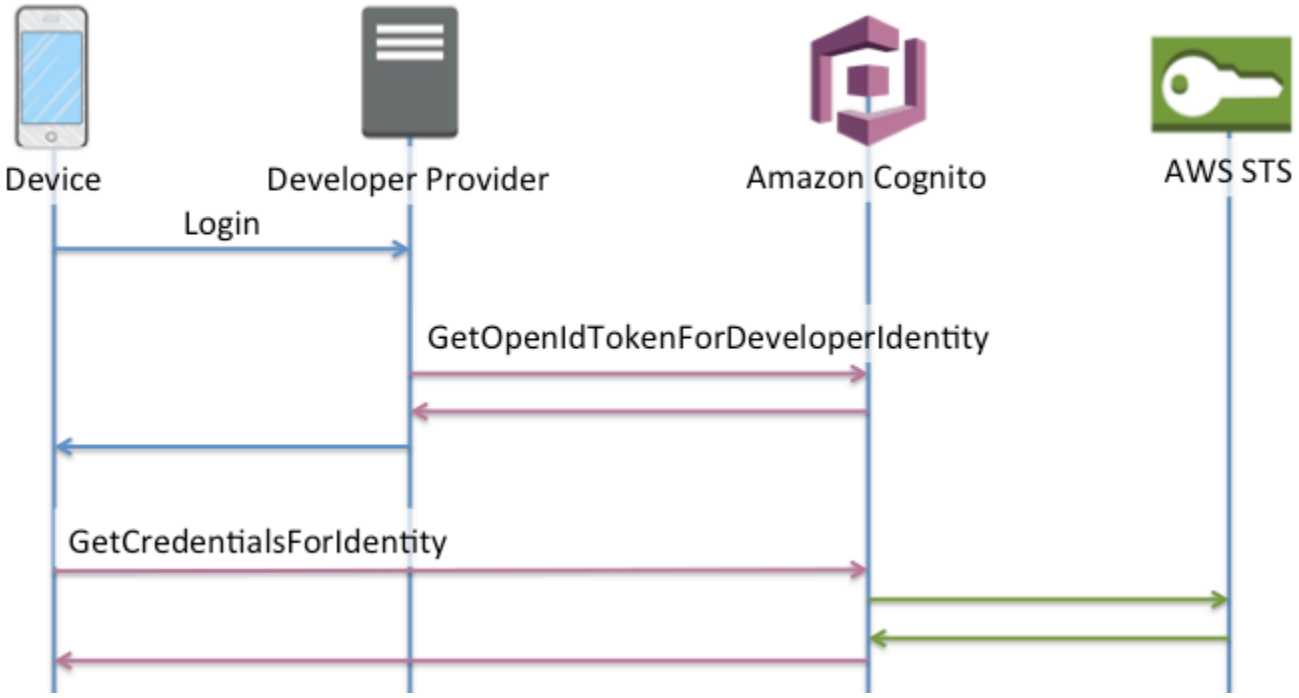


O fluxo de autenticação autenticado pelo desenvolvedor

Ao usar Identidades autenticadas pelo desenvolvedor, o cliente usa outro fluxo de autenticação que inclui o código fora do Amazon Cognito para validar o usuário em seu próprio sistema de autenticação. Do ponto de vista do banco de identidades, as declarações apresentadas na solicitação de identidade são identificadores arbitrários, e a autenticação é autorizada pelas credenciais do IAM codificadas em sua aplicação.

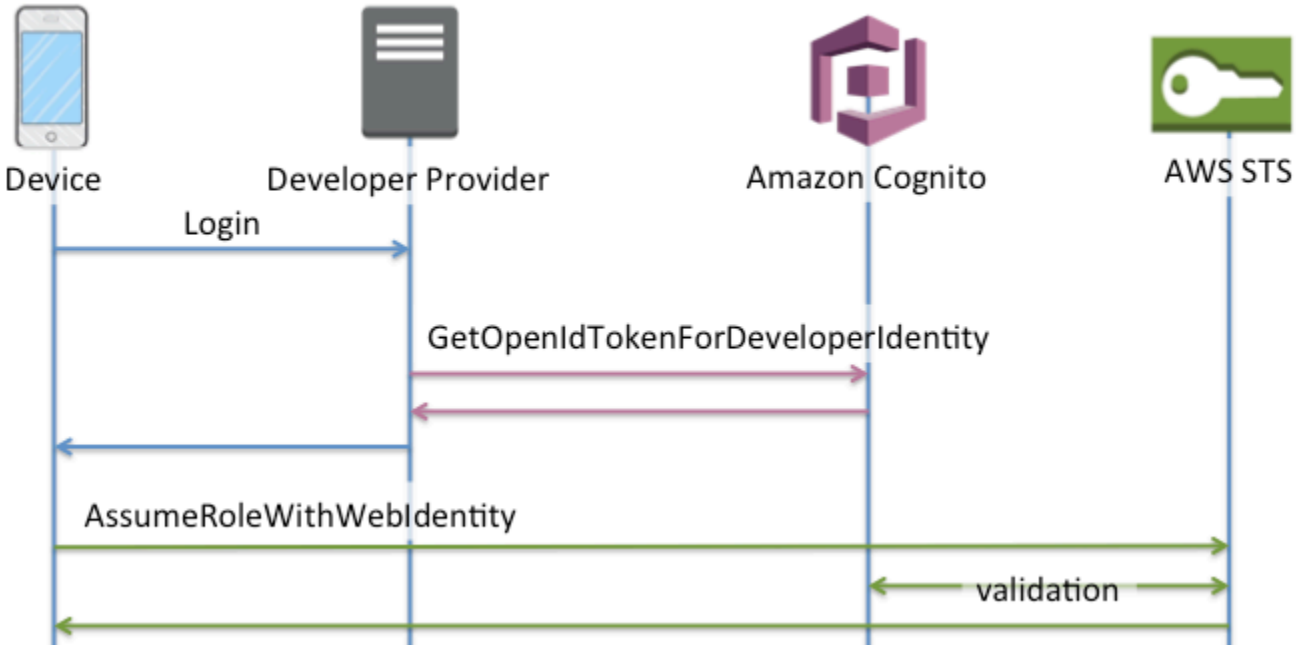
Ordem das operações na autenticação aprimorada com um provedor de desenvolvedores

1. Faça login por meio do provedor do desenvolvedor (código fora do Amazon Cognito)
2. Valide o login do usuário (código fora do Amazon Cognito).
3. GetOpenIdTokenForDeveloperIdentity
4. GetCredentialsForIdentity



Ordem das operações na autenticação básica com um provedor desenvolvedor

1. Implemente a lógica fora do banco de identidades para fazer login e gerar um identificador de provedor desenvolvedor.
2. Recupere as credenciais armazenadas do lado do servidor AWS .
3. Envie o identificador do provedor do desenvolvedor em uma solicitação de GetOpenIdTokenForDeveloperIdentityAPI assinada com AWS credenciais autorizadas.
4. Solicite as credenciais do aplicativo com AssumeRoleWithWebIdentity.



Qual fluxo de autenticação devo implementar?

O fluxo aprimorado é a opção mais segura com o menor nível de esforço do desenvolvedor:

- O fluxo aprimorado reduz a complexidade, o tamanho e a taxa das solicitações de API.
- Sua aplicação não precisa fazer solicitações adicionais de API para o AWS STS.
- Seu banco de identidades avalia as credenciais de perfil do IAM que os usuários devem receber. Você não precisa incorporar lógica para seleção de perfil no cliente.

> **Important**
>
> Ao criar um banco de identidades, não ative a autenticação básica (clássica) por padrão, como prática recomendada. Para implementar a autenticação básica, primeiro avalie as relações de confiança de seus perfis do IAM para identidades da web. Em seguida, incorpore a lógica para seleção de perfis no cliente, protegendo-o contra modificações dos usuários.

O fluxo de autenticação básica delega a lógica da seleção de perfil do IAM à sua aplicação. Nesse fluxo, o Amazon Cognito valida a sessão autenticada ou não autenticada do seu usuário e emite um token com o qual você pode trocar por credenciais. AWS STS Os usuários podem trocar os tokens

da autenticação básica por qualquer função do IAM que confie `amr` em seu grupo de identidade e/ou authenticated/unauthenticated estado.

Da mesma forma, entenda que a autenticação do desenvolvedor é um atalho para a validação da autenticação do provedor de identidades. O Amazon Cognito confia nas AWS credenciais que autorizam uma GetOpenIdTokenForDeveloperIdentitysolicitação sem validação adicional do conteúdo da solicitação. Proteja os segredos que autorizam a autenticação do desenvolvedor para que os usuários não tenham acesso.

Visão geral das operações de API de fluxo de autenticação

GetId

A chamada de API GetId é a primeira chamada necessária para estabelecer uma nova identidade no Amazon Cognito.

Acesso não autenticado

O Amazon Cognito pode conceder acesso de convidado não autenticado às aplicações. Se esse recurso for habilitado no grupo de identidades, os usuários poderão solicitar um novo ID de identidade a qualquer momento por meio da API `GetId`. Espera-se que a aplicação armazene em cache esse ID de identidade para fazer chamadas subsequentes para o Amazon Cognito. O AWS Mobile SDKs e o AWS SDK do Browser têm provedores de credenciais que gerenciam esse armazenamento em cache para você. JavaScript

Acesso autenticado

Quando você configura seu aplicativo com suporte para um provedor de login público (Facebook, Google+, Login with Amazon ou Sign in with Apple), os usuários também podem fornecer tokens (ou OAuth OpenID Connect) que os identificam nesses provedores. Quando for usado em uma chamada para `GetId`, o Amazon Cognito criará uma identidade autenticada ou retornará a identidade já associada a esse login específico. O Amazon Cognito faz isso validando o token com o provedor e garantindo que:

- O token seja válido e proveniente do provedor configurado.
- O token não tenha expirado.
- O token corresponde ao identificador de aplicação criado com esse provedor (por exemplo, ID do aplicativo do Facebook)
- O token corresponda ao identificador de usuário.

GetCredentialsForIdentity

A API GetCredentialsForIdentity pode ser chamada depois que você estabelecer um ID de identidade. Essa operação é funcionalmente equivalente a chamar GetOpenIdToken, então AssumeRoleWithWebIdentity.

Para que o Amazon Cognito chame `AssumeRoleWithWebIdentity` em seu nome, seu grupo de identidades deve ter funções do IAM associadas a ele. Você pode fazer isso por meio do console do Amazon Cognito ou, manualmente, pela operação SetIdentityPoolRoles.

GetOpenIdToken

Faça uma solicitação de API GetOpenIdToken depois de estabelecer um ID de identidade. Armazene a identidade em cache IDs após sua primeira solicitação e inicie as sessões básicas (clássicas) subsequentes dessa identidade com`GetOpenIdToken`.

A resposta a uma solicitação de API `GetOpenIdToken` é um token gerado pelo Amazon Cognito. Você pode enviar esse token como parâmetro `WebIdentityToken` em uma solicitação AssumeRoleWithWebIdentity.

Antes de enviar o token OpenID, verifique-o na aplicação. Você pode usar bibliotecas do OIDC em seu SDK ou em uma biblioteca, como aws-jwt-verify, para confirmar que o Amazon Cognito emitiu o token. O ID da chave de assinatura ou `kid` do token OpenID é um dos listados no documento jwks_uri do Amazon Cognito Identity †. Essas chaves estão sujeitas a alterações. Sua função que verifica os tokens do Amazon Cognito Identity deve atualizar periodicamente sua lista de chaves do documento jwks_uri. O Amazon Cognito define a duração da atualização no cabeçalho de resposta de controle de cache jwks_uri, atualmente definido como `max-age` de 30 dias.

Acesso não autenticado

Para obter um token para uma identidade não autenticada, você só precisa do próprio ID de identidade. Não é possível obter um token não autenticado para identidades autenticadas ou desativadas.

Acesso autenticado

Se você tem uma identidade autenticada, deve inserir ao menos um token válido para um login já associado a essa identidade. Todos os tokens inseridos durante a chamada de `GetOpenIdToken` devem passar na mesma validação mencionada anteriormente; se houver falha em um deles, toda a chamada falhará. A resposta da chamada de `GetOpenIdToken`

também inclui o ID de identidade. Isso ocorre porque o ID de identidade que você insere pode não ser o retornado.

Como vincular logins

Se você inserir um token referente a um login que ainda não está associado a nenhuma identidade, o login será considerado "vinculado" à identidade associada. Você só pode vincular um login por provedor de público. Tentativas de vincular mais de um login a um provedor público gerará uma resposta de erro `ResourceConflictException`. Se um login for meramente vinculado a uma identidade existente, o ID de identidade retornado por `GetOpenIdToken` será o mesmo que foi inserido.

Como mesclar identidades

Se você inserir um token referente a um login que não está atualmente vinculado à identidade fornecida, mas está vinculado a outra identidade, as duas identidades serão mescladas. Depois de mesclada, uma identidade se torna a parent/owner de todos os logins associados e a outra é desativada. Nesse caso, o ID de identidade do parent/owner é retornado. Você deverá atualizar seu cache local se esse valor for diferente. Os provedores no AWS celular SDKs ou JavaScript no AWS SDK do navegador realizam essa operação para você.

GetOpenIdTokenForDeveloperIdentity

A GetOpenIdTokenForDeveloperIdentityoperação substitui o uso GetOpenIdTokende GetIde para o dispositivo ao usar identidades autenticadas pelo desenvolvedor. Como a aplicação assina solicitações para essa operação de API com credenciais da AWS , o Amazon Cognito confia que o identificador de usuário fornecido na solicitação seja válido. A autenticação do desenvolvedor substitui a validação de token que o Amazon Cognito executa junto aos provedores externos.

A carga útil dessa API inclui um mapa de `logins`. Esse mapa deve conter a chave de seu provedor de desenvolvedor e um valor como identificador do usuário no sistema. Se o identificador de usuário ainda não estiver vinculado a uma identidade existente, o Amazon Cognito criará uma identidade e retornará o ID da nova identidade e um token do OpenID Connect para ela. Se o identificador de usuário já estiver vinculado, o Amazon Cognito retornará o ID de identidade preexistente e um token do OpenID Connect. Armazene em cache IDs a identidade do desenvolvedor após sua primeira solicitação e inicie as sessões básicas (clássicas) subsequentes dessa identidade com`GetOpenIdTokenForDeveloperIdentity`.

A resposta a uma solicitação de API `GetOpenIdTokenForDeveloperIdentity` é um token gerado pelo Amazon Cognito. Você pode enviar esse token como parâmetro `WebIdentityToken` em uma solicitação `AssumeRoleWithWebIdentity`.

Antes de enviar o token do OpenID Connect, verifique-o na aplicação. Você pode usar bibliotecas do OIDC em seu SDK ou em uma biblioteca, como aws-jwt-verify, para confirmar que o Amazon Cognito emitiu o token. O ID da chave de assinatura ou `kid` do token do OpenID Connect é um dos listados no documento jwks_uri do Amazon Cognito Identity†. Essas chaves estão sujeitas a alterações. Sua função que verifica os tokens do Amazon Cognito Identity deve atualizar periodicamente sua lista de chaves do documento jwks_uri. O Amazon Cognito define a duração da atualização no cabeçalho de resposta jwks_uri `cache-control`, atualmente definido como `max-age` de 30 dias.

Como vincular logins

Assim como ocorre com os provedores externos, o fornecimento de logins adicionais que ainda não estão associados a uma identidade os vinculará implicitamente a essa identidade. Se você vincular um login de provedor externo a uma identidade, o usuário poderá usar o fluxo de autenticação do provedor externo com esse provedor. No entanto, ele não pode usar o nome de seu provedor de desenvolvedor no mapa de logins ao chamar `GetId` ou `GetOpenIdToken`.

Como mesclar identidades

Quando as identidades são autenticadas pelo desenvolvedor, o Amazon Cognito comporta mesclagens implícitas e explícitas por meio da API MergeDeveloperIdentities. Com a mesclagem explícita, você pode marcar duas identidades com identificadores de usuário em seu sistema como uma única identidade. Se você fornecer os identificadores de usuário de origem e de destino, o Amazon Cognito os mesclará. Na próxima vez em que você solicitar um token do OpenID Connect para um dos identificadores de usuário, a mesma identidade será retornada.

AssumeRoleWithWebIdentity

Depois de ter um token do OpenID Connect, você pode trocá-lo por AWS credenciais temporárias por meio da solicitação da AssumeRoleWithWebIdentityAPI para AWS Security Token Service ().AWS STS

Como não há nenhuma restrição quanto ao número de identidades que podem ser criadas, é importante compreender as permissões que estão sendo concedidas aos usuários. Configure perfis diferentes do IAM para a aplicação: um para usuários não autenticados e outro para usuários autenticados. O console do Amazon Cognito os criará para você por padrão quando configurar seu banco de identidades pela primeira vez. Esses perfis efetivamente não têm permissões concedidas. Modifique-os de acordo com as suas necessidades.

Saiba mais sobre Permissões e confiança de função.

† O documento jwks_uri padrão do Amazon Cognito Identity contém informações sobre as chaves que assinam tokens para grupos de identidades na maioria das Regiões da AWS. As regiões a seguir têm documentos jwks_uri diferentes.

Amazon Cognito Identity JSON web key URIs in other Regiões da AWS

Região da AWS	Caminho para o documento jwks_uri
AWS GovCloud (Oeste dos EUA)	`https://cognito-identity.us-gov-west-1.amazonaws.com/.well-known/jwks_uri`
China (Pequim)	`https://cognito-identity.cn-north-1.amazonaws.com.cn/.well-known/jwks_uri`
Regiões opcionais, como Europa (Milão) e África (Cidade do Cabo)	`https://cognito-identity.`*`Region`*`.amazonaws.com/.well-known/jwks_uri`

Você também pode extrapolar o jwks_uri do emissor ou `iss` que você recebe no token do OpenID do Amazon Cognito. O endpoint de descoberta padrão do OIDC `<issuer>/.well-known/openid-configuration` lista um caminho para o jwks_uri de seu token.

Perfis do IAM

No processo de criação de um grupo de identidades, será solicitado que você atualize as funções do IAM assumidas por seus usuários. As funções do IAM funcionam assim: quando um usuário faz login na sua aplicação, o Amazon Cognito gera credenciais temporárias da AWS para o usuário. Essas credenciais temporárias são associadas a uma função do IAM específica. Com a função do IAM, você pode definir um conjunto de permissões para acessar os recursos da AWS .

Você pode especificar funções do IAM padrão para usuários autenticados e não autenticados. Além disso, você pode definir regras para escolher a função de cada usuário com base em reivindicações no token de ID do usuário. Para obter mais informações, consulte Controle de acesso com base em perfil.

Por padrão, o console do Amazon Cognito cria funções do IAM que fornecem acesso ao Amazon Mobile Analytics e ao Amazon Cognito Sync. Se desejar, você pode optar por usar as funções do IAM existentes.

Modifique as funções do IAM para permitir ou restringir o acesso a outros serviços. Para isso, faça login no console do IAM. Em seguida, clique em Roles (Funções) e selecione uma função. As políticas anexadas à função selecionada são listadas na guia Permissions (Permissões). Você pode personalizar uma política de acesso clicando no link Manage Policy (Gerenciar política) correspondente. Para saber mais sobre o uso e a definição de políticas, consulte Visão geral de políticas do IAM.

> **Note**
>
> Como uma prática recomendada, defina políticas que sigam os princípios da concessão do privilégio mínimo. Em outras palavras, as políticas incluem somente as permissões que os usuários exigem para executar suas tarefas. Para obter mais informações, consulte Conceder privilégio mínimo no Guia do usuário do IAM.
> Lembre-se de que identidades não autenticadas são assumidas por usuários que não fazem login no seu aplicativo. Normalmente, as permissões que você atribui para identidades não autenticadas devem ser mais restritivas do que aquelas para identidades autenticadas.

Tópicos

Configurar uma política de confiança

O Amazon Cognito utiliza as funções do IAM para gerar credenciais temporárias para os usuários de sua aplicação. O acesso a permissões é controlado pelos relacionamentos de confiança de uma função. Saiba mais sobre Permissões e confiança de função. O Amazon Cognito intermedia conexões entre um pool de AWS STS identidades. IdPs

O token apresentado AWS STS é gerado por um grupo de identidades, que traduz um token de grupo de usuários, rede social ou provedor OIDC, ou uma declaração SAML, em seu próprio token. O token do banco de identidades contém uma declaração aud que é o ID do banco de identidades.

Quando a `Principal` de uma política de confiança de perfil do IAM é uma entidade principal de serviço de bancos de identidades, como `cognito-identity.amazonaws.com`, não é possível criar ou modificar políticas de confiança de perfis de forma que permitam que qualquer banco de identidades assuma o perfil. Com a entidade principal do banco de identidades, o elemento `Action` deve ter uma `Condition` que exija que `AssumeRoleWithWebIndentity` seja executada somente por seus bancos de identidades, conforme especificado por uma chave de condição, como `cognito-identity.amazonaws.com:aud`. Outras chaves de condição estão disponíveis, mas `aud` é obrigatória. Se você tentar salvar uma política de confiança de perfil sem uma condição desse tipo, o IAM retornará um erro.

Para obter mais informações sobre as chaves de federação do OIDC (identidade da web), consulte Chaves disponíveis para a federação do AWS OIDC.

A seguir estão as chaves de condição de federação OIDC disponíveis para o Amazon Cognito.

`cognito-identity.amazonaws.com:aud`

Restringe o perfil às operações de um ou mais bancos de identidades. O Amazon Cognito indica o banco de identidades de origem na declaração `aud` no token do banco de identidades.

`cognito-identity.amazonaws.com:amr`

Restringe o perfil a usuários `authenticated` ou `unauthenticated` (convidados). O Amazon Cognito indica o estado da autenticação na declaração `amr` no token do banco de identidades.

`cognito-identity.amazonaws.com:sub`

Restringe o perfil a um ou mais usuários por UUID. Esse UUID é o ID de identidade do usuário no banco de identidades. Esse valor não é o valor `sub` do provedor de identidades original do usuário. O Amazon Cognito indica esse UUID na declaração `sub` no token do banco de identidades.

O exemplo de política de confiança de funções a seguir permite que o diretor do serviço federado chame `cognito-identity.amazonaws.com` a AWS STS API`AssumeRoleWithWebIdentity`. A solicitação só será bem-sucedida se o token do banco de identidades na solicitação da API tiver as declarações a seguir.

1. Uma declaração `aud` do ID do banco de identidades `us-west-2:abcdefg-1234-5678-910a-0e8443553f95`.

2. Uma declaração `amr` de `authenticated` adicionada quando o usuário faz login e não é um usuário convidado.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "cognito-identity.amazonaws.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "cognito-identity.amazonaws.com:aud": "us-
west-2:abcdefg-1234-5678-910a-0e8443553f95"
                },
                "ForAnyValue:StringLike": {
                    "cognito-identity.amazonaws.com:amr": "authenticated"
                }
            }
        }
    ]
}
```

Políticas de confiança para perfis do IAM na autenticação básica (clássica)

Resumo

Os bancos de identidades só podem assumir funções em nome dos usuários no Fluxo de autenticação básica quando a política de confiança da função de destino contém a condição aud.

A autenticação básica tem a mesma limitação em relação a políticas de confiança de perfil inseguras que a autenticação avançada: não é possível salvar uma política de confiança de perfil que falhe em limitar os bancos de identidades compatíveis com uma condição aud. Essa limitação não foi aplicada quando o serviço foi lançado. Antes da aplicação desse requisito, era possível políticas de confiança de perfil sem condições de segurança adicionais. Após a aplicação desse requisito, AWS

STS permite que as identidades da web assumam funções que não estão protegidas por condições, mas essas funções não podem ser modificadas sem a introdução dessas condições.

A autenticação de fluxo aprimorada exige que o perfil do IAM esteja na mesma Conta da AWS do banco de identidades. Já na autenticação básica, na qual sua aplicação compõe a solicitação `AssumeRoleWithWebIdentity`, sua aplicação pode solicitar assumir um perfil em uma conta diferente. No entanto, a solicitação de assumir um perfil entre contas falhará se o perfil de destino tiver uma política de confiança legada que não imponha a condição `aud`.

O token que um grupo de identidades emite para uma identidade contém informações sobre a origem Conta da AWS do grupo de identidades. Quando você apresenta um token do grupo de identidades em uma solicitação de AssumeRoleWithWebIdentityAPI, AWS STS verifica se o grupo de identidades de origem está na Conta da AWS mesma função do IAM. Se AWS STS determinar que a solicitação é entre contas, ela verifica se a política de confiança da função tem uma `aud` condição. A chamada assume-role falhará se essa condição não estiver presente na política de confiança do perfil. Se a solicitação não for entre contas, essa AWS STS restrição não será aplicada. Como prática recomendada, sempre aplique uma condição desse tipo às políticas de confiança dos perfis do seu banco de identidades.

Veja a seguir um exemplo de política de confiança que atende aos requisitos mínimos de um perfil do IAM para autenticação básica com vários bancos de identidades. Como prática recomendada, permita somente identidades autenticadas com uma condição `"cognito-identity.amazonaws.com:amr": "authenticated"`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "cognito-identity.amazonaws.com"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "cognito-identity.amazonaws.com:aud": [
                        "us-west-2:abcdefg-1234-5678-910a-0e8443553f95",
                        "us-west-2:hijklmo-5678-9101-112b-0e4221776g96",
```

```
                    "us-west-2:pqrstuv-9101-1121-314c-0e2110887h97"
                ]
            }
        }
    }
]
}
```

Condições adicionais da política de confiança

É possível usar as condições de política de confiança a seguir para definir os bancos de identidades de origem, as identidades e os provedores que podem assumir perfis do IAM.

> **Note**
>
> Não implemente a chave de aws:SourceIpcondição nas políticas de confiança para funções do IAM que os grupos de identidades assumem no fluxo de autenticação aprimorado. Como o fluxo aprimorado gera a AssumeRoleWithWebIdentitysolicitação em nome do seu aplicativo, o IP de origem da solicitação não será o IP do cliente do seu aplicativo e a condição nunca será satisfeita. As chaves de condição baseadas em rede são válidas para perfis que os bancos de identidades assumem somente por meio do fluxo básico, que não possui os recursos do lado do serviço do fluxo aprimorado.

Reutilizar funções entre grupos de identidades

Para reutilizar uma função entre vários grupos de identidades, pois eles compartilham o mesmo conjunto de permissões, você pode incluir vários grupos de identidades, como:

```
"StringEquals": {
    "cognito-identity.amazonaws.com:aud": [
        "us-east-1:12345678-abcd-abcd-abcd-123456790ab",
        "us-east-1:98765432-dcba-dcba-dcba-123456790ab"
    ]
}
```

Limitar o acesso a identidades específicas

Para criar uma política limitada a um conjunto específico de usuários de aplicativo, verifique o valor de `cognito-identity.amazonaws.com:sub`:

```
"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"
    ]
}
```

Limitar o acesso a provedores específicos

Para criar uma política limitada a usuários que fizeram login com um provedor específico (talvez seu próprio provedor de login), verifique o valor de `cognito-identity.amazonaws.com:amr`:

```
"ForAnyValue:StringLike": {
    "cognito-identity.amazonaws.com:amr": "login.myprovider.myapp"
}
```

Por exemplo, um aplicativo que confia somente no Facebook, teria a seguinte cláusula amr:

```
"ForAnyValue:StringLike": {
    "cognito-identity.amazonaws.com:amr": "graph.facebook.com"
}
```

Políticas de acesso

As permissões que você atribui a um perfil se aplicam a todos os usuários que assumem esse perfil. Para particionar o acesso dos usuários, use condições e variáveis de política. Para obter mais informações, consulte Elementos de política do IAM: variáveis e etiquetas. Você pode usar a `sub` condição para restringir ações à identidade do Amazon Cognito IDs em suas políticas de acesso. Use essa opção com cuidado, principalmente para identidades não autenticadas, que não têm um ID de usuário consistente. Para obter mais informações sobre as variáveis de política do IAM para federação da web com o Amazon Cognito, consulte IAM e chaves de contexto de AWS STS condição no Guia do AWS Identity and Access Management usuário.

Para proteção de segurança adicional, o Amazon Cognito aplica uma política de restrição de acesso às credenciais que você atribui a usuários não autenticados no fluxo avançado, usando `GetCredentialsForIdentity`. A política de restrição de acesso adiciona um Política de sessão em linha e um AWS política de sessão gerenciada às políticas do IAM que você aplica ao perfil não autenticado. Como você deve conceder acesso em ambas as políticas do IAM para o perfil e

as políticas de sessão, a política de restrição de acesso limita o acesso dos usuários a serviços diferentes dos indicados na lista a seguir.

> **Note**
>
> No fluxo básico (clássico), você faz sua própria solicitação da API AssumeRoleWithWebIdentity e pode aplicar essas restrições à solicitação. Como prática recomendada de segurança, não atribua nenhuma permissão acima dessa política de restrição de acesso a usuários não autenticados.

O Amazon Cognito também impede que usuários autenticados e não autenticados façam solicitações de API aos bancos de identidades do Amazon Cognito e ao Amazon Cognito Sync. Outros Serviços da AWS podem impor restrições ao acesso ao serviço a partir de identidades da web.

Em uma solicitação bem-sucedida com o fluxo avançado, o Amazon Cognito faz uma solicitação da API `AssumeRoleWithWebIdentity` em segundo plano. Entre os parâmetros dessa solicitação, o Amazon Cognito inclui o seguinte.

1. ID de identidade do usuário.
2. O ARN do perfil do IAM que o usuário deseja assumir.
3. Um parâmetro `policy` que adiciona uma política de sessão em linha.
4. Um `PolicyArns.member.N` parâmetro cujo valor é uma política AWS gerenciada que concede permissões adicionais na Amazon CloudWatch.

Serviços que usuários não autenticados podem acessar

Quando você usa o fluxo aprimorado, as políticas de redução de escopo que o Amazon Cognito aplica à sessão do usuário impedem que ele use qualquer serviço diferente dos listados na tabela a seguir. Para um subconjunto de serviços, somente ações específicas são permitidas.

Categoria	Serviço
Analytics	Amazon Data Firehose
	Amazon Managed Service for Apache Flink
Integração de aplicativo	Amazon Simple Queue Service

Categoria	Serviço
AR e VR	Amazon Sumerian[1]
Aplicativos de negócios	Amazon Mobile Analytics Amazon Simple Email Service
Computação	AWS Lambda
Criptografia e PKI	AWS Key Management Service[1]
Banco de dados	Amazon DynamoDB Amazon SimpleDB
Web e móvel de front-end	AWS AppSync Amazon Location Service Amazon Simple Notification Service Amazon Pinpoint Amazon Location Service
Desenvolvimento de jogos	GameLift Servidores Amazon
Internet das coisas (IoT)	AWS IoT

Categoria	Serviço
Machine Learning	Amazon CodeWhisperer Amazon Comprehend Amazon Lex Amazon Machine Learning Amazon Personalize Amazon Polly Amazon Rekognition Amazon SageMaker AI[1] Amazon Textract[1] Amazon Transcribe Amazon Translate
Gestão e Governança	Amazon CloudWatch CloudWatch Registros da Amazon
Redes e entrega de conteúdo	Amazon API Gateway
Segurança, identidade e conformidade	Grupos de usuários do Amazon Cognito
Armazenamento	Amazon Simple Storage Service

[1] Para a tabela a seguir, a política Serviços da AWS em linha concede um subconjunto de ações. A tabela exibe as ações disponíveis em cada uma delas.

AWS service (Serviço da AWS)	Permissões máximas para usuários não autenticados de fluxo avançado

AWS Key Management Service	`Encrypt`
	`Decrypt`
	`ReEncryptTo`
	`ReEncryptFrom`
	`GenerateDataKey`
	`GenerateDataKeyPair`
	`GenerateDataKeyPair`
	`GenerateDataKeyPairWithoutPlaintext`
	`GenerateDataKeyWithoutPlaintext`

SageMaker IA da Amazon	`InvokeEndpoint`
Amazon Textract	`DetectDocumentText` `AnalyzeDocument`
Amazon Sumerian	`View*`

Amazon Location Service	`SearchPlaceIndex*`
	`GetPlace`
	`CalculateRoute*`
	`*Geofence`
	`*Geofences`
	`*DevicePosition*`

Para conceder acesso Serviços da AWS além dessa lista, ative o fluxo de autenticação básico (clássico) em seu grupo de identidades. Se seus usuários virem `NotAuthorizedException` erros Serviços da AWS que são permitidos pelas políticas atribuídas à função do IAM para usuários não autenticados, avalie se você pode remover esse serviço do seu caso de uso. Se você não conseguir, mude para o fluxo básico.

A política de sessão em linha para usuários convidados

O Amazon Cognito primeiro aplica uma política em linha na solicitação de credenciais do IAM. A política de sessão em linha restringe as permissões efetivas do usuário de incluir o acesso a qualquer Serviços da AWS fora daqueles na lista a seguir. Você também deve conceder permissões a eles Serviços da AWS nas políticas que você aplica à função do IAM do usuário. As permissões efetivas de um usuário para uma sessão de perfil assumido são a interseção das políticas atribuídas ao perfil e a política de sessão. Para ter mais informações, consulte Políticas de sessão no Guia do usuário do AWS Identity and Access Management .

O Amazon Cognito adiciona a política em linha a seguir às sessões dos usuários nas Regiões da AWS que estão habilitadas por padrão. Para obter uma visão geral do efeito final da política em linha e de outras políticas de sessão, consulte Serviços que usuários não autenticados podem acessar.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "logs:*",
                "dynamodb:*",
                "kinesis:*",
                "mobileanalytics:*",
                "s3:*",
                "ses:*",
                "sns:*",
                "sqs:*",
                "lambda:*",
                "machinelearning:*",
                "execute-api:*",
```

```
                "iot:*",
                "gamelift:*",
                "cognito-identity:*",
                "cognito-idp:*",
                "lex:*",
                "polly:*",
                "comprehend:*",
                "translate:*",
                "transcribe:*",
                "rekognition:*",
                "mobiletargeting:*",
                "firehose:*",
                "appsync:*",
                "personalize:*",
                "sagemaker:InvokeEndpoint",
                "cognito-sync:*",
                "codewhisperer:*",
                "textract:DetectDocumentText",
                "textract:AnalyzeDocument",
                "sdb:*"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

Para todas as outras regiões, a política de redução do escopo em linha inclui tudo o que está listado nas regiões padrão, exceto as declarações `Action` a seguir.

```
            "cognito-sync:*",
            "sumerian:View*",
            "codewhisperer:*",
            "textract:DetectDocumentText",
            "textract:AnalyzeDocument",
            "sdb:*"
```

A política de sessões AWS gerenciadas para convidados

O Amazon Cognito também aplica uma política AWS gerenciada como política de sessão às sessões de fluxo aprimorado de convidados não autenticados. Essa política limita o escopo das permissões

de usuários não autenticados com `AmazonCognitoUnAuthedIdentitiesSessionPolicy` da política.

Você também deve conceder essa permissão nas políticas vinculadas ao seu perfil do IAM não autenticado. As permissões efetivas de um usuário para uma sessão de perfil assumido são a interseção das políticas do IAM atribuídas ao perfil e as políticas de sessão. Para ter mais informações, consulte Políticas de sessão no Guia do usuário do AWS Identity and Access Management .

Para obter uma visão geral do efeito líquido dessa política AWS gerenciada e de outras políticas de sessão, consulteServiços que usuários não autenticados podem acessar.

A política gerenciada `AmazonCognitoUnAuthedIdentitiesSessionPolicy` contém as permissões a seguir.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "rum:PutRumEvents",
            "polly:*",
            "comprehend:*",
            "translate:*",
            "transcribe:*",
            "rekognition:*",
            "mobiletargeting:*",
            "firehose:*",
            "personalize:*",
            "sagemaker:InvokeEndpoint",
            "geo:GetMap*",
            "geo:SearchPlaceIndex*",
            "geo:GetPlace",
            "geo:CalculateRoute*",
            "geo:*Geofence",
            "geo:*Geofences",
            "geo:*DevicePosition*",
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncryptTo",
```

```
            "kms:ReEncryptFrom",
            "kms:GenerateDataKey",
            "kms:GenerateDataKeyPair",
            "kms:GenerateDataKeyPairWithoutPlaintext",
            "kms:GenerateDataKeyWithoutPlaintext"
        ],
        "Resource": "*"
    }]
}
```

Exemplos de políticas de acesso

Nesta seção, você encontrará exemplos de políticas de acesso do Amazon Cognito que concedem aos usuários as permissões necessárias para realizarem uma operação específica. Você pode limitar ainda mais as permissões de um determinado ID de identidade usando variáveis de política sempre que possível. Por exemplo, usando ${cognito-identity.amazonaws.com:sub}. Para obter mais informações, consulte Entender a autenticação do Amazon Cognito, parte 3: Funções e políticas no blog do AWS Mobile.

> **Note**
>
> Como prática recomendada de segurança, as políticas devem incluir somente as permissões que os usuários exigem para executar suas tarefas. Isso significa que, sempre que possível, você deve tentar definir o escopo de acesso de uma identidade individual para objetos.

Conceder acesso de leitura de identidade a um único objeto no Amazon S3

A seguinte política de acesso concede permissões de leitura a uma identidade para recuperar um único objeto de um determinado bucket do S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject"
```

```
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/assets/my_picture.jpg"]
    }
  ]
}
```

Conceder a uma identidade acesso de leitura e gravação a caminhos específicos de identidade no Amazon S3

A seguinte política de acesso concede permissões de leitura e de gravação para acessar um prefixo específico "folder" em um bucket do S3 ao mapeá-lo para a variável `${cognito-identity.amazonaws.com:sub}`.

Com essa política, uma identidade como `us-east-1:12345678-1234-1234-1234-123456790ab` inserida por `${cognito-identity.amazonaws.com:sub}` poderá obter, colocar e listar objetos no `arn:aws:s3:::amzn-s3-demo-bucket/us-east-1:12345678-1234-1234-1234-123456790ab`. No entanto, a identidade não receberia acesso a outros objetos no `arn:aws:s3:::amzn-s3-demo-bucket`.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${cognito-
identity.amazonaws.com:sub}/*"]}}
    },
    {
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
```

```
      "Resource": ["arn:aws:s3:::amzn-s3-demo-bucket/${cognito-
identity.amazonaws.com:sub}/*"]
    }
  ]
}
```

Um modelo de acesso semelhante é obtido com a Concessão de Acesso do Amazon S3.

Atribuir acesso detalhado ao Amazon DynamoDB para identidades

A política de acesso a seguir fornece controle de acesso granular aos recursos do Amazon DynamoDB usando variáveis de ambiente do Amazon Cognito. Essas variáveis concedem acesso a itens no DynamoDB por meio de ID de identidade. Para obter mais informações, consulte Uso de condições de política do IAM para controle de acesso refinado no Guia do desenvolvedor do Amazon DynamoDB.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:BatchGetItem",
        "dynamodb:Query",
        "dynamodb:PutItem",
        "dynamodb:UpdateItem",
        "dynamodb:DeleteItem",
        "dynamodb:BatchWriteItem"
      ],
      "Resource": [
        "arn:aws:dynamodb:us-west-2:123456789012:table/MyTable"
      ],
      "Condition": {
        "ForAllValues:StringEquals": {
          "dynamodb:LeadingKeys":  ["${cognito-identity.amazonaws.com:sub}"]
        }
      }
    }
```

```
  ]
}
```

Conceder uma permissão de identidade para chamar uma função do Lambda

A política de acesso a seguir concede a uma identidade permissão para invocar uma função do Lambda.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
         {
             "Effect": "Allow",
             "Action": "lambda:InvokeFunction",
             "Resource": [
                 "arn:aws:lambda:us-west-2:123456789012:function:MyFunction"
             ]
         }
     ]
 }
```

Conceder permissão a uma identidade para publicar registros no Kinesis Data Streams

A seguinte política de acesso permite que uma identidade use a operação `PutRecord` com qualquer Kinesis Data Stream. Ela pode ser aplicada a usuários que precisam adicionar registros de dados a todos os streams em uma conta. Para obter mais informações, consulte Controle do acesso aos recursos do Amazon Kinesis Data Streams usando o IAM no Guia do desenvolvedor do Amazon Kinesis Data Streams.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kinesis:PutRecord",
```

```
            "Resource": [
                "arn:aws:kinesis:us-east-1:111122223333:stream/stream1"
            ]
        }
    ]
}
```

Conceder uma identidade acesso aos respectivos dados no armazenamento do Amazon Cognito Sync

A política de acesso a seguir concede a uma identidade permissões apenas para os respectivos dados no armazenamento do Amazon Cognito Sync.

JSON

```
{
  "Version":"2012-10-17",
  "Statement":[{
      "Effect":"Allow",
      "Action":"cognito-sync:*",
      "Resource":["arn:aws:cognito-sync:us-east-1:123456789012:identitypool/
${cognito-identity.amazonaws.com:aud}/identity/${cognito-
identity.amazonaws.com:sub}/*"]
      }]
  }
```

Permissões e confiança de função

A diferença dessas funções está em seus relacionamentos de confiança. Veja a seguir um exemplo de política de confiança para uma função não autenticada:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
```

```
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-corner-
cafe-123456790ab"
        },
        "ForAnyValue:StringLike": {
          "cognito-identity.amazonaws.com:amr": "unauthenticated"
        }
      }
    }
  ]
}
```

Essa política concede a usuários federados do `cognito-identity.amazonaws.com` (o emissor do token do OpenID Connect) permissão para assumir essa função. Além disso, a política restringe o `aud` do token, neste caso o ID do banco de identidades, de acordo com o banco de identidades. Por fim, a política especifica que um dos membros da matriz do da declaração `amr` de múltiplo valor do token emitido pela operação da API `GetOpenIdToken` do Amazon Cognito tem o valor `unauthenticated`.

Quando o Amazon Cognito cria um token, ele define o `amr` do token como `unauthenticated` ou `authenticated`. Se `amr` for `authenticated`, o token incluirá todos os provedores usados durante a autenticação. Isso significa que você pode criar uma função que confie apenas nos usuários que fizeram login por meio do Facebook, alterando a condição `amr` tal como mostrado a seguir:

```
"ForAnyValue:StringLike": {
  "cognito-identity.amazonaws.com:amr": "graph.facebook.com"
}
```

Tenha cuidado ao alterar os relacionamentos de confiança em suas funções ou tentar usar funções entre grupos de identidades. Se você não configurar sua função corretamente para confiar em seu grupo de identidades, ocorrerá uma exceção no STS, semelhante à seguinte:

```
AccessDenied -- Not authorized to perform sts:AssumeRoleWithWebIdentity
```

Se você vir essa mensagem, verifique se seu grupo de identidades e o tipo de autenticação têm uma função apropriada.

Práticas recomendadas de segurança para bancos de identidades do Amazon Cognito

Os grupos de identidade do Amazon Cognito fornecem AWS credenciais temporárias para seu aplicativo. Contas da AWS geralmente contêm os recursos de que os usuários do seu aplicativo precisam e recursos privados de back-end. As funções e políticas do IAM que compõem as AWS credenciais podem conceder acesso a qualquer um desses recursos.

A principal prática recomendada da configuração do banco de identidades é garantir a aplicação funcione sem privilégios excessivos ou indesejados. Para evitar configurações incorretas de segurança, leia estas recomendações antes de lançar cada aplicação que você queira colocar em produção.

Tópicos

Práticas recomendadas para configuração do IAM

Quando um convidado ou usuário autenticado inicia uma sessão na aplicação que exige credenciais do banco de identidades, a aplicação recupera credenciais temporárias da AWS para um perfil do IAM. As credenciais podem ser para um perfil padrão, um perfil escolhido pelas regras na configuração do banco de identidades ou para um perfil personalizado escolhido pela aplicação. Com as permissões atribuídas a cada perfil, seu usuário ganha acesso aos seus recursos da AWS .

Para obter mais informações sobre as melhores práticas gerais do IAM, consulte as melhores práticas do IAM no Guia AWS Identity and Access Management do usuário.

Usar condições de política de confiança nos perfis do IAM

O IAM exige que os perfis dos bancos de identidades tenham pelo menos uma condição de política de confiança. Essa condição pode, por exemplo, definir o escopo da função somente para usuários autenticados. AWS STS também exige que as solicitações de autenticação básica entre contas tenham duas condições específicas: `cognito-identity.amazonaws.com:aud` e. `cognito-`

`identity.amazonaws.com:amr` Como prática recomendada, aplique essas duas condições em todos os perfis do IAM que confiam na entidade principal do serviço dos bancos de identidades de `cognito-identity.amazonaws.com`.

- `cognito-identity.amazonaws.com:aud`: a reivindicação aud no token do banco de identidades deve corresponder a um ID confiável do banco de identidades.
- `cognito-identity.amazonaws.com:amr`: a declaração amr no token do banco de identidade deve ser autenticada ou não autenticada. Com essa condição, você pode reservar o acesso a um perfil somente para convidados não autenticados ou somente para usuários autenticados. Você pode refinar ainda mais o valor dessa condição para restringir o perfil aos usuários de um provedor específico, por exemplo, `graph.facebook.com`.

A política de confiança do perfil do exemplo a seguir concede acesso a um perfil nas seguintes condições:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "us-east-1:a1b2c3d4-5678-90ab-
cdef-EXAMPLE11111"
        },
        "ForAnyValue:StringLike": {
          "cognito-identity.amazonaws.com:amr": "authenticated"
        }
      }
    }
  ]
}
```

Elementos relacionados a bancos de identidades

- `"Federated": "cognito-identity.amazonaws.com"`: os usuários devem vir de um banco de identidades.
- `"cognito-identity.amazonaws.com:aud": "us-east-1:a1b2c3d4-5678-90ab-cdef-example11111"`: os usuários devem vir de um banco de identidades `us-east-1:a1b2c3d4-5678-90ab-cdef-example11111` específico.
- `"cognito-identity.amazonaws.com:amr": "authenticated"`: os usuários devem ser autenticados. Usuários convidados não podem assumir o perfil.

Aplicar permissões de privilégio mínimo

Ao definir permissões com as políticas do IAM para acesso autenticado ou acesso de convidado, conceda apenas as permissões específicas necessárias para executar tarefas específicas, ou permissões de privilégio mínimo. A política do IAM do exemplo a seguir, quando aplicada a um perfil, concede acesso somente leitura a um único arquivo de imagem em um bucket do Amazon S3.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/assets/my_picture.jpg"]
    }
  ]
}
```

Práticas recomendadas de configuração do banco de identidades

Os grupos de identidades têm opções flexíveis para a geração de AWS credenciais. Não use atalhos padrão quando sua aplicação pode funcionar com métodos mais seguros.

Compreender os efeitos do acesso de convidados

O acesso de convidado não autenticado permite que os usuários recuperem seus dados da Conta da AWS antes de fazerem login. Qualquer pessoa que saiba o ID do seu banco de identidades pode solicitar credenciais não autenticadas. Seu ID do banco de identidades não é uma informação confidencial. Quando você ativa o acesso de convidado, as AWS permissões que você concede às sessões não autenticadas ficam disponíveis para todos.

Como prática recomendada, deixe o acesso de convidado desativado e busque os recursos necessários somente depois de autenticar os usuários. Se a aplicação exigir acesso aos recursos antes do login, tome as seguintes precauções:

- Familiarize-se com as limitações automáticas impostas aos perfis não autenticados.
- Monitore e ajuste as permissões dos perfis não autenticados do IAM para atender às necessidades específicas da aplicação.
- Conceda acesso a recursos específicos.
- Proteja a política de confiança do seu perfil padrão não autenticado do IAM.
- Ative o acesso de convidado somente quando tiver certeza de que concederia as permissões do perfil do IAM a qualquer pessoa na Internet.

Usar a autenticação aprimorada por padrão

Com a autenticação básica (clássica), o Amazon Cognito delega a seleção do perfil do IAM à aplicação. Por outro lado, o fluxo aprimorado usa a lógica centralizada do seu banco de identidades para determinar o perfil do IAM. Ele também fornece segurança adicional para identidades não autenticadas com uma política de redução de escopo que define um limite máximo para as permissões do IAM. O fluxo aprimorado é a opção mais segura com o menor nível de esforço do desenvolvedor. Para saber mais sobre essas opções, consulte Fluxo de autenticação dos bancos de identidades.

O fluxo básico pode expor a lógica do lado do cliente que entra na seleção de funções e na montagem da solicitação de credenciais da API AWS STS. O fluxo aprimorado oculta a lógica e a solicitação de assumir perfil por trás da automação do banco de identidades.

Ao configurar a autenticação básica, aplique as práticas recomendadas do IAM aos perfis do IAM e às permissões deles.

Usar provedores de desenvolvedores com segurança

As identidades autenticadas do desenvolvedor são um recurso dos bancos de identidades para aplicações do lado do servidor. A única evidência de autenticação que os grupos de identidades exigem para a autenticação do desenvolvedor são as AWS credenciais de um desenvolvedor do grupo de identidades. Os bancos de identidades não impõem restrição à validade dos identificadores de desenvolvedor-provedor que você apresenta nesse fluxo de autenticação.

Como prática recomendada, implemente somente provedores de desenvolvedores sob as seguintes condições:

- Para criar a responsabilidade pelo uso de credenciais autenticadas pelo desenvolvedor, crie o nome e os identificadores do provedor do desenvolvedor para indicar a fonte de autenticação. Por exemplo: `"Logins" : {"MyCorp provider" : "[provider application ID]"}`.
- Evite credenciais de usuário duradouras. Configure seu cliente do lado do servidor para solicitar identidades com perfis vinculados a serviços, como perfis de instância do EC2 e funções de execução do Lambda.
- Evite misturar fontes de confiança internas e externas no mesmo banco de identidades. Adicione o provedor de desenvolvedor e seus provedores de autenticação única (SSO) em bancos de identidades separados.

Usar atributos para controle de acesso

Os atributos do controle de acesso correspondem à implementação de controle de acesso por atributo (ABAC) nos bancos de identidades do Amazon Cognito. Use as políticas do IAM para controlar o acesso a recursos da AWS por meio de conjuntos de identidades do Amazon Cognito baseados em atributos do usuário. Esses atributos podem ser extraídos de provedores de identidade social e corporativa. Você pode mapear atributos nos tokens de acesso e de ID dos provedores ou asserções SAML para tags que podem ser referenciadas nas políticas de permissões do IAM.

Você pode escolher mapeamentos padrão ou criar seus próprios mapeamentos personalizados nos conjuntos de identidades do Amazon Cognito. Os mapeamentos padrão permitem que você grave políticas do IAM com base em um conjunto fixo de atributos do usuário. Os mapeamentos personalizados permitem que você selecione um conjunto personalizado de atributos de usuário que são referenciados nas políticas de permissões do IAM. Os Attribute names (Nomes de atributo) no console do Amazon Cognito são mapeados para Tag key for principal (Chave de tag para entidade principal), que são as tags referenciadas na política de permissões do IAM.

Por exemplo, vamos supor que você tenha um serviço de transmissão de mídia com uma assinatura gratuita e paga. Você armazena os arquivos de mídia no Amazon S3 e os marca com tags gratuitas ou premium. Você pode usar atributos para controle de acesso a fim de permitir acesso a conteúdo gratuito e pago com base no nível de associação do usuário, que faz parte do perfil do usuário. Você pode mapear o atributo de associação para uma chave de tag para entidade principal a ser passada para a política de permissões do IAM. Dessa forma, você pode criar uma única política de permissões e permitir condicionalmente o acesso a conteúdo premium com base no valor do nível de associação e na tag dos arquivos de conteúdo.

Tópicos

Usar atributos para controlar o acesso traz vários benefícios:

- O gerenciamento de permissões é mais eficiente quando você usa atributos para controle de acesso. Você pode criar uma política de permissões básicas que usa atributos de usuário em vez de criar várias políticas para funções de trabalho diferentes.
- Você não precisa atualizar suas políticas sempre que adicionar ou remover recursos ou usuários da sua aplicação. A política de permissões só concederá acesso aos usuários com os atributos de usuário correspondentes. Por exemplo, talvez seja necessário controlar o acesso a determinados buckets do S3 com base no cargo dos usuários. Nesse caso, você pode criar uma política de permissões para permitir o acesso a esses arquivos somente para usuários dentro do cargo definido. Para obter mais informações, consulte Tutorial do IAM: Usar tags de sessão SAML para ABAC.
- Os atributos podem ser passados como tags de entidades para uma política que permita ou negue permissões com base nos valores destes atributos.

Uso de atributos para controle de acesso com conjuntos de identidades do Amazon Cognito

Antes de usar atributos para controle de acesso, verifique se você atende aos seguintes pré-requisitos:

- Uma AWS conta
- Grupo de usuários
- Grupo de identidades
- Configurar um SDK
- Provedores de identidades integrados
- Credenciais

Para usar atributos para controle de acesso, a Declaração que você define como fonte de dados define o valor da Chave de tag selecionada. O Amazon Cognito aplica a chave e o valor da tag à sessão do usuário. Suas políticas do IAM podem avaliar o acesso do usuário com base na condição `${aws:PrincipalTag/tagkey}`. O IAM avalia o valor da tag do usuário em relação à política.

Você deve preparar perfis do IAM cujas credenciais você deseja passar aos usuários. A política de confiança desses perfis deve permitir que o Amazon Cognito assuma o perfil para o usuário. Quanto a atributos de controle de acesso, você também deve permitir que o Amazon Cognito aplique tags de entidade principal à sessão temporária do usuário. Conceda permissão para assumir a função com a ação AssumeRoleWithWebIdentity. Conceda permissão para marcar as sessões dos usuários com a ação somente com permissão `sts:TagSession`. Para receber mais informações, consulte Passar tags de sessão no AWS Security Token Service no Guia do usuário do AWS Identity and Access Management . Por ver um exemplo de política de confiança que concede as permissões `sts:AssumeRoleWithWebIdentity` e `sts:TagSession` à entidade principal do serviço Amazon Cognito `cognito-identity.amazonaws.com`, consulte Usar atributos para exemplo de política de controle de acesso.

Como configurar atributos para controle de acesso no console

1. Faça login no console do Amazon Cognito e selecione Bancos de identidades. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Localize Provedores de identidade. Selecione o provedor de identidades a ser editado. Se você quiser adicionar um novo IdP, selecione Adicionar provedor de identidade.
4. Para alterar as tags de entidade principal que o Amazon Cognito atribui quando emite credenciais para usuários que se autenticaram com esse provedor, selecione Editar em Atributos para controle de acesso.

a. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

b. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

c. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.

5. Selecione Salvar alterações.

Usar atributos para exemplo de política de controle de acesso

Considere uma situação em que um funcionário do departamento jurídico de uma empresa precisa listar todos os arquivos em buckets que pertencem ao departamento e são classificados com seu nível de segurança. Suponha que o token que esse funcionário recebe do provedor de identidade contenha as seguintes solicitações:

Reivindicações

```
{ .
  .
"sub" : "57e7b692-4f66-480d-98b8-45a6729b4c88",
"department" : "legal",
"clearance" : "confidential",
 .
 .
}
```

Esses atributos podem ser mapeados para tags e referenciados nas políticas de permissões do IAM como tags de entidades. Agora, você pode gerenciar o acesso alterando o perfil do usuário no lado do provedor de identidade. Como alternativa, você pode alterar atributos no lado do recurso usando nomes ou tags sem alterar a própria política.

A política de permissões a seguir faz duas coisas:

- Permite acesso da lista a todos os buckets do S3 que terminam com um prefixo correspondente ao nome do departamento do usuário.

- Permite acesso de leitura em arquivos nesses buckets, desde que a marca de depuração no arquivo corresponda ao atributo de depuração do usuário.

Política de permissões

JSON

```
   {
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:List*",
            "Resource": "arn:aws:s3:::*-${aws:PrincipalTag/department}"
        },
        {
            "Effect": "Allow",
            "Action": "s3:GetObject*",
            "Resource": "arn:aws:s3:::*-${aws:PrincipalTag/department}/*",
            "Condition": {
                "StringEquals": {
                    "s3:ExistingObjectTag/clearance": "${aws:PrincipalTag/
clearance}"
                }
            }
        }
    ]
}
```

A política de confiança determina quem pode assumir essa função. A política de relacionamento de confiança permite o uso de `sts:AssumeRoleWithWebIdentity` e `sts:TagSession` para permitir o acesso. Ela adiciona condições para restringir a política ao banco de identidades que você criou e garante que ela seja para uma função autenticada.

Política de confiança

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRoleWithWebIdentity",
        "sts:TagSession"
      ],
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "IDENTITY-POOL-ID"
        },
        "ForAnyValue:StringLike": {
          "cognito-identity.amazonaws.com:amr": "authenticated"
        }
      }
    }
  ]
}
```

Desativar atributos para controle de acesso (console)

Siga este procedimento para desativar atributos para controle de acesso.

Como desativar atributos para controle de acesso no console

1. Faça login no console do Amazon Cognito e selecione Bancos de identidades. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Localize Provedores de identidade. Selecione o provedor de identidades a ser editado.
4. Selecione Editar em Atributos para controle de acesso.
5. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.
6. Selecione Salvar alterações.

Mapeamentos padrão do provedor

A tabela a seguir possui as informações de mapeamento padrão para os provedores de autenticação que são compatíveis com o Amazon Cognito.

Fornecedor	Tipo de token	Valores de tag da entidade principal	Exemplo
Conjunto de usuários do Amazon Cognito	Token de ID	aud (ID do cliente) e sub (ID do usuário)	“6jk8ltokc7ac9es6jrtg9q572f”, “57e7b692-4f66-480d-98b8-45a6729b4c88”
Facebook	Token de acesso	aud(app_id), sub(user_id)	"492844718097981", "112177216992379"
Google	Token de ID	aud (ID do cliente) e sub (ID do usuário)	"620493171733-eebk7c0hcp5lj3e1tlqp1gntt3k0rncv.apps.googleusercontent.com", "109220063452404746097"
SAML	Asserções	“http://schemas.xmlsoap. org/ws/2005/05/identity/claims/nameidentifier" , "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name”	“auth0|5e28d196f8f55a0eaaa95de3”, “user123@gmail.com”
Apple	Token de ID	aud (ID do cliente) e sub (ID do usuário)	"com.amazonaws.ec2-54-80-172-243.compute-1.client", "001968.a6ca34e9c1e742458a26cf8005854be9.0733"

Fornecedor	Tipo de token	Valores de tag da entidade principal	Exemplo
Amazon	Token de acesso	aud (ID do cliente no Amzn Dev Ac), user_id (ID do usuário)	“amzn1.application-oa2-client.9d70d9382d3446108aaee3dd763a0fa6", “amzn1.account.AGHNIFJQMFSBG36G XCPVB35ORAAA”
Provedores padrão OIDC	Tokens de ID e de acesso	aud (como client_id), sub (como ID do usuário)	"620493171733-eebk7c0hcp5lj3e1tlqp1gntt3k0rncv.apps.googleusercontent.com", "109220063452404746097"
Twitter	Token de acesso	aud (ID da aplicação; segredo da aplicação), sub (ID do usuário)	“DfwifTtKEX1FiIBRn OTl R0CFK; xGJ5xB8xIR Xg W7FxMWC FvNok91y5z1", “IVCPj1269003884292222976" LIdk JJr gwZkLexo
DevAuth	Mapa	Não aplicável	“tag1", “tag2"

> **Note**
>
> A opção de mapeamentos de atributo padrão é preenchida automaticamente para os nomes de Tag Key for Principal (Chave de tag para entidade principal) e Attribute (Atributo). Não é possível alterar os mapeamentos padrão.

Controle de acesso com base em perfil

Os grupos de identidade do Amazon Cognito atribuem aos usuários autenticados um conjunto de credenciais temporárias com privilégios limitados para acessar seus recursos. AWS As permissões para cada usuário são controladas por meio das funções do IAM que você cria. É possível definir regras para escolher a função de cada usuário com base em reivindicações no token de ID do usuário. Você pode definir uma função padrão para usuários autenticados. Você também pode definir uma função do IAM separada com permissões limitadas para usuários convidados que não são autenticados.

Como criar funções para mapeamento de função

É importante adicionar a política de confiança apropriada para cada função para que ela só possa ser assumida pelo Amazon Cognito para os usuários autenticados no grupo de identidades. Aqui está um exemplo dessa política de confiança:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Federated": "cognito-identity.amazonaws.com"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "cognito-identity.amazonaws.com:aud": "us-east-1:12345678-corner-
cafe-123456790ab"
        },
        "ForAnyValue:StringLike": {
          "cognito-identity.amazonaws.com:amr": "authenticated"
        }
      }
    }
  ]
}
```

Essa política permite que os usuários federados do `cognito-identity.amazonaws.com` (o emissor do token do OpenID Connect) assumam essa função. Além disso, a política restringe o `aud` do token, neste caso o ID do banco de identidades, de acordo com o banco de identidades. Por fim, a política especifica que um dos membros da matriz do da declaração `amr` de múltiplo valor do token emitido pela ação da API `GetOpenIdToken` do Amazon Cognito tem o valor `authenticated`.

Conceder permissão para perfil de transmissão

Para permitir que um usuário configure perfis com permissões além das já existentes para o usuário em um grupo de identidades, conceda a ele a permissão `iam:PassRole` para transmitir o perfil à API `set-identity-pool-roles`. Por exemplo, se o usuário não pode gravar no Amazon S3, mas a função do IAM que o usuário configurou no grupo de identidades concede permissão de gravação no Amazon S3, o usuário só pode configurar essa função se a permissão `iam:PassRole` for concedida para a função. O exemplo a seguir mostra como conceder a permissão `iam:PassRole`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1",
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::123456789012:role/myS3WriteAccessRole"
            ]
        }
    ]
}
```

Neste exemplo de política, a permissão `iam:PassRole` é concedida para a função `myS3WriteAccessRole`. A função é especificada usando o nome do recurso da Amazon (ARN) da função. Também é necessário anexar essa política ao usuário. Para obter mais informações, consulte Como trabalhar com políticas gerenciadas.

> **Note**
>
> As funções Lambda usam política baseada em recursos, em que a política é anexada diretamente à própria função do Lambda. Ao criar uma regra que invoca uma função do Lambda, você não transmite uma função. Assim, o usuário que está criando a regra não precisa da permissão do `iam:PassRole`. Para obter mais informações sobre a autorização de funções Lambda, consulte Gerenciar permissões: Usar uma política da função do Lambda.

Como usar tokens para atribuir funções a usuários

Para os usuários que fazem login por meio de grupos de usuários do Amazon Cognito, as funções podem ser passadas no token de ID que foi atribuído pelo grupo de usuários. As funções são exibidas nas seguintes solicitações no token de ID:

- A solicitação `cognito:preferred_role` é o nome de região da Amazon (ARN) da função.
- A `cognito:roles` afirmação é uma string separada por vírgula contendo um conjunto de funções permitidas. ARNs

As solicitações são configuradas da seguinte forma:

- A solicitação `cognito:preferred_role` é configurada como a função do grupo com o melhor (menor) valor `Precedence`. Se há somente uma função permitida, `cognito:preferred_role` é configurado para essa função. Se há várias funções e nenhuma função única tem a melhor precedência, essa solicitação não é configurada.
- A solicitação `cognito:roles` é configurada se há pelo menos uma função.

Ao usar tokens para atribuir funções, se houver várias funções que podem ser atribuídas ao usuário, os grupos de identidades do Amazon Cognito (identidades federadas) escolherão a função da seguinte forma:

- Use o GetCredentialsForIdentity`CustomRoleArn`parâmetro se ele estiver definido e corresponder a uma função na `cognito:roles` declaração. Se esse parâmetro não corresponde a uma função em `cognito:roles`, negue o acesso.
- Se a solicitação `cognito:preferred_role` está configurada, use-a.

- Se a `cognito:preferred_role` declaração não estiver definida, a `cognito:roles` declaração será definida e `CustomRoleArn` não especificada na chamada para`GetCredentialsForIdentity`, a configuração de resolução de função no console ou no `AmbiguousRoleResolution` campo (no `RoleMappings` parâmetro da SetIdentityPoolRolesAPI) será usada para determinar a função a ser atribuída.

Como usar mapeamento baseado em regras para atribuir funções a usuários

As regras permitem que você mapeie solicitações de um token de provedor de identidades para perfis do IAM.

Cada regra especifica uma solicitação de token (como um atributo de usuário no token de ID de um grupo de usuários do Amazon Cognito), o tipo de correspondência, um valor e uma função do IAM. O tipo de correspondência pode ser `Equals`, `NotEqual`, `StartsWith` ou `Contains`. Se um usuário tem um valor correspondente para a solicitação, pode assumir essa função quando recebe credenciais. Por exemplo, é possível criar uma regra que atribui uma função do IAM específica para usuários com um valor de atributo personalizado `custom:dept` de `Sales`.

> **Note**
>
> Nas configurações da regra, os atributos personalizados exigem o prefixo `custom:` para diferenciá-los dos atributos padrão.

As regras são avaliadas em ordem, e a função do IAM para a primeira regra de correspondência é usada, a menos que `CustomRoleArn` seja especificado para substituir a ordem. Para obter mais informações sobre atributos de usuário em grupos de usuários do Amazon Cognito, consulte Trabalhar com atributos do usuário.

Você pode definir várias regras para um provedor de autenticação no console do grupo de identidades (identidades federadas). As regras são aplicadas em ordem. É possível arrastar as regras para alterar a ordem. A primeira regra de correspondência tem precedência. Se o tipo de correspondência é `NotEqual` e a solicitação não existe, a regra não é avaliada. Se não houver correspondência com nenhuma regra, a configuração Resolução de perfil será aplicada a Usar perfil autenticado padrão ou Negar solicitação.

Na API e na CLI, você pode especificar a função a ser atribuída quando nenhuma regra coincide no `AmbiguousRoleResolution` campo do RoleMappingtipo, que é especificado no `RoleMappings` parâmetro da SetIdentityPoolRolesAPI.

Para adicionar mapeamento baseado em regras a um provedor de identidades no console do Amazon Cognito, adicione ou atualize um IdP e selecione Escolher função com regras em Seleção de perfil. A partir daí, você pode adicionar regras que mapeiam as declarações do provedor aos perfis do IAM.

Você pode configurar o mapeamento baseado em regras para provedores de identidade na API AWS CLI ou com o `RulesConfiguration` campo do RoleMappingtipo. Você pode especificar esse campo no `RoleMappings` parâmetro da SetIdentityPoolRolesAPI.

Por exemplo, o AWS CLI comando a seguir adiciona uma regra que atribui a função `arn:aws:iam::123456789012:role/Sacramento_team_S3_admin` aos usuários em sua localização em Sacramento que foram autenticados pelo OIDC IdP: `arn:aws:iam::123456789012:oidc-provider/myOIDCIdP`

```
aws cognito-identity set-identity-pool-roles --region us-east-1 --cli-input-json
 file://role-mapping.json
```

Conteúdo de **`role-mapping.json`**:

```
{
    "IdentityPoolId": "us-east-1:12345678-corner-cafe-123456790ab",
    "Roles": {
        "authenticated": "arn:aws:iam::123456789012:role/myS3WriteAccessRole",
        "unauthenticated": "arn:aws:iam::123456789012:role/myS3ReadAccessRole"
    },
    "RoleMappings": {
        "arn:aws:iam::123456789012:oidc-provider/myOIDCIdP": {
            "Type": "Rules",
            "AmbiguousRoleResolution": "AuthenticatedRole",
            "RulesConfiguration": {
                "Rules": [
                    {
                        "Claim": "locale",
                        "MatchType": "Equals",
                        "Value": "Sacramento",
                        "RoleARN": "arn:aws:iam::123456789012:role/
Sacramento_team_S3_admin"
```

```
                    }
                ]
            }
        }
    }
}
```

Para cada grupo de usuários ou outro provedor de autenticação configurado para um grupo de identidades, é possível criar até 25 regras. Este limite não é ajustável. Para obter mais informações, consulte Cotas no Amazon Cognito.

Declarações de token para uso em mapeamento baseado em regras

Amazon Cognito

Um token de ID do Amazon Cognito é representado como um token web JSON (JWT). O token contém solicitações sobre a identidade do usuário autenticado, como `name`, `family_name` e `phone_number`. Para obter mais informações sobre solicitações padrão, consulte a especificação OpenID Connect. Além das solicitações padrão, os itens a seguir são as solicitações adicionais específicas para o Amazon Cognito:

- `cognito:groups`
- `cognito:roles`
- `cognito:preferred_role`

Amazon

As solicitações a seguir, juntamente com valores possíveis para essas solicitações, podem ser usadas com o Login with Amazon:

- `iss`: www.amazon.com
- `aud`: ID da aplicação
- `sub`: `sub` do token do Login with Amazon

Facebook

As solicitações a seguir, juntamente com valores possíveis para essas solicitações, podem ser usadas com o Facebook:

- `iss`: graph.facebook.com
- `aud`: ID da aplicação
- `sub`: `sub` do token do Facebook

Google

Um token do Google contém solicitações padrão da especificação do OpenID Connect. Todas as solicitações no token do OpenID estão disponíveis para mapeamento com base em regras. Consulte o site do OpenID Connect do Google para saber mais sobre as solicitações disponíveis no token do Google.

Apple

Um token da Apple contém solicitações padrão da especificação do OpenID Connect. Consulte Authenticating Users with Sign in with Apple na documentação da Apple para saber mais sobre a solicitação disponível no token da Apple. O token da Apple nem sempre contém `email`.

OpenID

Todas as solicitações no token do OpenID estão disponíveis para mapeamento com base em regras. Para obter mais informações sobre solicitações padrão, consulte a especificação OpenID Connect. Consulte a documentação do provedor do OpenID para saber mais sobre as solicitações adicionais disponíveis.

SAML

As solicitações são analisadas a partir da declaração do SAML recebida. Todas as solicitações disponíveis na declaração do SAML podem ser usadas no mapeamento com base em regras.

Práticas recomendadas para controle de acesso baseado em função

> **Important**
>
> Se a solicitação que você está mapeando para uma função pode ser modificada pelo usuário final, qualquer usuário final pode assumir a função e definir a política de acordo com a necessidade. Somente mapeie as solicitações que não podem ser configuradas diretamente pelo usuário final para funções com permissões elevadas. Em um grupo de usuários do Amazon Cognito, é possível configurar permissões de leitura e gravação por aplicação para cada atributo de usuário.

> **Important**
>
> Se você configura funções para grupos em um grupo de usuários do Amazon Cognito, essas funções são transmitidas por meio do token de ID do usuário. Para usar essas funções, também é necessário configurar Choose role from token para a seleção de função autenticada para o grupo de identidades.
> Você pode usar a configuração de resolução de função no console e o `RoleMappings` parâmetro da SetIdentityPoolRolesAPI para especificar qual é o comportamento padrão quando a função correta não pode ser determinada a partir do token.

Como obter credenciais

Você pode usar o Amazon Cognito para fornecer credenciais temporárias com privilégios limitados ao seu aplicativo, para que seus usuários possam acessar os recursos. AWS Esta seção descreve como obter credenciais e como recuperar uma identidade do Amazon Cognito de um grupo de identidades.

O Amazon Cognito é compatível com identidades autenticadas e não autenticadas. Usuários não autenticados não têm a identidade verificada, tornando essa função apropriada para usuários convidados de seu aplicativo ou nos casos em que não importa se os usuários têm suas identidades verificadas. Os usuários autenticados fazem login na aplicação por meio de um provedor de identidades de terceiros ou de um grupo de usuários, que verifica as identidades. Certifique-se de definir o escopo das permissões dos recursos de forma apropriada para que você não conceda acesso a eles a partir de usuários não autenticados.

As identidades do Amazon Cognito não são credenciais. Eles são trocados por credenciais usando o suporte à federação de identidade da web no AWS Security Token Service ()AWS STS. A maneira recomendada para obter credenciais da AWS para os usuários da sua aplicação é usar `AWS.CognitoIdentityCredentials`. A identidade no objeto de credenciais é então trocada por credenciais usando. AWS STS

> **Note**
>
> Se você criou o banco de identidades antes de fevereiro de 2015, precisará associar novamente os perfis ao banco de identidades para usar o construtor `AWS.CognitoIdentityCredentials` sem os perfis como parâmetros. Para isso, abra o Console do Amazon Cognito, escolha Gerencie grupos de identidades, selecione seu banco

de identidades, escolha Editar grupo de identidades, especifique suas funções autenticadas e não autenticadas e salve as alterações.

Os provedores de credenciais de identidade da Web fazem parte da cadeia de provedores de credenciais padrão em. AWS SDKs Para definir seu token do grupo de identidades em um `config` arquivo local para um AWS SDK ou para o AWS CLI, adicione uma entrada `web_identity_token_file` de perfil. Consulte Assumir a função de provedor de credenciais no Guia de referência de ferramentas AWS SDKs e ferramentas.

Para saber mais sobre como preencher credenciais de identidade da web em seu SDK, consulte o guia do desenvolvedor do SDK. Para obter melhores resultados, inicie seu projeto com a integração do pool de identidades incorporada ao AWS Amplify.

AWS Recursos do SDK para obter e definir credenciais com grupos de identidades

- Federação de bancos de identidades (Android) no Amplify Dev Center
- Federação de bancos de identidades (iOS) no Amplify Dev Center
- Usando o Amazon Cognito Identity para autenticar usuários no Guia do desenvolvedor AWS SDK para JavaScript
- Provedor de credenciais do Amazon Cognito no Guia do desenvolvedor AWS SDK para .NET
- Especifique as credenciais programaticamente no Guia do desenvolvedor AWS SDK para Go
- Forneça credenciais temporárias em código no Guia do AWS SDK for Java 2.x desenvolvedor
- assumeRoleWithWebIdentityCredentialProviderprovedor no Guia do AWS SDK para PHP desenvolvedor
- Assumir o perfil de provedor de identidades na web na documentação do AWS SDK para Python (Boto3)
- Especificando suas credenciais e a região padrão no Guia do desenvolvedor AWS SDK para Rust

As seções a seguir fornecem exemplos de código em algumas versões antigas AWS SDKs.

Android

Você pode usar o Amazon Cognito para fornecer credenciais temporárias com privilégios limitados ao seu aplicativo, para que seus usuários possam acessar os recursos. AWS O Amazon Cognito é compatível com identidades autenticadas e não autenticadas. Para fornecer AWS credenciais para seu aplicativo, siga as etapas abaixo.

Para usar um pool de identidade do Amazon Cognito em um aplicativo Android, configure. AWS Amplify Para ter mais informações, consulte Autenticação no Amplify Dev Center.

Como recuperar uma identidade do Amazon Cognito

Se você permite usuários não autenticados, pode recuperar um identificador exclusivo (ID de identidade) do Amazon Cognito para o usuário final imediatamente. Se você está autenticando usuários, pode recuperar o ID de identidade depois de configurar os tokens de login no provedor de credenciais:

```
String identityId = credentialsProvider.getIdentityId();
Log.d("LogTag", "my ID is " + identityId);
```

> **Note**
>
> Não chame `getIdentityId()`, `refresh()` ou `getCredentials()` no thread principal do aplicativo. A partir do Android 3.0 (API de nível 11), seu aplicativo falhará automaticamente e lançará um, NetworkOnMainThreadExceptionse você executar uma rede, I/O no thread principal do aplicativo. Você precisa mover o código para uma thread de fundo usando `AsyncTask`. Para obter mais informações, consulte a documentação do Android. Você também pode chamar `getCachedIdentityId()` para recuperar um ID, mas somente se já existe algum em cache localmente. Caso contrário, o método retornará null.

iOS – Objective-C

Você pode usar o Amazon Cognito para fornecer credenciais temporárias com privilégios limitados ao seu aplicativo, para que seus usuários possam acessar os recursos. AWS Os bancos de identidades do Amazon Cognito oferecem suporte a identidades autenticadas e não autenticadas. Para fornecer AWS credenciais ao seu aplicativo, conclua as etapas a seguir.

Para usar um pool de identidade do Amazon Cognito em um aplicativo iOS, configure. AWS Amplify Para ter mais informações, consulte Autenticação do Swift e Autenticação do Flutter no Amplify Dev Center.

Como recuperar uma identidade do Amazon Cognito

É possível recuperar um identificador exclusivo (ID de identidade) do Amazon Cognito para o usuário final imediatamente se você estiver permitindo usuários não autenticados ou depois de configurar os tokens de login no provedor de credenciais se estiver autenticando usuários:

```
// Retrieve your Amazon Cognito ID
[[credentialsProvider getIdentityId] continueWithBlock:^id(AWSTask *task) {
    if (task.error) {
        NSLog(@"Error: %@", task.error);
    }
    else {
        // the task result will contain the identity id
        NSString *cognitoId = task.result;
    }
    return nil;
}];
```

> **Note**
>
> `getIdentityId` é uma chamada assíncrona. Se um ID de identidade já estiver configurado no provedor, você pode chamar `credentialsProvider.identityId` para recuperar essa identidade, que é armazenada em cache localmente. No entanto, se um ID de identidade não estiver configurado no provedor, chamar `credentialsProvider.identityId` retornará `nil`. Para obter mais informações, consulte a Referência do Amplify iOS SDK.

iOS – Swift

Você pode usar o Amazon Cognito para fornecer credenciais temporárias com privilégios limitados ao seu aplicativo para que seus usuários possam acessar os recursos. AWS O Amazon Cognito é compatível com identidades autenticadas e não autenticadas. Para fornecer AWS credenciais para seu aplicativo, siga as etapas abaixo.

Para usar um pool de identidade do Amazon Cognito em um aplicativo iOS, configure. AWS Amplify Para ter mais informações, consulte Autenticação do Swift no Amplify Dev Center.

Como recuperar uma identidade do Amazon Cognito

É possível recuperar um identificador exclusivo (ID de identidade) do Amazon Cognito para o usuário final imediatamente se você estiver permitindo usuários não autenticados ou depois de configurar os tokens de login no provedor de credenciais se estiver autenticando usuários:

```
// Retrieve your Amazon Cognito ID
credentialsProvider.getIdentityId().continueWith(block: { (task) -> AnyObject? in
```

```
    if (task.error != nil) {
        print("Error: " + task.error!.localizedDescription)
    }
    else {
        // the task result will contain the identity id
        let cognitoId = task.result!
        print("Cognito id: \(cognitoId)")
    }
    return task;
})
```

> **Note**
>
> `getIdentityId` é uma chamada assíncrona. Se um ID de identidade já estiver configurado no provedor, você pode chamar `credentialsProvider.identityId` para recuperar essa identidade, que é armazenada em cache localmente. No entanto, se um ID de identidade não estiver configurado no provedor, chamar `credentialsProvider.identityId` retornará `nil`. Para obter mais informações, consulte a Referência do Amplify iOS SDK.

JavaScript

Se você ainda não tiver criado, crie um grupo de identidades no console do Amazon Cognito antes de usar `AWS.CognitoIdentityCredentials`.

Depois de configurar um grupo de identidades com seus provedores de identidades, você poderá usar `AWS.CognitoIdentityCredentials` para autenticar usuários. Para configurar as credenciais de seu aplicativo para usar `AWS.CognitoIdentityCredentials`, defina a propriedade `credentials` do `AWS.Config` ou uma configuração por serviço. O exemplo a seguir usa `AWS.Config`:

```
// Set the region where your identity pool exists (us-east-1, eu-west-1)
AWS.config.region = 'us-east-1';

// Configure the credentials provider to use your identity pool
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
    IdentityPoolId: 'IDENTITY_POOL_ID',
    Logins: { // optional tokens, used for authenticated login
        'graph.facebook.com': 'FBTOKEN',
```

```
        'www.amazon.com': 'AMAZONTOKEN',
        'accounts.google.com': 'GOOGLETOKEN',
        'appleid.apple.com': 'APPLETOKEN'
    }
});

// Make the call to obtain credentials
AWS.config.credentials.get(function(){

    // Credentials will be available when this function is called.
    var accessKeyId = AWS.config.credentials.accessKeyId;
    var secretAccessKey = AWS.config.credentials.secretAccessKey;
    var sessionToken = AWS.config.credentials.sessionToken;

});
```

A propriedade opcional `Logins` é um mapa de nomes de provedor de identidade para os tokens de identidade para esses provedores. Como você obtém o token do seu provedor de identidade depende do provedor que usa. Por exemplo, se o Facebook for um de seus provedores de identidade, você poderá usar a `FB.login` função do Facebook SDK para obter um token de provedor de identidade:

```
FB.login(function (response) {
    if (response.authResponse) { // logged in
        AWS.config.credentials = new AWS.CognitoIdentityCredentials({
          IdentityPoolId: 'us-east-1:1699ebc0-7900-4099-b910-2df94f52a030',
          Logins: {
            'graph.facebook.com': response.authResponse.accessToken
          }
        });

        console.log('You are now logged in.');
    } else {
        console.log('There was a problem logging you in.');
    }
});
```

Como recuperar uma identidade do Amazon Cognito

É possível recuperar um identificador exclusivo (ID de identidade) do Amazon Cognito para o usuário final imediatamente se você estiver permitindo usuários não autenticados ou depois de configurar os tokens de login no provedor de credenciais se estiver autenticando usuários:

```
var identityId = AWS.config.credentials.identityId;
```

Unity

Você pode usar o Amazon Cognito para fornecer credenciais temporárias com privilégios limitados ao seu aplicativo, para que seus usuários possam acessar os recursos. AWS O Amazon Cognito é compatível com identidades autenticadas e não autenticadas. Para fornecer AWS credenciais para seu aplicativo, siga as etapas abaixo.

O AWS SDK for Unity agora faz parte do SDK para .NET. Para começar a usar o Amazon Cognito no SDK para .NET, consulte o provedor de credenciais do Amazon Cognito no Guia do desenvolvedor. AWS SDK para .NET Ou consulte o Amplify Dev Center para obter opções para criar um aplicativo com. AWS Amplify

Como recuperar uma identidade do Amazon Cognito

É possível recuperar um identificador exclusivo (ID de identidade) do Amazon Cognito para o usuário final imediatamente se você estiver permitindo usuários não autenticados ou depois de configurar os tokens de login no provedor de credenciais se estiver autenticando usuários:

```
credentials.GetIdentityIdAsync(delegate(AmazonCognitoIdentityResult<string> result) {
    if (result.Exception != null) {
        //Exception!
    }
    string identityId = result.Response;
});
```

Xamarin

Você pode usar o Amazon Cognito para fornecer credenciais temporárias com privilégios limitados ao seu aplicativo para que seus usuários possam acessar os recursos. AWS O Amazon Cognito é compatível com identidades autenticadas e não autenticadas. Para fornecer AWS credenciais para seu aplicativo, siga as etapas abaixo.

O AWS SDK for Xamarin agora faz parte do SDK para .NET. Para começar a usar o Amazon Cognito no SDK para .NET, consulte o provedor de credenciais do Amazon Cognito no Guia do desenvolvedor. AWS SDK para .NET Ou consulte o Amplify Dev Center para obter opções para criar um aplicativo com. AWS Amplify

> **Note**
>
> Observação: se você criou o grupo de identidades antes de fevereiro de 2015, precisa reassociar as funções ao grupo de identidades para usar esse construtor sem as funções como parâmetros. Para isso, abra o Console do Amazon Cognito, escolha Manage identity pools (Gerenciar grupos de identidades), selecione seu grupo de identidades, escolha Edit Identity Pool (Editar grupo de identidades), especifique suas funções autenticadas e não autenticadas e salve as alterações.

Como recuperar uma identidade do Amazon Cognito

É possível recuperar um identificador exclusivo (ID de identidade) do Amazon Cognito para o usuário final imediatamente se você estiver permitindo usuários não autenticados ou depois de configurar os tokens de login no provedor de credenciais se estiver autenticando usuários:

```
var identityId = await credentials.GetIdentityIdAsync();
```

Acessando Serviços da AWS com credenciais temporárias

O resultado de uma autenticação bem-sucedida com um banco de identidades é um conjunto de credenciais da AWS . Com essas credenciais, seu aplicativo pode fazer solicitações para AWS recursos protegidos com a autenticação do IAM. Com as várias AWS SDKs que você pode adicionar aos seus aplicativos para acessar as operações de API de grupos de identidades, você pode fazer solicitações de API não autenticadas que produzem credenciais temporárias. Em seguida, você pode adicionar SDKs outros Serviços da AWS ao seu cliente e assinar solicitações com essas credenciais temporárias. As permissões do IAM concedidas ao seu perfil de credenciais temporárias devem permitir as operações que você solicita de outros serviços.

Depois de configurar seu provedor de credenciais do Amazon Cognito e recuperar as AWS credenciais, crie um cliente. AWS service (Serviço da AWS) Veja a seguir alguns exemplos da documentação do AWS SDK.

AWS Recursos do SDK para criar um cliente

- AWS Configuração do cliente no Guia do AWS SDK para C++ desenvolvedor
- Usando a AWS SDK para Go V2 com o Serviços da AWS Guia do AWS SDK para Go Desenvolvedor

- Configurando clientes HTTP no Guia do AWS SDK for Java 2.x desenvolvedor
- Criação e chamada de objetos de serviço no Guia do AWS SDK para JavaScript desenvolvedor
- Criação de clientes na AWS SDK para Python (Boto3) documentação
- Criação de um cliente de serviço no Guia do AWS SDK para Rust desenvolvedor
- Usando clientes no Guia do AWS SDK para Swift desenvolvedor

O seguinte trecho inicializa um cliente do Amazon DynamoDB:

Android

Para usar um pool de identidade do Amazon Cognito em um aplicativo Android, configure. AWS Amplify Para ter mais informações, consulte Autenticação no Amplify Dev Center.

```
// Create a service client with the provider
AmazonDynamoDB client = new AmazonDynamoDBClient(credentialsProvider);
```

O provedor de credenciais se comunica com o Amazon Cognito, recuperando o identificador exclusivo para usuários autenticados e não autenticados, bem como as credenciais temporárias com privilégios limitados para o Mobile SDK. AWS AWS As credenciais recuperadas são válidas por uma hora, e o provedor as atualiza quando elas expiram.

iOS – Objective-C

Para usar um pool de identidade do Amazon Cognito em um aplicativo iOS, configure. AWS Amplify Para ter mais informações, consulte Autenticação do Swift e Autenticação do Flutter no Amplify Dev Center.

```
// create a configuration that uses the provider
AWSServiceConfiguration *configuration = [AWSServiceConfiguration
 configurationWithRegion:AWSRegionUSEast1 provider:credentialsProvider];
// get a client with the default service configuration
AWSDynamoDB *dynamoDB = [AWSDynamoDB defaultDynamoDB];
```

O provedor de credenciais se comunica com o Amazon Cognito, recuperando o identificador exclusivo para usuários autenticados e não autenticados, bem como as credenciais temporárias com privilégios limitados para o Mobile SDK. AWS AWS As credenciais recuperadas são válidas por uma hora, e o provedor as atualiza quando elas expiram.

iOS – Swift

Para usar um pool de identidade do Amazon Cognito em um aplicativo iOS, configure. AWS Amplify Para ter mais informações, consulte Autenticação do Swift no Amplify Dev Center.

```
// get a client with the default service configuration
let dynamoDB = AWSDynamoDB.default()

// get a client with a custom configuration
AWSDynamoDB.register(with: configuration!, forKey: "USWest2DynamoDB");
let dynamoDBCustom = AWSDynamoDB(forKey: "USWest2DynamoDB")
```

O provedor de credenciais se comunica com o Amazon Cognito, recuperando o identificador exclusivo para usuários autenticados e não autenticados, bem como as credenciais temporárias com privilégios limitados para o Mobile SDK. AWS AWS As credenciais recuperadas são válidas por uma hora, e o provedor as atualiza quando elas expiram.

JavaScript

```
// Create a service client with the provider
var dynamodb = new AWS.DynamoDB({region: 'us-west-2'});
```

O provedor de credenciais se comunica com o Amazon Cognito, recuperando o identificador exclusivo para usuários autenticados e não autenticados, bem como as credenciais temporárias com privilégios limitados para o Mobile SDK. AWS AWS As credenciais recuperadas são válidas por uma hora, e o provedor as atualiza quando elas expiram.

Unity

O AWS SDK for Unity agora faz parte do SDK para .NET. Para começar a usar o Amazon Cognito no SDK para .NET, consulte o provedor de credenciais do Amazon Cognito no Guia do desenvolvedor. AWS SDK para .NET Ou consulte o Amplify Dev Center para obter opções para criar um aplicativo com. AWS Amplify

```
// create a service client that uses credentials provided by Cognito
AmazonDynamoDBClient client = new AmazonDynamoDBClient(credentials, REGION);
```

O provedor de credenciais se comunica com o Amazon Cognito, recuperando o identificador exclusivo para usuários autenticados e não autenticados, bem como as credenciais temporárias com

privilégios limitados para o Mobile SDK. AWS AWS As credenciais recuperadas são válidas por uma hora, e o provedor as atualiza quando elas expiram.

Xamarin

O AWS SDK for Xamarin agora faz parte do SDK para .NET. Para começar a usar o Amazon Cognito no SDK para .NET, consulte o provedor de credenciais do Amazon Cognito no Guia do desenvolvedor. AWS SDK para .NET Ou consulte o Amplify Dev Center para obter opções para criar um aplicativo com. AWS Amplify

```
// create a service client that uses credentials provided by Cognito
var client = new AmazonDynamoDBClient(credentials, REGION)
```

O provedor de credenciais se comunica com o Amazon Cognito, recuperando o identificador exclusivo para usuários autenticados e não autenticados, bem como as credenciais temporárias com privilégios limitados para o Mobile SDK. AWS AWS As credenciais recuperadas são válidas por uma hora, e o provedor as atualiza quando elas expiram.

Bancos de identidades: provedores de identidade de terceiros

Com os grupos de identidade do Amazon Cognito, você pode se integrar a uma variedade de provedores de identidade externos (IdPs) para fornecer AWS credenciais temporárias por meio de autenticação federada em seu aplicativo. Ao configurar seu grupo de identidades para trabalhar com esses recursos externos IdPs, você pode autorizar o acesso aos AWS recursos de back-end para seus usuários com autenticação por grupos de usuários, provedores sociais, provedores de OIDC ou provedores de SAML do Amazon Cognito. Esta seção aborda as etapas de configuração e integração IdPs com seu pool de identidade do Amazon Cognito.

Usando a propriedade `logins`, é possível definir as credenciais recebidas de um provedor de identidade (IdP). Você também pode associar um grupo de identidades a vários IdPs. Por exemplo, é possível definir tokens do Facebook e do Google na propriedade `logins` para associar a identidade exclusiva do Amazon Cognito aos logins de ambos os IdPs. O usuário pode autenticar com ambas as contas, mas o Amazon Cognito retorna o mesmo identificador de usuário.

As instruções a seguir orientam você na autenticação com os grupos de identidades IdPs compatíveis com o Amazon Cognito.

Tópicos

Configurar o Facebook como um IdP de bancos de identidades

Os bancos de identidades do Amazon Cognito trabalham com o Facebook para fornecer autenticação federada aos usuários da aplicação. Esta seção explica como inscrever e configurar a aplicação com o Facebook como IdP.

Configurar o Facebook

Registre seu aplicativo no Facebook antes de autenticar os usuários do Facebook e interagir com o Facebook APIs.

O Portal de desenvolvedores do Facebook ajuda você a configurar sua aplicação. Siga esse procedimento antes de integrar o Facebook no grupo de identidades do Amazon Cognito:

> **Note**
>
> A federação de bancos de identidades do Amazon Cognito não é compatível com o Login limitado do Facebook. Para obter mais informações sobre como configurar o Login do Facebook para iOS sem exceder as permissões definidas para o Login limitado, consulte Login do Facebook para iOS - Início rápido, em Meta para desenvolvedores.

Configurar o Facebook

1. No Facebook Developers portal, faça login com as credenciais do Facebook.
2. No menu Apps, selecione Add a New App.
3. Selecione uma plataforma e conclua o processo de início rápido.

Android

Para obter mais informações sobre como integrar aplicativos Android ao Login do Facebook, consulte o Guia de conceitos básicos do Facebook.

iOS – Objective-C

Para obter mais informações sobre como integrar aplicativos iOS Objective-C ao Login do Facebook, consulte o Guia de conceitos básicos do Facebook.

iOS – Swift

Para obter mais informações sobre como integrar aplicativos iOS Swift ao Login do Facebook, consulte o Guia de conceitos básicos do Facebook.

JavaScript

Para obter mais informações sobre como integrar aplicativos JavaScript da web com o Login do Facebook, consulte o Guia de introdução do Facebook.

Configurar um provedor de identidades no console de bancos de identidades do Amazon Cognito

Use o procedimento a seguir para configurar seu provedor de identidades.

Como adicionar um provedor de identidades (IdP) Facebook

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Selecione Adicionar provedor de identidade.
4. Selecione Facebook.
5. Insira o ID do aplicativo do OAuth projeto que você criou no Meta for Developers. Para ter mais informações, consulte Login do Facebook nos documentos do Meta for Developers.
6. Para alterar o perfil que o Amazon Cognito solicita ao emitir credenciais para usuários que se autenticaram com esse provedor, defina Configurações de perfil.

    - Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras.

i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.

7. Para alterar as tags de identidade principal que o Amazon Cognito atribui ao emitir credenciais para usuários que se autenticaram com esse provedor, configure Atributos para controle de acesso.

   a. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

   b. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

   c. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.

8. Selecione Salvar alterações.

Uso do Facebook

Android

Para adicionar a autenticação do Facebook, primeiro siga o Guia do Facebook para integrar o SDK do Facebook à aplicação. Em seguida, adicione um botão Login with Facebook à interface do usuário Android. O SDK do Facebook usa um objeto de sessão para rastrear o estado. O Amazon Cognito usa o token de acesso desse objeto de sessão para autenticar o usuário, gerar o identificador exclusivo e, se necessário, conceder ao usuário acesso a outros recursos. AWS

Depois de autenticar o usuário com o SDK do Facebook, adicione o token de sessão ao provedor de credenciais do Amazon Cognito.

Facebook SDK 4.0 ou posterior:

```
Map<String, String> logins = new HashMap<String, String>();
logins.put("graph.facebook.com", AccessToken.getCurrentAccessToken().getToken());
```

```
credentialsProvider.setLogins(logins);
```

Facebook SDK antes de 4.0:

```
Map<String, String> logins = new HashMap<String, String>();
logins.put("graph.facebook.com", Session.getActiveSession().getAccessToken());
credentialsProvider.setLogins(logins);
```

O processo de login do Facebook inicializa uma sessão singleton no respectivo SDK. O objeto de sessão do Facebook contém um OAuth token que o Amazon Cognito usa para gerar AWS credenciais para seu usuário final autenticado. O Amazon Cognito também usa o token para verificar se existe um usuário no banco de dados de usuário que corresponda a essa identidade específica do Facebook. Se o usuário já existe, a API retorna o identificador existente. Do contrário, a API retorna um novo identificador. Os identificadores são automaticamente armazenados em cache no dispositivo local pelo SDK cliente.

> **Note**
>
> Depois de definir o mapa de logins, faça uma chamada para `refresh` ou `get` para recuperar as AWS credenciais.

iOS – Objective-C

Para adicionar a autenticação do Facebook, primeiro siga o Guia do Facebook para integrar o SDK do Facebook à aplicação. Em seguida, adicione um botão Login with Facebook à interface de usuário. O SDK do Facebook usa um objeto de sessão para rastrear o estado. O Amazon Cognito usa o token de acesso desse objeto de sessão para autenticar o usuário e vinculá-lo a grupos de identidades exclusivas do Amazon Cognito (identidades federadas).

Para fornecer o token de acesso do Facebook ao Amazon Cognito, implemente o protocolo AWSIdentityProviderManager.

Ao implementar o método `logins`, retorne um dicionário contendo `AWSIdentityProviderFacebook`. Esse dicionário atua como a chave, ao passo que o token de acesso atual do usuário autenticado do Facebook atua como o valor, conforme mostrado no exemplo de código a seguir.

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
```

```
    FBSDKAccessToken* fbToken = [FBSDKAccessToken currentAccessToken];
    if(fbToken){
        NSString *token = fbToken.tokenString;
        return [AWSTask taskWithResult: @{ AWSIdentityProviderFacebook : token }];
    }else{
            return [AWSTask taskWithError:[NSError errorWithDomain:@"Facebook Login"
                                                              code:-1
                                                          userInfo:@{@"error":@"No current
 Facebook access token"}]];
    }
}
```

Ao instanciar o `AWSCognitoCredentialsProvider`, passe a classe que implementa `AWSIdentityProviderManager` como o valor de `identityProviderManager` no construtor. Para obter mais informações, acesse a página de AWSCognitoCredentialsProviderreferência e escolha initWithRegionTipo:identityPoolId: identityProviderManager.

iOS – Swift

Para adicionar a autenticação do Facebook, primeiro siga o Guia do Facebook para integrar o SDK do Facebook à aplicação. Em seguida, adicione um botão Login with Facebook à interface de usuário. O SDK do Facebook usa um objeto de sessão para rastrear o estado. O Amazon Cognito usa o token de acesso desse objeto de sessão para autenticar o usuário e vinculá-lo a grupos de identidades exclusivas do Amazon Cognito (identidades federadas).

> **Note**
>
> A federação de bancos de identidades do Amazon Cognito não é compatível com o Login limitado do Facebook. Para obter mais informações sobre como configurar o Login do Facebook para iOS sem exceder as permissões definidas para o Login limitado, consulte Login do Facebook para iOS - Início rápido, em Meta para desenvolvedores.

Para fornecer o token de acesso do Facebook ao Amazon Cognito, implemente o protocolo AWSIdentityProviderManager.

Na implementação do método `logins`, retorne um dicionário contendo `AWSIdentityProviderFacebook`. Esse dicionário atua como a chave, ao passo que o token de acesso atual do usuário autenticado do Facebook atua como o valor, conforme mostrado no exemplo de código a seguir.

```
class FacebookProvider: NSObject, AWSIdentityProviderManager {
    func logins() -> AWSTask<NSDictionary> {
        if let token = AccessToken.current?.authenticationToken {
            return AWSTask(result: [AWSIdentityProviderFacebook:token])
        }
        return AWSTask(error:NSError(domain: "Facebook Login", code: -1 , userInfo:
 ["Facebook" : "No current Facebook access token"]))
    }
}
```

Ao instanciar o `AWSCognitoCredentialsProvider`, passe a classe que implementa `AWSIdentityProviderManager` como o valor de `identityProviderManager` no construtor. Para obter mais informações, acesse a página de AWSCognitoCredentialsProviderreferência e escolha initWithRegionTipo:identityPoolId: identityProviderManager.

JavaScript

Para adicionar a autenticação do Facebook, siga o Login do Facebook para web e adicione o botão Login with Facebook ao seu site. O SDK do Facebook usa um objeto de sessão para rastrear o estado. O Amazon Cognito usa o token de acesso desse objeto de sessão para autenticar o usuário, gerar o identificador exclusivo e, se necessário, conceder ao usuário acesso a outros recursos. AWS

Depois de autenticar o usuário com o SDK do Facebook, adicione o token de sessão ao provedor de credenciais do Amazon Cognito.

```
FB.login(function (response) {

  // Check if the user logged in successfully.
  if (response.authResponse) {

    console.log('You are now logged in.');

    // Add the Facebook access token to the Amazon Cognito credentials login map.
    AWS.config.credentials = new AWS.CognitoIdentityCredentials({
      IdentityPoolId: 'IDENTITY_POOL_ID',
      Logins: {
        'graph.facebook.com': response.authResponse.accessToken
      }
    });

    // Obtain AWS credentials
    AWS.config.credentials.get(function(){
```

```
        // Access AWS resources here.
    });

  } else {
    console.log('There was a problem logging you in.');
  }

});
```

O SDK do Facebook obtém um OAuth token que o Amazon Cognito usa para AWS gerar credenciais para seu usuário final autenticado. O Amazon Cognito também usa o token para fazer a verificação em relação ao banco de dados de usuário quanto à existência de um usuário que corresponda a essa identidade específica do Facebook. Se o usuário já existe, a API retorna o identificador existente. Caso contrário, um novo identificador é retornado. Identificadores são automaticamente armazenados em cache pelo cliente SDK no dispositivo local.

> **Note**
>
> Depois de configurar o mapa de logins, chame `refresh` ou `get` para obter as credenciais. Para obter um exemplo de código, consulte “Caso de uso 17, Integrando grupos de usuários com a Identidade Cognito”, JavaScript no arquivo README.

Unity

Para adicionar a autenticação do Facebook, primeiro siga o Guia do Facebook para integrar o SDK do Facebook à aplicação. O Amazon Cognito usa o token de acesso do Facebook do objeto FB para gerar um identificador exclusivo do usuário que está associado a uma identidade do Amazon Cognito.

Depois de autenticar o usuário com o SDK do Facebook, adicione o token de sessão ao provedor de credenciais do Amazon Cognito:

```
void Start()
{
    FB.Init(delegate() {
        if (FB.IsLoggedIn) { //User already logged in from a previous session
            AddFacebookTokenToCognito();
        } else {
            FB.Login ("email", FacebookLoginCallback);
        }
```

```
    });
}

void FacebookLoginCallback(FBResult result)
{
    if (FB.IsLoggedIn)
    {
        AddFacebookTokenToCognito();
    }
    else
    {
        Debug.Log("FB Login error");
    }
}

void AddFacebookTokenToCognito()
{
    credentials.AddLogin ("graph.facebook.com",
 AccessToken.CurrentAccessToken.TokenString);
}
```

Antes de usar `FB.AccessToken`, chame `FB.Login()` e verifique se `FB.IsLoggedIn` é verdadeiro.

Xamarin

Xamarin para Android:

```
public void InitializeFacebook() {
    FacebookSdk.SdkInitialize(this.ApplicationContext);
    callbackManager = CallbackManagerFactory.Create();
    LoginManager.Instance.RegisterCallback(callbackManager, new FacebookCallback &lt;
 LoginResult &gt; () {
      HandleSuccess = loginResult = &gt; {
        var accessToken = loginResult.AccessToken;
        credentials.AddLogin("graph.facebook.com", accessToken.Token);
        //open new activity
      },
      HandleCancel = () = &gt; {
        //throw error message
      },
      HandleError = loginError = &gt; {
        //throw error message
```

```
      }
    });
    LoginManager.Instance.LogInWithReadPermissions(this, new List &lt; string &gt; {
      "public_profile"
    });
  }
```

Xamarin para iOS:

```
public void InitializeFacebook() {
  LoginManager login = new LoginManager();
  login.LogInWithReadPermissions(readPermissions.ToArray(),
 delegate(LoginManagerLoginResult result, NSError error) {
    if (error != null) {
      //throw error message
    } else if (result.IsCancelled) {
      //throw error message
    } else {
      var accessToken = loginResult.AccessToken;
      credentials.AddLogin("graph.facebook.com", accessToken.Token);
      //open new view controller
    }
  });
}
```

Configurar o Login with Amazon como um IdP de bancos de identidades

Os bancos de identidades do Amazon Cognito trabalham com o Login with Amazon para fornecer autenticação federada aos usuários da aplicação Web e do aplicativo móvel. Esta seção explica como inscrever e configurar a aplicação com o Login with Amazon como provedor de identidade (IdP).

No Portal do desenvolvedor, configure o Login with Amazon para funcionar com o Amazon Cognito. Para obter mais informações, consulte Configuração do Login with Amazon em Perguntas frequentes sobre Login with Amazon.

> **Note**
>
> Para integrar o Login with Amazon a uma aplicação Xamarin, siga o Guia de conceitos básicos do Xamarin.

> **Note**
>
> Você não pode integrar nativamente o Login with Amazon na plataforma Unity. Em vez disso, use uma visualização da Web e siga o fluxo de login do navegador.

Como configurar o Login with Amazon

Implementar o Login with Amazon

No portal do desenvolvedor da Amazon, você pode configurar um OAuth aplicativo para se integrar ao seu grupo de identidades, encontrar a documentação do Login with Amazon e fazer o download SDKs. No Portal do desenvolvedor, escolha Developer console (Console do desenvolvedor) e, em seguida, Login with Amazon. Você pode criar um perfil de segurança e, em seguida, mecanismos de autenticação do Login with Amazon em sua aplicação. Consulte Como obter credenciais para obter mais informações sobre como integrar a autenticação Login with Amazon à sua aplicação.

A Amazon emite um ID de cliente OAuth 2.0 para seu novo perfil de segurança. Você pode encontrar o ID de cliente na guia do perfil de segurança Web Settings (Configurações da Web). Digite o ID do perfil de segurança no campo ID da aplicação do Login com Amazon IdP no banco de identidades.

> **Note**
>
> Digite o ID do perfil de segurança no campo ID da aplicação do Login com Amazon IdP no banco de identidades. Isso difere dos grupos de usuários, que usam ID do cliente.

Configurar o provedor externo no console do Amazon Cognito

Como adicionar um login com o provedor de identidades (IdP) da Amazon

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Selecione Adicionar provedor de identidade.
4. Selecione Login with Amazon.

5. Insira o ID do aplicativo do OAuth projeto que você criou em Login with Amazon. Para ter mais informações, consulte a documentação do Login with Amazon.
6. Para alterar o perfil que o Amazon Cognito solicita ao emitir credenciais para usuários que se autenticaram com esse provedor, defina Configurações de perfil.

    - Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras.

        i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

        ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.
7. Para alterar as tags de identidade principal que o Amazon Cognito atribui ao emitir credenciais para usuários que se autenticaram com esse provedor, configure Atributos para controle de acesso.

    a. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

    b. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

    c. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.
8. Selecione Salvar alterações.

Use o Login with Amazon: Android

Depois de autenticar o login da Amazon, você pode passar o token para o provedor de credenciais do Amazon Cognito no método OnSuccess da interface. TokenListener O código é semelhante a:

```
@Override
public void onSuccess(Bundle response) {
    String token = response.getString(AuthzConstants.BUNDLE_KEY.TOKEN.val);
    Map<String, String> logins = new HashMap<String, String>();
```

```
    logins.put("www.amazon.com", token);
    credentialsProvider.setLogins(logins);
}
```

Use o Login with Amazon: iOS – Objective-C

Depois de autenticar o login da Amazon, você pode passar o token para o provedor de credenciais do Amazon Cognito requestDidSucceed no método de: AMZNAccess TokenDelegate

```
- (void)requestDidSucceed:(APIResult \*)apiResult {
    if (apiResult.api == kAPIAuthorizeUser) {
        [AIMobileLib getAccessTokenForScopes:[NSArray arrayWithObject:@"profile"]
 withOverrideParams:nil delegate:self];
    }
    else if (apiResult.api == kAPIGetAccessToken) {
        credentialsProvider.logins = @{ @(AWSCognitoLoginProviderKeyLoginWithAmazon):
 apiResult.result };
    }
}}
```

Use o Login with Amazon: iOS – Swift

Depois de autenticar o login da Amazon, você pode passar o token para o provedor de credenciais do Amazon Cognito no método `requestDidSucceed` de `AMZNAccessTokenDelegate`:

```
func requestDidSucceed(apiResult: APIResult!) {
    if apiResult.api == API.AuthorizeUser {
        AIMobileLib.getAccessTokenForScopes(["profile"], withOverrideParams: nil,
 delegate: self)
    } else if apiResult.api == API.GetAccessToken {
        credentialsProvider.logins =
 [AWSCognitoLoginProviderKey.LoginWithAmazon.rawValue: apiResult.result]
    }
}
```

Use o Login with Amazon: JavaScript

Depois que o usuário se autentica com o Login with Amazon e é redirecionado de volta para o site, o access_token Login with Amazon é fornecido na string de consulta. Passe esse token para mapear as credenciais de login.

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
```

```
    IdentityPoolId: 'IDENTITY_POOL_ID',
    Logins: {
        'www.amazon.com': 'Amazon Access Token'
    }
});
```

Configurar o Google como um IdP do banco de identidades

Os bancos de identidades do Amazon Cognito trabalham com o Facebook para fornecer autenticação federada aos usuários da aplicação móvel. Esta seção explica como inscrever e configurar a aplicação com o Google como IdP.

Android

> **Note**
>
> Se a sua aplicação usar o Google e estiver disponível em várias plataformas móveis, você deverá configurá-lo como um provedor OpenID Connect. Adicione todos os clientes criados IDs como valores adicionais de público para uma melhor integração. Para saber mais sobre o modelo de identidade entre clientes do Google, consulte Identidade em vários clientes.

Como configurar o Google

Para ativar o Login do Google para Android, crie um projeto de console do Google Developers para a aplicação.

1. Vá para o console do Google Developers e crie um novo projeto.
2. Escolha APIs & Serviços e, em seguida, tela de OAuth consentimento. Personalize as informações que o Google mostra aos usuários quando ele solicita o consentimento deles para compartilhar os dados do perfil com sua aplicação.
3. Escolha Credentials (Credenciais) e, em seguida, Create credentials (Criar credenciais). Escolha a ID OAuth do cliente. Selecione Android como Application type (Tipo de aplicação). Crie um ID de cliente distinto para cada plataforma em que você desenvolve sua aplicação.
4. Em Credenciais (Credenciais), escolha Manage service accounts (Gerenciar contas de serviço). Escolha Create service account (Criar conta de serviço). Insira os detalhes de sua conta de serviço e, em seguida, escolha Create and continue (Criar e continuar).

5. Conceda à conta de serviço acesso ao seu projeto. Conceda aos usuários acesso à conta de serviço conforme a aplicação exigir.
6. Escolha sua nova conta de serviço, a guia Keys (Chaves) e Add key (Adicionar chave). Crie e baixe uma nova chave JSON.

Para obter mais informações sobre como usar o console Google Developers, consulte Como criar e gerenciar projetos na documentação do Google Cloud.

Para obter mais informações sobre como integrar o Google ao aplicativo Android, consulte Autenticar usuários com Login com o Google na documentação do Google Identity.

Como adicionar um provedor de identidades (IdP) Google

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Selecione Adicionar provedor de identidade.
4. Selecione Google.
5. Insira o ID do cliente do OAuth projeto que você criou no Google Cloud Platform. Para mais informações, consulte Configuração OAuth 2.0 na Ajuda do Console do Google Cloud Platform.
6. Para alterar o perfil que o Amazon Cognito solicita ao emitir credenciais para usuários que se autenticaram com esse provedor, defina Configurações de perfil.

    - Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras.

        i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

        ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.

7. Para alterar as tags de identidade principal que o Amazon Cognito atribui ao emitir credenciais para usuários que se autenticaram com esse provedor, configure Atributos para controle de acesso.

   a. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

   b. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

   c. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.

8. Selecione Salvar alterações.

Usar o Google

Para habilitar o login com o Google na aplicação, siga as instruções na documentação do Google para Android. Quando um usuário faz login, ele solicita um token de autenticação OpenID Connect do Google. Em seguida, o Amazon Cognito usa o token para autenticar o usuário e gerar um identificador exclusivo.

O código de exemplo a seguir mostra como recuperar o token de autenticação do Google Play Service:

```
GooglePlayServicesUtil.isGooglePlayServicesAvailable(getApplicationContext());
AccountManager am = AccountManager.get(this);
Account[] accounts = am.getAccountsByType(GoogleAuthUtil.GOOGLE_ACCOUNT_TYPE);
String token = GoogleAuthUtil.getToken(getApplicationContext(), accounts[0].name,
        "audience:server:client_id:YOUR_GOOGLE_CLIENT_ID");
Map<String, String> logins = new HashMap<String, String>();
logins.put("accounts.google.com", token);
credentialsProvider.setLogins(logins);
```

iOS – Objective-C

> **Note**
>
> Se a sua aplicação usar o Google e estiver disponível em várias plataformas móveis, configure-o como um provedor OpenID Connect. Adicione todos os clientes criados IDs como

valores adicionais de público para uma melhor integração. Para saber mais sobre o modelo de identidade entre clientes do Google, consulte Identidade em vários clientes.

Como configurar o Google

Para habilitar o Login do Google para iOS, crie um projeto de console do Google Developers para a aplicação.

1. Vá para o console do Google Developers e crie um novo projeto.
2. Escolha APIs & Serviços e, em seguida, tela de OAuth consentimento. Personalize as informações que o Google mostra aos usuários quando ele solicita o consentimento deles para compartilhar os dados do perfil com sua aplicação.
3. Escolha Credentials (Credenciais) e, em seguida, Create credentials (Criar credenciais). Escolha a ID OAuth do cliente. Selecione iOS como Application type (Tipo de aplicação). Crie um ID de cliente distinto para cada plataforma em que você desenvolve sua aplicação.
4. Em Credenciais (Credenciais), escolha Manage service accounts (Gerenciar contas de serviço). Escolha Create service account (Criar conta de serviço). Insira os detalhes de sua conta de serviço e escolha Create and continue (Criar e continuar).
5. Conceda à conta de serviço acesso ao seu projeto. Conceda aos usuários acesso à conta de serviço conforme a aplicação exigir.
6. Escolha sua nova conta de serviço. Escolha a guia Keys (Chaves) e Add key (Adicionar chave). Crie e baixe uma nova chave JSON.

Para obter mais informações sobre como usar o console Google Developers, consulte Como criar e gerenciar projetos na documentação do Google Cloud.

Para mais informações sobre a integração do Google ao aplicativo iOS, consulte a Google Sign-In for iOS na documentação do Google Identity.

Como adicionar um provedor de identidades (IdP) Google

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Selecione Adicionar provedor de identidade.

4. Selecione Google.
5. Insira o ID do cliente do OAuth projeto que você criou no Google Cloud Platform. Para mais informações, consulte Configuração OAuth 2.0 na Ajuda do Console do Google Cloud Platform.
6. Para alterar o perfil que o Amazon Cognito solicita ao emitir credenciais para usuários que se autenticaram com esse provedor, defina Configurações de perfil.

    - Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras.

        i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

        ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.
7. Para alterar as tags de identidade principal que o Amazon Cognito atribui ao emitir credenciais para usuários que se autenticaram com esse provedor, configure Atributos para controle de acesso.

    a. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

    b. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

    c. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.
8. Selecione Salvar alterações.

Usar o Google

Para habilitar o login com o Google no aplicativo, siga a documentação do Google para iOS. A autenticação bem-sucedida resulta em um token de autenticação OpenID Connect, que o Amazon Cognito usa para autenticar o usuário e gerar um identificador exclusivo.

A autenticação bem-sucedida resulta em um objeto `GTMOAuth2Authentication` que contém um `id_token`, que o Amazon Cognito usa para autenticar o usuário e gerar um identificador exclusivo:

```
- (void)finishedWithAuth: (GTMOAuth2Authentication *)auth error: (NSError *) error {
        NSString *idToken = [auth.parameters objectForKey:@"id_token"];
        credentialsProvider.logins = @{ @(AWSCognitoLoginProviderKeyGoogle): idToken };
    }
```

iOS – Swift

> **Note**
>
> Se a sua aplicação usar o Google e estiver disponível em várias plataformas móveis, configure-o como um provedor OpenID Connect. Adicione todos os clientes criados IDs como valores adicionais de público para uma melhor integração. Para saber mais sobre o modelo de identidade entre clientes do Google, consulte Identidade em vários clientes.

Como configurar o Google

Para habilitar o Login do Google para iOS, crie um projeto de console do Google Developers para a aplicação.

1. Vá para o console do Google Developers e crie um novo projeto.
2. Escolha APIs & Serviços e, em seguida, tela de OAuth consentimento. Personalize as informações que o Google mostra aos usuários quando ele solicita o consentimento deles para compartilhar os dados do perfil com sua aplicação.
3. Escolha Credentials (Credenciais) e, em seguida, Create credentials (Criar credenciais). Escolha a ID OAuth do cliente. Selecione iOS como Application type (Tipo de aplicação). Crie um ID de cliente distinto para cada plataforma em que você desenvolve sua aplicação.
4. Em Credenciais (Credenciais), escolha Manage service accounts (Gerenciar contas de serviço). Escolha Create service account (Criar conta de serviço). Insira os detalhes de sua conta de serviço e escolha Create and continue (Criar e continuar).
5. Conceda à conta de serviço acesso ao seu projeto. Conceda aos usuários acesso à conta de serviço conforme a aplicação exigir.
6. Escolha sua nova conta de serviço, a guia Keys (Chaves) e Add key (Adicionar chave). Crie e baixe uma nova chave JSON.

Para obter mais informações sobre como usar o console Google Developers, consulte Como criar e gerenciar projetos na documentação do Google Cloud.

Para mais informações sobre a integração do Google ao aplicativo iOS, consulte a Google Sign-In for iOS na documentação do Google Identity.

Escolha Manage Identity Pools (Gerenciar grupos de identidades) na página inicial do console do Amazon Cognito:

Configurar o provedor externo no console do Amazon Cognito

1. Escolha o nome do grupo de identidades no qual deseja habilitar o Google como provedor externo. A página Dashboard (Painel) do grupo de identidades será exibida.
2. No canto superior direito da página Dashboard (Painel), selecione Edit identity pool (Editar grupo de identidades). A página Edit identity pool (Editar grupo de identidades) será exibida.
3. Role para baixo e escolha Authentication providers (Provedores de autenticação) para expandir a seção.
4. Escolha a guia Google.
5. Selecione Unlock (Desbloquear).
6. Insira o ID de cliente do Google que você obteve do Google e escolha Save Changes (Salvar alterações).

Usar o Google

Para habilitar o login com o Google no aplicativo, siga a documentação do Google para iOS. A autenticação bem-sucedida resulta em um token de autenticação OpenID Connect, que o Amazon Cognito usa para autenticar o usuário e gerar um identificador exclusivo.

A autenticação bem-sucedida resulta em um objeto `GTMOAuth2Authentication` que contém um `id_token`. O Amazon Cognito usa esse token para autenticar o usuário e gerar um identificador exclusivo:

```
func finishedWithAuth(auth: GTMOAuth2Authentication!, error: NSError!) {
    if error != nil {
      print(error.localizedDescription)
    }
    else {
      let idToken = auth.parameters.objectForKey("id_token")
      credentialsProvider.logins = [AWSCognitoLoginProviderKey.Google.rawValue:
 idToken!]
    }
```

```
}
```

JavaScript

> **Note**
>
> Se a sua aplicação usar o Google e estiver disponível em várias plataformas móveis, você deverá configurá-lo como provedor OpenID Connect. Adicione todos os clientes criados IDs como valores adicionais de público para uma melhor integração. Para saber mais sobre o modelo de identidade entre clientes do Google, consulte Identidade em vários clientes.

Como configurar o Google

Para ativar o login do Google em um aplicativo JavaScript da web, crie um projeto de console do Google Developers para seu aplicativo.

1. Vá para o console do Google Developers e crie um novo projeto.
2. Escolha APIs & Serviços e, em seguida, tela de OAuth consentimento. Personalize as informações que o Google mostra aos usuários quando ele solicita o consentimento deles para compartilhar os dados do perfil com sua aplicação.
3. Escolha Credentials (Credenciais) e, em seguida, Create credentials (Criar credenciais). Escolha a ID OAuth do cliente. Selecione Web application (Aplicação Web) como Application type (Tipo de aplicação). Crie um ID de cliente distinto para cada plataforma em que você desenvolve sua aplicação.
4. Em Credenciais (Credenciais), escolha Manage service accounts (Gerenciar contas de serviço). Escolha Create service account (Criar conta de serviço). Insira os detalhes de sua conta de serviço e escolha Create and continue (Criar e continuar).
5. Conceda à conta de serviço acesso ao seu projeto. Conceda aos usuários acesso à conta de serviço conforme a aplicação exigir.
6. Escolha sua nova conta de serviço, a guia Keys (Chaves) e Add key (Adicionar chave). Crie e baixe uma nova chave JSON.

Para obter mais informações sobre como usar o console Google Developers, consulte Como criar e gerenciar projetos na documentação do Google Cloud.

Para obter mais informações sobre como integrar o Google à aplicação Web, consulte Sign in With Google na documentação do Google Identity.

Configurar o provedor externo no console do Amazon Cognito

Como adicionar um provedor de identidades (IdP) Google

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Selecione Adicionar provedor de identidade.
4. Selecione Google.
5. Insira o ID do cliente do OAuth projeto que você criou no Google Cloud Platform. Para mais informações, consulte Configuração OAuth 2.0 na Ajuda do Console do Google Cloud Platform.
6. Para alterar o perfil que o Amazon Cognito solicita ao emitir credenciais para usuários que se autenticaram com esse provedor, defina Configurações de perfil.
   - Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras.
     i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.
     ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.
7. Para alterar as tags de identidade principal que o Amazon Cognito atribui ao emitir credenciais para usuários que se autenticaram com esse provedor, configure Atributos para controle de acesso.
   a. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.
   b. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

c. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.

8. Selecione Salvar alterações.

Usar o Google

Para habilitar o login com o Google no aplicativo, siga a documentação do Google para Web.

A autenticação bem-sucedida resulta em um objeto de resposta contendo um `id_token`, que o Amazon Cognito usa para autenticar o usuário e gerar um identificador exclusivo:

```
function signinCallback(authResult) {
  if (authResult['status']['signed_in']) {

     // Add the Google access token to the Amazon Cognito credentials login map.
     AWS.config.credentials = new AWS.CognitoIdentityCredentials({
        IdentityPoolId: 'IDENTITY_POOL_ID',
        Logins: {
           'accounts.google.com': authResult['id_token']
        }
     });

     // Obtain AWS credentials
     AWS.config.credentials.get(function(){
        // Access AWS resources here.
     });
  }
}
```

Configurar o Login com a Apple como um IdP do banco de identidades

Os bancos de identidades do Amazon Cognito trabalham com o recurso Fazer login com a Apple para fornecer autenticação federada aos usuários da aplicação Web e móvel. Esta seção explica como inscrever e configurar a aplicação usando Sign in with Apple como provedor de identidade (IdP).

Para adicionar Sign in with Apple como provedor de autenticação para um grupo de identidades, você deve realizar dois procedimentos. Primeiro, integre o Sign in with Apple a uma aplicação e, em seguida, configure-o nos grupos de identidades. Para up-to-date obter mais informações sobre como

configurar o Login com a Apple, consulte Configurando seu ambiente para fazer login com a Apple na documentação do desenvolvedor da Apple.

Configurar o Sign in with Apple

Para configurar Sign in with Apple como IdP, é necessário inscreva sua aplicação na Apple para receber o ID de cliente.

1. Crie uma conta de desenvolvedor com a Apple.
2. Faça login com as credenciais da Apple.
3. No painel de navegação esquerdo, escolha Certificados IDs e perfis.
4. No painel de navegação à esquerda, escolha Identificadores.
5. Na página Identifiers (Identificadores), escolha o ícone +.
6. Na página Registrar um novo identificador, escolha Aplicativo e IDs, em seguida, escolha Continuar.
7. Na página Register an App ID (Registrar ID de uma aplicação), faça o seguinte:
    a. Em Description (Descrição), digite uma descrição.
    b. Em ID do pacote, digite um identificador. Anote esse ID de pacote, pois você precisará desse valor para configurar a Apple como provedor no grupo de identidades.
    c. Em Capabilities (Recursos), escolha Sign In with Apple (Fazer login com a Apple) e, depois, selecione Edit (Editar).
    d. Na página Sign in with Apple: configuração do ID da aplicação, selecione a configuração adequada para sua aplicação. Em seguida, escolha Salvar.
    e. Escolha Continue (Continuar).
8. Na página Confirm your App ID (Confirmar ID do seu app), escolha Register (Registrar).
9. Siga para a etapa 10 se quiser integrar o recurso Fazer login com a Apple a uma aplicação iOS nativa. A etapa 11 é para aplicativos que você deseja integrar ao recurso Fazer login com o Apple JS.
10. Na página Identificadores, escolha o IDs menu Aplicativo e, em seguida, Serviços IDs. Escolha o ícone +.
11. Na página Registrar um novo identificador, escolha Serviços e IDs, em seguida, escolha Continuar.
12. Na página Register a Services ID (Registrar um ID de serviços), faça o seguinte:

a. Em Description (Descrição), digite uma descrição.

b. Em Identifier (Identificador), digite um identificador. Anote o ID de serviços, pois você precisará desse valor para configurar a Apple como provedor no grupo de identidades.

c. Selecione Fazer login com a Apple e escolha Configurar.

d. Na página Web Authentication Configuration (Configuração de autenticação na web), escolha um Primary App ID (ID de app primário). Em Site URLs, escolha o ícone +. Em Domínios e subdomínios, insira o nome de domínio do seu aplicativo. Em Return URLs, insira o URL de retorno de chamada para o qual a autorização redireciona o usuário após a autenticação por meio do Sign in with Apple.

e. Escolha Próximo.

f. Escolha Continue (Continuar) e, depois, Register (Registrar).

13. No painel de navegação à esquerda, selecione Chaves.
14. Na página Keys (Chaves), escolha o ícone +.
15. Na página Register a New Key (Registrar uma chave nova), faça o seguinte:

a. Em Key Name (Nome da chave), digite um nome de chave.

b. Escolha Sign In with Apple (Fazer login com a Apple) e escolha Configure (Configurar).

c. Na página Configurar chave, escolha um ID de aplicativo primário e selecione Salvar.

d. Escolha Continue (Continuar) e, depois, Register (Registrar).

> **Note**
>
> Para integrar o recurso Fazer login com a Apple a um aplicativo iOS nativo, consulte Implementar a autenticação de usuário com o recurso Fazer login com a Apple.
> Para integrar o recurso Fazer login com a Apple em uma plataforma diferente do iOS nativo, consulte Fazer login com o Apple JS.

Configurar o provedor externo no console de identidades federadas do Amazon Cognito

Use o procedimento a seguir para configurar seu provedor externo.

Como adicionar um provedor de identidades (IdP) Sign in with Apple

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Selecione Adicionar provedor de identidade.
4. Selecione Sign in with Apple.
5. Insira o ID de serviços do OAuth projeto que você criou com o Apple Developer. Para ter mais informações, consulte Authenticating users with Sign in with Apple na Documentação do Sign in with Apple.
6. Para alterar o perfil que o Amazon Cognito solicita ao emitir credenciais para usuários que se autenticaram com esse provedor, defina Configurações de perfil.

   - Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras.

     i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

     ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.
7. Para alterar as tags de identidade principal que o Amazon Cognito atribui ao emitir credenciais para usuários que se autenticaram com esse provedor, configure Atributos para controle de acesso.

   a. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

   b. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

   c. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.
8. Selecione Salvar alterações.

Sign in with Apple como provedor nos exemplos de CLI de identidades federadas do Amazon Cognito

Esse exemplo cria um grupo de identidades denominado `MyIdentityPool` com o Sign in with Apple como IdP.

```
aws cognito-identity create-identity-pool --identity-
pool-name MyIdentityPool --supported-login-providers
appleid.apple.com="sameple.apple.clientid"
```

Para obter mais informações, consulte Criar grupo de identidades

Gerar um ID de identidade do Amazon Cognito

Esse exemplo gera (ou recupera) um ID do Amazon Cognito. Esta é uma API pública, portanto você não precisa de credenciais para chamar essa API.

```
aws cognito-identity get-id --identity-pool-id SampleIdentityPoolId --
logins appleid.apple.com="SignInWithAppleIdToken"
```

Para obter mais informações, consulte get-id.

Obter credenciais para um ID de identidade do Amazon Cognito

Este exemplo retorna credenciais para o ID de identidade fornecido e o recurso Fazer login com a Apple. Esta é uma API pública, portanto você não precisa de credenciais para chamar essa API.

```
aws cognito-identity get-credentials-for-identity --identity-id
SampleIdentityId --logins appleid.apple.com="SignInWithAppleIdToken"
```

Para obter mais informações, consulte get-credentials-for-identity.

Usar o recurso Fazer login com a Apple: Android

A Apple não fornece um SDK compatível com o recurso Fazer login com a Apple para Android. Em vez disso, é possível usar o fluxo da Web em uma visualização da Web.

- Para configurar o recurso Fazer login com a Apple no aplicativo, siga Configuring Your Web page for Sign In with Apple na documentação da Apple.
- Para adicionar um botão Sign in with Apple à interface de usuário do Android, siga Displaying Sign in with Apple buttons on the web na documentação da Apple.

- Para autenticar usuários com segurança usando Sign in with Apple, siga Authenticating Users with Sign In with Apple (Autenticar usuários com o Sign in with Apple) na documentação da Apple.

Fazer login com a Apple usa um objeto de sessão para rastrear o estado. O Amazon Cognito usa o token de ID desse objeto de sessão para autenticar o usuário, gerar o identificador exclusivo e, se necessário, conceder ao usuário acesso a outros recursos. AWS

```
@Override
public void onSuccess(Bundle response) {
    String token = response.getString("id_token");
    Map<String, String> logins = new HashMap<String, String>();
    logins.put("appleid.apple.com", token);
    credentialsProvider.setLogins(logins);
}
```

Usar o recurso Fazer login com a Apple: iOS – Objective-C

A Apple forneceu suporte ao SDK para o recurso Fazer login com a Apple em aplicativos nativos do iOS. Para implementar a autenticação de usuário com o recurso Fazer login com a Apple em dispositivos nativos do iOS, siga Implementing User Authentication with Sign in with Apple na documentação da Apple.

O Amazon Cognito usa o token de ID para autenticar o usuário, gerar o identificador exclusivo e, se necessário, conceder ao usuário acesso a outros recursos. AWS

```
(void)finishedWithAuth: (ASAuthorizationAppleIDCredential *)auth error: (NSError *)
 error {
        NSString *idToken = [ASAuthorizationAppleIDCredential
 objectForKey:@"identityToken"];
        credentialsProvider.logins = @{ "appleid.apple.com": idToken };
    }
```

Usar o recurso Fazer login com a Apple: iOS – Swift

A Apple forneceu suporte ao SDK para o recurso Fazer login com a Apple em aplicativos nativos do iOS. Para implementar a autenticação de usuário com o recurso Fazer login com a Apple em dispositivos nativos do iOS, siga Implementing User Authentication with Sign in with Apple na documentação da Apple.

O Amazon Cognito usa o token de ID para autenticar o usuário, gerar o identificador exclusivo e, se necessário, conceder ao usuário acesso a outros recursos. AWS

Para obter mais informações sobre como configurar o Sign in with Apple no iOS, consulte Sign in with Apple.

```
func finishedWithAuth(auth: ASAuthorizationAppleIDCredential!, error: NSError!) {
    if error != nil {
      print(error.localizedDescription)
    }
    else {
      let idToken = auth.identityToken,
      credentialsProvider.logins = ["appleid.apple.com": idToken!]
    }
}
```

Use o Login com a Apple: JavaScript

A Apple não fornece um SDK compatível com o Sign in with Apple for JavaScript. Em vez disso, é possível usar o fluxo da Web em uma visualização da Web.

- Para configurar o recurso Fazer login com a Apple no aplicativo, siga Configuring Your Web page for Sign In with Apple na documentação da Apple.
- Para adicionar um botão Fazer login com a Apple à sua interface de JavaScript usuário, siga Exibindo os botões de login com a Apple na web na documentação da Apple.
- Para autenticar usuários com segurança usando Sign in with Apple, siga Authenticating Users with Sign In with Apple (Autenticar usuários com o Sign in with Apple) na documentação da Apple.

Fazer login com a Apple usa um objeto de sessão para rastrear o estado. O Amazon Cognito usa o token de ID desse objeto de sessão para autenticar o usuário, gerar o identificador exclusivo e, se necessário, conceder ao usuário acesso a outros recursos. AWS

```
function signinCallback(authResult) {
     // Add the apple's id token to the Amazon Cognito credentials login map.
     AWS.config.credentials = new AWS.CognitoIdentityCredentials({
        IdentityPoolId: 'IDENTITY_POOL_ID',
        Logins: {
           'appleid.apple.com': authResult['id_token']
        }
```

```
      });

      // Obtain AWS credentials
      AWS.config.credentials.get(function(){
         // Access AWS resources here.
      });
}
```

Configurar um provedor OIDC como um IdP do banco de identidades

O OpenID Connect é um padrão aberto para autenticação que é compatível com vários provedores de login. O Amazon Cognito permite vincular identidades com provedores OpenID Connect que você configura por meio do AWS Identity and Access Management.

Como adicionar um provedor OpenID Connect

Para ter informações sobre como criar um provedor OpenID Connect, consulte Criar provedores de identidades OpenID Connect (OIDC) no Guia do usuário do AWS Identity and Access Management .

Como associar um provedor ao Amazon Cognito

Como adicionar um provedor de identidades (IdP) OIDC

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Selecione Adicionar provedor de identidade.
4. Selecione OpenID Connect (OIDC).
5. Escolha um provedor de identidade OIDC do IAM IdPs em seu. Conta da AWS Se você quiser adicionar um novo provedor SAML, selecione Criar provedor para navegar até o console do IAM.
6. Para alterar o perfil que o Amazon Cognito solicita ao emitir credenciais para usuários que se autenticaram com esse provedor, defina Configurações de perfil.

    - Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras.

        i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir

quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.

7. Para alterar as tags de identidade principal que o Amazon Cognito atribui ao emitir credenciais para usuários que se autenticaram com esse provedor, configure Atributos para controle de acesso.

   a. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

   b. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

   c. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.

8. Selecione Salvar alterações.

É possível associar vários provedores OpenID Connect a um único grupo de identidades.

Uso do OpenID Connect

Consulte a documentação do provedor sobre como fazer login e receber um token de ID.

Assim que tiver um token, adicione-o ao mapa de logins. Use o URI do provedor como chave.

Como validar um token de OpenID Connect

Ao fazer a primeira integração com o Amazon Cognito, você poderá receber uma exceção `InvalidToken`. É importante entender como o Amazon Cognito valida tokens de OpenID Connect.

> **Note**
>
> Conforme especificado aqui (https://tools.ietf.org/html/rfc7523), o Amazon Cognito oferece um período de carência de 5 minutos para lidar com qualquer distorção do relógio entre os sistemas.

1. O parâmetro `iss` deve corresponder à chave que o mapa de logins usa (por exemplo, login.provider.com).
2. A assinatura deve ser válida. A assinatura deve ser verificável por meio de uma chave de ativação pública RSA.

> **Note**
>
> Os bancos de identidades mantêm um cache da chave de assinatura do IdP OIDC por um breve período. Se o provedor alterar a chave de assinatura, o Amazon Cognito poderá retornar um erro `NoKeyFound` até que esse cache seja atualizado. Se você encontrar esse erro, aguarde cerca de 10 minutos para que o banco de identidades atualize a chave de assinatura.

3. A impressão digital da chave pública do certificado corresponde à impressão digital que você definiu no IAM quando criou seu provedor OIDC.
4. Se o `azp` parâmetro estiver presente, verifique esse valor em relação ao cliente listado IDs em seu provedor OIDC.
5. Se o `azp` parâmetro não estiver presente, verifique o `aud` parâmetro em relação ao cliente listado IDs em seu provedor OIDC.

O site jwt.io é um recurso valioso que você pode usar para decodificar tokens e verificar esses valores.

Android

```
Map<String, String> logins = new HashMap<String, String>();
logins.put("login.provider.com", token);
credentialsProvider.setLogins(logins);
```

iOS – Objective-C

```
credentialsProvider.logins = @{ "login.provider.com": token }
```

JavaScript

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
```

```
  IdentityPoolId: 'IDENTITY_POOL_ID',
  Logins: {
     'login.provider.com': token
  }
});
```

Configurar um provedor SAML como um IdP do banco de identidades

Com os grupos de identidade do Amazon Cognito, você pode autenticar usuários com provedores de identidade (IdPs) por meio do SAML 2.0. No Amazon Cognito, é possível usar um IdP compatível com SAML para fornecer um fluxo simples de integração aos usuários. Seu IdP compatível com SAML especifica as funções do IAM que seus usuários podem assumir. Dessa forma, diferentes usuários podem receber diferentes conjuntos de permissões.

Como configurar seu grupo de identidades para um IdP SAML

As etapas a seguir descrevem como configurar o grupo de identidades para usar um IdP baseado em SAML.

> **Note**
>
> Antes de configurar o grupo de identidades para ser compatível com um provedor SAML, primeiro configure o IdP SAML no console do IAM. Para obter mais informações, consulte Integrar provedores de soluções SAML de terceiros com a AWS no Guia do usuário do IAM.

Como adicionar um provedor de identidades (IdP) SAML

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Selecione Adicionar provedor de identidade.
4. Escolha SAML.
5. Escolha um provedor de identidade SAML do IAM IdPs em seu Conta da AWS. Se você quiser adicionar um novo provedor SAML, selecione Criar provedor para navegar até o console do IAM.
6. Para alterar o perfil que o Amazon Cognito solicita ao emitir credenciais para usuários que se autenticaram com esse provedor, defina Configurações de perfil.

- Você pode atribuir aos usuários desse IdP o Perfil padrão que você configurou ao definir seu Perfil autenticado ou Escolher perfil com regras.

    i. Se você escolheu Escolher perfil com regras, insira a Declaração de origem da autenticação do usuário, o Operador pelo qual deseja comparar a declaração, o Valor que gerará uma correspondência com essa opção de perfil e o Perfil que deseja atribuir quando houver correspondência com a Atribuição de função. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

    ii. Selecione uma Resolução de perfil. Quando as declarações do usuário não correspondem às suas regras, você pode negar ou emitir credenciais para seu Perfil autenticado.

7. Para alterar as tags de identidade principal que o Amazon Cognito atribui ao emitir credenciais para usuários que se autenticaram com esse provedor, configure Atributos para controle de acesso.

    a. Para não aplicar nenhuma tag de entidade principal, selecione Inativo.

    b. Para aplicar tags de entidade principal com base em declarações `sub` e `aud`, selecione Usar mapeamentos padrão.

    c. Para criar seu próprio esquema personalizado de atributos para as tags de entidade principal, selecione Usar mapeamentos personalizados. Depois, insira a Chave de tag que você deseja obter de cada declaração e representar em uma tag.

8. Selecione Salvar alterações.

Como configurar seu IdP SAML

Depois de criar o provedor SAML, configure o IdP SAML para adicionar uma confiança de terceira parte confiável entre o IdP e a AWS. Com muitos IdPs, você pode especificar uma URL que o IdP pode usar para ler informações e certificados de terceiros confiáveis de um documento XML. Para AWS, você pode usar o https://signin.aws.amazon.com/static/saml-metadata.xml. A próxima etapa é configurar a resposta da asserção SAML do seu IdP para preencher as declarações necessárias. AWS Para obter detalhes sobre a configuração de reivindicação, consulte Configuração de declarações SAML para a resposta de autenticação.

Quando o IdP SAML inclui mais de um certificado de assinatura nos metadados do SAML, no login, o banco de identidades determina que a declaração SAML é válida se corresponder a qualquer certificado nos metadados do SAML.

Como personalizar a função do usuário com SAML

Ao usar o SAML com a identidade do Amazon Cognito, você pode personalizar a função para o usuário final. O Amazon Cognito só aceita o fluxo avançado com o IdP baseado em SAML. Não é necessário especificar uma função autenticada ou não autenticada para o grupo de identidades para usar um IdP com base em SAML. O atributo de reivindicação `https://aws.amazon.com/SAML/Attributes/Role` especifica um ou mais pares de ARN de provedor e função delimitados por vírgulas. Essas são as funções que o usuário pode assumir. Você pode configurar o IdP SAML para preencher os atributos de função com base nas informações do atributo de usuário disponíveis no IdP. Se você receber várias funções na declaração SAML, preencha o parâmetro `customRoleArn` opcional ao chamar `getCredentialsForIdentity`. O usuário assumirá esse `customRoleArn` se a função corresponder a uma função na declaração SAML.

Como autenticar usuários com um IdP SAML

Para federar com o IdP baseado em SAML, determine a URL em que o usuário inicia o login. AWS a federação usa login iniciado pelo IDP. No AD FS 2.0, o URL tem a forma de `https://<fqdn>/adfs/ls/IdpInitiatedSignOn.aspx?loginToRp=urn:amazon:webservices`.

Para adicionar compatibilidade com IdP SAML no Amazon Cognito, primeiro autentique os usuários com o provedor de identidade SAML do aplicativo iOS ou Android. O código que você usa para integrar e autenticar com o IdP SAML é específico para os provedores SAML. Depois de autenticar seu usuário, você pode usar o Amazon APIs Cognito para fornecer a declaração SAML resultante para o Amazon Cognito Identity.

Você não pode repetir nem reproduzir uma declaração SAML no mapa `Logins` da sua solicitação de API de grupo de identidades. Uma declaração SAML reproduzida tem um ID de declaração que duplica o ID de uma resposta anterior da API. As operações de API que podem aceitar uma declaração SAML no `Logins` mapa incluem GetId, GetCredentialsForIdentityGetOpenIdToken, e. GetOpenIDTokenForDeveloperIdentity Você pode reproduzir um ID de declaração SAML uma vez por solicitação de API em um fluxo de autenticação de grupo de identidades. Por exemplo, você pode fornecer a mesma declaração SAML em uma solicitação `GetId` e em uma solicitação `GetCredentialsForIdentity` subsequente, mas não em uma segunda solicitação `GetId`.

Identidades autenticadas pelo desenvolvedor

O Amazon Cognito é compatível com identidades autenticadas pelo desenvolvedor, além da federação de identidades da web por meio de Configurar o Facebook como um IdP de bancos de identidades, Configurar o Google como um IdP do banco de identidades, Configurar o Login with

Amazon como um IdP de bancos de identidades e Configurar o Login com a Apple como um IdP do banco de identidades. Com identidades autenticadas pelo desenvolvedor, você pode registrar e autenticar usuários por meio de seu próprio processo de autenticação existente, sem deixar de usar o Amazon Cognito para sincronizar dados do usuário e acessar recursos. AWS O uso de identidades autenticadas pelo desenvolvedor engloba a interação entre o dispositivo do usuário final, o back-end para autenticação e o Amazon Cognito. Para obter mais detalhes, consulte Entendendo a autenticação do Amazon Cognito, Parte 2: Identidades autenticadas pelo desenvolvedor no blog. AWS

Como entender o fluxo de autenticação

A operação GetOpenIdTokenForDeveloperIdentityda API pode iniciar a autenticação do desenvolvedor para autenticação avançada e básica. Essa API autentica uma solicitação com credenciais administrativas. O mapa `Logins` é um nome de provedor do desenvolvedor do banco de identidades, como `login.mydevprovider`, emparelhado com um identificador personalizado.

Exemplo:

```
"Logins": {
        "login.mydevprovider": "my developer identifier"
    }
```

Autenticação aprimorada

Chame a operação da GetCredentialsForIdentityAPI com um `Logins` mapa com o nome `cognito-identity.amazonaws.com` e o valor do token de`GetOpenIdTokenForDeveloperIdentity`.

Exemplo:

```
"Logins": {
        "cognito-identity.amazonaws.com": "eyJra12345EXAMPLE"
    }
```

`GetCredentialsForIdentity` com identidades autenticadas pelo desenvolvedor retorna credenciais temporárias para a função autenticada padrão do banco de identidades.

Autenticação básica

Chame a operação `RoleArn` da AssumeRoleWithWebIdentityAPI e solicite qualquer função do IAM que tenha uma relação de confiança apropriada definida. Defina o valor de `WebIdentityToken` para o token obtido de `GetOpenIdTokenForDeveloperIdentity`.

Para obter informações sobre o fluxo de autenticação das identidades autenticadas pelo desenvolvedor e como elas diferem das identidades do provedor externo, consulte Fluxo de autenticação dos bancos de identidades.

Defina um nome de provedor do desenvolvedor e associe-o a um grupo de identidades

Para usar identidades autenticadas pelo desenvolvedor, você precisará de um banco de identidades associado ao provedor do desenvolvedor. Para fazer isso, siga estas etapas:

Como adicionar um provedor de desenvolvedor personalizado

1. Selecione Bancos de identidades no console do Amazon Cognito. Selecione um banco de identidades.
2. Selecione a guia Acesso do usuário.
3. Selecione Adicionar provedor de identidade.
4. Escolha Provedor de desenvolvedor personalizado.
5. Insira um Nome de provedor de desenvolvedor. Você não poderá alterar nem excluir o provedor de desenvolvedor depois de adicioná-lo.
6. Selecione Salvar alterações.

Observação: depois que o nome do provedor for definido, ele não poderá ser alterado.

Implementar um provedor de identidade

Android

Para usar as identidades autenticadas pelo desenvolvedor, implemente sua própria classe de provedor de identidades, que estende `AWSAbstractCognitoIdentityProvider`. A classe de provedor de identidade deve retornar um objeto de resposta contendo o token como um atributo.

Veja a seguir um exemplo básico de um provedor de identidades.

```
public class DeveloperAuthenticationProvider extends
 AWSAbstractCognitoDeveloperIdentityProvider {

  private static final String developerProvider = "<Developer_provider_name>";
```

```
 public DeveloperAuthenticationProvider(String accountId, String identityPoolId,
Regions region) {
   super(accountId, identityPoolId, region);
   // Initialize any other objects needed here.
 }

 // Return the developer provider name which you choose while setting up the
 // identity pool in the &COG; Console

 @Override
 public String getProviderName() {
   return developerProvider;
 }

 // Use the refresh method to communicate with your backend to get an
 // identityId and token.

 @Override
 public String refresh() {

   // Override the existing token
   setToken(null);

   // Get the identityId and token by making a call to your backend
   // (Call to your backend)

   // Call the update method with updated identityId and token to make sure
   // these are ready to be used from Credentials Provider.

   update(identityId, token);
   return token;

 }

 // If the app has a valid identityId return it, otherwise get a valid
 // identityId from your backend.

 @Override
 public String getIdentityId() {

   // Load the identityId from the cache
   identityId = cachedIdentityId;

   if (identityId == null) {
```

```
        // Call to your backend
    } else {
        return identityId;
    }

  }
}
```

Para usar esse provedor de identidade, você precisa inseri-lo em `CognitoCachingCredentialsProvider`. Veja um exemplo abaixo:

```
DeveloperAuthenticationProvider developerProvider = new
 DeveloperAuthenticationProvider( null, "IDENTITYPOOLID", context, Regions.USEAST1);
CognitoCachingCredentialsProvider credentialsProvider = new
 CognitoCachingCredentialsProvider( context, developerProvider, Regions.USEAST1);
```

iOS - objective-C

Para usar as identidades autenticadas pelo desenvolvedor, implemente sua própria classe de provedor de identidades, que estende AWSCognitoCredentialsProviderHelper. A classe de provedor de identidade deve retornar um objeto de resposta contendo o token como um atributo.

```
@implementation DeveloperAuthenticatedIdentityProvider
/*
 * Use the token method to communicate with your backend to get an
 * identityId and token.
 */

- (AWSTask <NSString*> *) token {
    //Write code to call your backend:
    //Pass username/password to backend or some sort of token to authenticate user
    //If successful, from backend call getOpenIdTokenForDeveloperIdentity with logins
 map
    //containing "your.provider.name":"enduser.username"
    //Return the identity id and token to client
    //You can use AWSTaskCompletionSource to do this asynchronously

    // Set the identity id and return the token
    self.identityId = response.identityId;
    return [AWSTask taskWithResult:response.token];
}
```

```
@end
```

Para usar esse provedor de identidade, insira-o em `AWSCognitoCredentialsProvider`, conforme mostrado no exemplo a seguir:

```
DeveloperAuthenticatedIdentityProvider * devAuth =
 [[DeveloperAuthenticatedIdentityProvider alloc]
 initWithRegionType:AWSRegionYOUR_IDENTITY_POOL_REGION
                                          identityPoolId:@"YOUR_IDENTITY_POOL_ID"
                                         useEnhancedFlow:YES
                                identityProviderManager:nil];
AWSCognitoCredentialsProvider *credentialsProvider = [[AWSCognitoCredentialsProvider
 alloc]

 initWithRegionType:AWSRegionYOUR_IDENTITY_POOL_REGION
                                                        identityProvider:devAuth];
```

Para oferecer compatibilidade com identidades não autenticadas e identidades autenticadas pelo desenvolvedor, substitua o método `logins` na implementação de `AWSCognitoCredentialsProviderHelper`.

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
    if(/*logic to determine if user is unauthenticated*/) {
        return [AWSTask taskWithResult:nil];
    }else{
        return [super logins];
    }
}
```

Para oferecer compatibilidade com identidades autenticadas pelo desenvolvedor e provedores de redes sociais, gerencie o provedor atual da implementação `logins` de `AWSCognitoCredentialsProviderHelper`.

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
    if(/*logic to determine if user is unauthenticated*/) {
        return [AWSTask taskWithResult:nil];
    }else if (/*logic to determine if user is Facebook*/){
        return [AWSTask taskWithResult: @{ AWSIdentityProviderFacebook :
 [FBSDKAccessToken currentAccessToken] }];
    }else {
        return [super logins];
```

```
    }
}
```

iOS - swift

Para usar as identidades autenticadas pelo desenvolvedor, implemente sua própria classe de provedor de identidades, que estende AWSCognitoCredentialsProviderHelper. A classe de provedor de identidade deve retornar um objeto de resposta contendo o token como um atributo.

```
import AWSCore
/*
 * Use the token method to communicate with your backend to get an
 * identityId and token.
 */
class DeveloperAuthenticatedIdentityProvider : AWSCognitoCredentialsProviderHelper {
    override func token() -> AWSTask<NSString> {
    //Write code to call your backend:
    //pass username/password to backend or some sort of token to authenticate user, if
 successful,
    //from backend call getOpenIdTokenForDeveloperIdentity with logins map containing
 "your.provider.name":"enduser.username"
    //return the identity id and token to client
    //You can use AWSTaskCompletionSource to do this asynchronously

    // Set the identity id and return the token
    self.identityId = resultFromAbove.identityId
    return AWSTask(result: resultFromAbove.token)
}
```

Para usar esse provedor de identidade, insira-o em `AWSCognitoCredentialsProvider`, conforme mostrado no exemplo a seguir:

```
let devAuth =
 DeveloperAuthenticatedIdentityProvider(regionType: .YOUR_IDENTITY_POOL_REGION,
 identityPoolId: "YOUR_IDENTITY_POOL_ID", useEnhancedFlow: true,
 identityProviderManager:nil)
let credentialsProvider =
 AWSCognitoCredentialsProvider(regionType: .YOUR_IDENTITY_POOL_REGION,
 identityProvider:devAuth)
let configuration = AWSServiceConfiguration(region: .YOUR_IDENTITY_POOL_REGION,
 credentialsProvider:credentialsProvider)
AWSServiceManager.default().defaultServiceConfiguration = configuration
```

Para oferecer compatibilidade com identidades não autenticadas e identidades autenticadas pelo desenvolvedor, substitua o método `logins` na implementação de `AWSCognitoCredentialsProviderHelper`.

```
override func logins () -> AWSTask<NSDictionary> {
    if(/*logic to determine if user is unauthenticated*/) {
        return AWSTask(result:nil)
    }else {
        return super.logins()
    }
}
```

Para oferecer compatibilidade com identidades autenticadas pelo desenvolvedor e provedores de redes sociais, gerencie o provedor atual da implementação `logins` de `AWSCognitoCredentialsProviderHelper`.

```
override func logins () -> AWSTask<NSDictionary> {
    if(/*logic to determine if user is unauthenticated*/) {
        return AWSTask(result:nil)
    }else if (/*logic to determine if user is Facebook*/){
        if let token = AccessToken.current?.authenticationToken {
            return AWSTask(result: [AWSIdentityProviderFacebook:token])
        }
        return AWSTask(error:NSError(domain: "Facebook Login", code: -1 , userInfo:
 ["Facebook" : "No current Facebook access token"]))
    }else {
        return super.logins()
    }
}
```

JavaScript

Depois de obter um ID de identidade e um token de sessão no back-end, você deve inseri-los no provedor `AWS.CognitoIdentityCredentials`. Aqui está um exemplo.

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
   IdentityPoolId: 'IDENTITY_POOL_ID',
   IdentityId: 'IDENTITY_ID_RETURNED_FROM_YOUR_PROVIDER',
   Logins: {
      'cognito-identity.amazonaws.com': 'TOKEN_RETURNED_FROM_YOUR_PROVIDER'
   }
```

```
});
```

Unity

Para usar identidades autenticadas pelo desenvolvedor, é necessário estender `CognitoAWSCredentials` e substituir o método `RefreshIdentity` para recuperar o ID de identidade e o token do usuário no back-end e retorná-los. Veja a seguir um exemplo simples de um provedor de identidades que entrará em contato com um back-end hipotético em “example.com”:

```
using UnityEngine;
using System.Collections;
using Amazon.CognitoIdentity;
using System.Collections.Generic;
using ThirdParty.Json.LitJson;
using System;
using System.Threading;

public class DeveloperAuthenticatedCredentials : CognitoAWSCredentials
{
    const string PROVIDER_NAME = "example.com";
    const string IDENTITY_POOL = "IDENTITY_POOL_ID";
    static readonly RegionEndpoint REGION = RegionEndpoint.USEast1;

    private string login = null;

    public DeveloperAuthenticatedCredentials(string loginAlias)
        : base(IDENTITY_POOL, REGION)
    {
        login = loginAlias;
    }

    protected override IdentityState RefreshIdentity()
    {
        IdentityState state = null;
        ManualResetEvent waitLock = new ManualResetEvent(false);
        MainThreadDispatcher.ExecuteCoroutineOnMainThread(ContactProvider((s) =>
        {
            state = s;
            waitLock.Set();
        }));
        waitLock.WaitOne();
        return state;
    }
```

```
    IEnumerator ContactProvider(Action<IdentityState> callback)
    {
        WWW www = new WWW("http://example.com/?username="+login);
        yield return www;
        string response = www.text;

        JsonData json = JsonMapper.ToObject(response);

        //The backend has to send us back an Identity and a OpenID token
        string identityId = json["IdentityId"].ToString();
        string token = json["Token"].ToString();

        IdentityState state = new IdentityState(identityId, PROVIDER_NAME, token,
 false);
        callback(state);
    }
}
```

O código acima usa um objeto dispatcher de thread para chamar uma corrotina. Se você não tem uma maneira de fazer isso no seu projeto, use o seguinte script em suas cenas:

```
using System;
using UnityEngine;
using System.Collections;
using System.Collections.Generic;

public class MainThreadDispatcher : MonoBehaviour
{
    static Queue<IEnumerator> _coroutineQueue = new Queue<IEnumerator>();
    static object _lock = new object();

    public void Update()
    {
        while (_coroutineQueue.Count > 0)
        {
            StartCoroutine(_coroutineQueue.Dequeue());
        }
    }

    public static void ExecuteCoroutineOnMainThread(IEnumerator coroutine)
    {
        lock (_lock) {
```

```
            _coroutineQueue.Enqueue(coroutine);
        }
    }
}
```

Xamarin

Para usar identidades autenticadas pelo desenvolvedor, é necessário estender `CognitoAWSCredentials` e substituir o método `RefreshIdentity` para recuperar o ID de identidade e o token do usuário no back-end e retorná-los. Veja a seguir um exemplo básico de um provedor de identidades que entrará em contato com um back-end hipotético em “example.com”:

```
public class DeveloperAuthenticatedCredentials : CognitoAWSCredentials
{
    const string PROVIDER_NAME = "example.com";
    const string IDENTITY_POOL = "IDENTITY_POOL_ID";
    static readonly RegionEndpoint REGION = RegionEndpoint.USEast1;
    private string login = null;

    public DeveloperAuthenticatedCredentials(string loginAlias)
        : base(IDENTITY_POOL, REGION)
    {
        login = loginAlias;
    }

    protected override async Task<IdentityState> RefreshIdentityAsync()
    {
        IdentityState state = null;
        //get your identity and set the state
        return state;
    }
}
```

Como atualizar o mapa de logins (apenas Android e iOS)

Android

Depois de autenticar o usuário com êxito por meio do sistema de autenticação, atualize o mapa de logins com o nome do provedor do desenvolvedor e um identificador de usuário do desenvolvedor. Essa é uma sequência alfanumérica que identifica exclusivamente um usuário em seu sistema de

autenticação. Não deixe de chamar o método `refresh` após a atualização do mapa de logins, pois `identityId` pode ter sido alterado:

```
HashMap<String, String> loginsMap = new HashMap<String, String>();
loginsMap.put(developerAuthenticationProvider.getProviderName(),
 developerUserIdentifier);

credentialsProvider.setLogins(loginsMap);
credentialsProvider.refresh();
```

iOS - objective-C

O iOS SDK chama o método `logins` apenas para obter o mapa de logins mais recente, caso não haja credenciais ou elas tenham expirado. Se você quiser forçar o SDK a obter novas credenciais (por exemplo, se o usuário final tiver passado de não autenticado para autenticado e você precisar das credenciais com base no usuário autenticado), chame `clearCredentials` no `credentialsProvider`.

```
[credentialsProvider clearCredentials];
```

iOS - swift

O iOS SDK chama o método `logins` apenas para obter o mapa de logins mais recente, caso não haja credenciais ou elas tenham expirado. Se você quiser forçar o SDK a obter novas credenciais (por exemplo, se o usuário final era não autenticado e se tornou autenticado, e você precisar das credenciais com base no usuário autenticado), chame `clearCredentials` no `credentialsProvider`.

```
credentialsProvider.clearCredentials()
```

Como obter um token (lado do servidor)

Você obtém um token ligando GetOpenIdTokenForDeveloperIdentity. Essa API deve ser invocada do seu back-end usando as credenciais do AWS desenvolvedor. Ele não deve ser invocada no SDK do cliente. A API recebe o ID do banco de identidades do Cognito; um mapa de logins contendo o nome do provedor de identidades como chave e o identificador como valor; e, opcionalmente, o ID de identidade do Cognito (por exemplo, você está autenticando um usuário não autenticado). O identificador pode ser o nome de usuário do seu usuário, um endereço de e-mail ou um valor

numérico. A API responde à chamada com um ID exclusivo do Cognito para o usuário e um token do OpenID Connect para o usuário final.

Tenha em mente as seguintes informações sobre o token retornado por `GetOpenIdTokenForDeveloperIdentity`:

- Você pode especificar um período de expiração personalizado para o token, a fim de que possa armazená-lo em cache. Se você não fornecer o período de expiração personalizado, o token ficará válido por 15 minutos.
- A duração máxima do token que você pode definir é 24 horas.
- Tenha em mente as implicações de segurança relacionadas ao aumento da duração do token. Se um invasor obtiver esse token, ele poderá trocá-lo por AWS credenciais para o usuário final durante a duração do token.

O trecho Java a seguir mostra como inicializar um cliente do Amazon Cognito e recuperar um token para uma identidade autenticada pelo desenvolvedor.

```
// authenticate your end user as appropriate
// ....

// if authenticated, initialize a cognito client with your AWS developer credentials
AmazonCognitoIdentity identityClient = new AmazonCognitoIdentityClient(
  new BasicAWSCredentials("access_key_id", "secret_access_key")
);

// create a new request to retrieve the token for your end user
GetOpenIdTokenForDeveloperIdentityRequest request =
  new GetOpenIdTokenForDeveloperIdentityRequest();
request.setIdentityPoolId("YOUR_COGNITO_IDENTITY_POOL_ID");

request.setIdentityId("YOUR_COGNITO_IDENTITY_ID"); //optional, set this if your client
 has an
                                                    //identity ID that you want to link
 to this
                                                    //developer account

// set up your logins map with the username of your end user
HashMap<String,String> logins = new HashMap<>();
logins.put("YOUR_IDENTITY_PROVIDER_NAME","YOUR_END_USER_IDENTIFIER");
request.setLogins(logins);
```

```
// optionally set token duration (in seconds)
request.setTokenDuration(60 * 15l);
GetOpenIdTokenForDeveloperIdentityResult response =
  identityClient.getOpenIdTokenForDeveloperIdentity(request);

// obtain identity id and token to return to your client
String identityId = response.getIdentityId();
String token = response.getToken();

//code to return identity id and token to client
//...
```

Após as etapas acima, você conseguirá integrar as identidades autenticadas pelo desenvolvedor em sua aplicação. Se você tiver problemas ou dúvidas, fique à vontade para publicar em nossos fóruns.

Conectar-se a uma identidade social existente

Todos os vínculos de provedores durante o uso de identidades autenticadas pelo desenvolvedor devem ser feitos no back-end. Para conectar uma identidade personalizada à identidade social de um usuário (Login com Amazon, Faça login com Apple, Facebook ou Google), adicione o token do provedor de identidade ao mapa de logins ao ligar GetOpenIdTokenForDeveloperIdentity. Para que isso seja possível, ao chamar o backend no SDK do cliente para autenticar o usuário final, insira também o token do provedor de redes sociais do usuário final.

Por exemplo, se você estiver tentando vincular uma identidade personalizada ao Facebook, adicione o token do Facebook, além do identificador do provedor de identidade, ao mapa de logins quando chamar `GetOpenIdTokenForDeveloperIdentity`.

```
logins.put("YOUR_IDENTITY_PROVIDER_NAME","YOUR_END_USER_IDENTIFIER");
logins.put("graph.facebook.com","END_USERS_FACEBOOK_ACCESSTOKEN");
```

Dar suporte à transição entre provedores

Android

Talvez sua aplicação exija compatibilidade com identidades autenticadas ou não autenticadas por meio de provedores públicos (Login with Amazon, Sign in with Apple, Facebook ou Google), bem como as identidades autenticadas pelo desenvolvedor. A principal diferença entre as identidades autenticadas pelo desenvolvedor e outras identidades (identidades autenticadas e não autenticadas

por meio do provedor público) é a maneira como o ID de identidade e o token são obtidos. Para outras identidades, o aplicativo móvel vai interagir diretamente com o Amazon Cognito, em vez de entrar em contato com o sistema de autenticação. Portanto, o aplicativo para dispositivos móveis deve ser capaz de oferecer suporte a dois fluxos distintos, dependendo da opção feita pelo usuário do aplicativo. Para isso, você precisará fazer algumas alterações no provedor de identidades personalizado.

O método `refresh` confere o mapa de logins. Se o mapa não estiver vazio e tiver uma chave com o nome do provedor do desenvolvedor, chame o back-end. Caso contrário, chame o getIdentityId método e retorne null.

```
public String refresh() {

   setToken(null);

   // If the logins map is not empty make a call to your backend
   // to get the token and identityId
   if (getProviderName() != null &&
      !this.loginsMap.isEmpty() &&
      this.loginsMap.containsKey(getProviderName())) {

      /**
       * This is where you would call your backend
       **/

      // now set the returned identity id and token in the provider
      update(identityId, token);
      return token;

   } else {
      // Call getIdentityId method and return null
      this.getIdentityId();
      return null;
   }
}
```

Da mesma forma, o método `getIdentityId` terá dois fluxos de acordo com o conteúdo do mapa de logins:

```
public String getIdentityId() {

   // Load the identityId from the cache
```

```
    identityId = cachedIdentityId;

    if (identityId == null) {

       // If the logins map is not empty make a call to your backend
       // to get the token and identityId

       if (getProviderName() != null && !this.loginsMap.isEmpty()
          && this.loginsMap.containsKey(getProviderName())) {

          /**
            * This is where you would call your backend
           **/

          // now set the returned identity id and token in the provider
          update(identityId, token);
          return token;

       } else {
          // Otherwise call &COG; using getIdentityId of super class
          return super.getIdentityId();
       }

    } else {
       return identityId;
    }

}
```

iOS - objective-C

Talvez sua aplicação exija compatibilidade com identidades autenticadas ou não autenticadas por meio de provedores públicos (Login with Amazon, Sign in with Apple, Facebook ou Google), bem como as identidades autenticadas pelo desenvolvedor. Para fazer isso, substitua o AWSCognitoCredentialsProviderHelper`logins`método para poder retornar o mapa de logins correto com base no provedor de identidade atual. Este exemplo mostra como alternar entre identidade não autenticada, Facebook e identidade autenticada pelo desenvolvedor.

```
- (AWSTask<NSDictionary<NSString *, NSString *> *> *)logins {
    if(/*logic to determine if user is unauthenticated*/) {
        return [AWSTask taskWithResult:nil];
    }else if (/*logic to determine if user is Facebook*/){
```

```
        return [AWSTask taskWithResult: @{ AWSIdentityProviderFacebook :
 [FBSDKAccessToken currentAccessToken] }];
    }else {
        return [super logins];
    }
}
```

Ao fazer a transição de não autenticada para autenticada, você deverá chamar `[credentialsProvider clearCredentials];` para forçar o SDK a obter novas credenciais autenticadas. Quando você alternar entre dois provedores autenticados e estiver tentando vincular os dois provedores (por exemplo, se não estiver fornecendo tokens a vários provedores no dicionário de logins), chame `[credentialsProvider clearKeychain];`. Isso limpará as credenciais e a identidade, e forçará o SDK a obter novas.

iOS - swift

Talvez sua aplicação exija compatibilidade com identidades autenticadas ou não autenticadas por meio de provedores públicos (Login with Amazon, Sign in with Apple, Facebook ou Google), bem como as identidades autenticadas pelo desenvolvedor. Para fazer isso, substitua o AWSCognitoCredentialsProviderHelper`logins`método para poder retornar o mapa de logins correto com base no provedor de identidade atual. Este exemplo mostra como alternar entre identidade não autenticada, Facebook e identidade autenticada pelo desenvolvedor.

```
override func logins () -> AWSTask<NSDictionary> {
    if(/*logic to determine if user is unauthenticated*/) {
        return AWSTask(result:nil)
    }else if (/*logic to determine if user is Facebook*/){
        if let token = AccessToken.current?.authenticationToken {
            return AWSTask(result: [AWSIdentityProviderFacebook:token])
        }
        return AWSTask(error:NSError(domain: "Facebook Login", code: -1 , userInfo:
 ["Facebook" : "No current Facebook access token"]))
    }else {
        return super.logins()
    }
}
```

Ao fazer a transição de não autenticada para autenticada, você deverá chamar `credentialsProvider.clearCredentials()` para forçar o SDK a obter novas credenciais autenticadas. Quando você alternar entre dois provedores autenticados e estiver tentando vincular

os dois provedores (ou seja, se você não estiver fornecendo tokens para vários provedores no dicionário de logins), chame `credentialsProvider.clearKeychain()`. Isso limpará as credenciais e a identidade, e forçará o SDK a obter novas.

Unity

Talvez sua aplicação exija compatibilidade com identidades autenticadas ou não autenticadas por meio de provedores públicos (Login with Amazon, Sign in with Apple, Facebook ou Google), bem como as identidades autenticadas pelo desenvolvedor. A principal diferença entre as identidades autenticadas pelo desenvolvedor e outras identidades (identidades autenticadas e não autenticadas por meio do provedor público) é a maneira como o ID de identidade e o token são obtidos. Para outras identidades, o aplicativo móvel vai interagir diretamente com o Amazon Cognito, em vez de entrar em contato com o sistema de autenticação. Portanto, o aplicativo móvel deve ser compatível com dois fluxos distintos, dependendo da opção feita pelo respectivo usuário. Para isso, você precisará fazer algumas alterações no provedor de identidade personalizado.

A maneira recomendada de fazer isso no Unity é estender seu provedor de identidade de AmazonCognitoEnhancedIdentityProvide em vez de e chamar o RefreshAsync método pai em vez do seu AbstractCognitoIdentityProvider, caso o usuário não esteja autenticado com seu próprio back-end. Se o usuário estiver autenticado, use o mesmo fluxo descrito antes.

Xamarin

Talvez sua aplicação exija compatibilidade com identidades autenticadas ou não autenticadas por meio de provedores públicos (Login with Amazon, Sign in with Apple, Facebook ou Google), bem como as identidades autenticadas pelo desenvolvedor. A principal diferença entre as identidades autenticadas pelo desenvolvedor e outras identidades (identidades autenticadas e não autenticadas por meio do provedor público) é a maneira como o ID de identidade e o token são obtidos. Para outras identidades, o aplicativo móvel vai interagir diretamente com o Amazon Cognito, em vez de entrar em contato com o sistema de autenticação. Portanto, o aplicativo móvel deve ser compatível com dois fluxos distintos, dependendo da opção feita pelo respectivo usuário. Para isso, você precisará fazer algumas alterações no provedor de identidades personalizado.

Alternar usuários não autenticados para usuários autenticados

Os grupos de identidade do Amazon Cognito são compatíveis com usuários autenticados e não autenticados. Usuários não autenticados recebem acesso aos seus AWS recursos mesmo que não estejam conectados com nenhum dos seus provedores de identidade (). IdPs Esse nível de

acesso é útil para exibir conteúdo para os usuários antes que eles de façam login. Cada usuário não autenticado tem uma identidade exclusiva no grupo de identidades, embora não tenha feito login e sido autenticado individualmente.

Esta seção descreve o caso em que o usuário escolhe mudar de fazer login com uma identidade não autenticada para usar uma identidade autenticada.

Android

Os usuários podem fazer login em seu aplicativo como convidados não autenticados. Eventualmente, eles podem decidir fazer login usando um dos compatíveis IdPs. O Amazon Cognito garante que uma identidade antiga mantenha o mesmo identificador exclusivo da nova e que os dados do perfil sejam mesclados automaticamente.

Seu aplicativo é informado sobre uma mesclagem de perfil por meio da interface `IdentityChangedListener`. Implemente o método `identityChanged` na interface para receber estas mensagens:

```
@override
public void identityChanged(String oldIdentityId, String newIdentityId) {
    // handle the change
}
```

iOS - objective-C

Os usuários podem fazer login em seu aplicativo como convidados não autenticados. Eventualmente, eles podem decidir fazer login usando um dos compatíveis IdPs. O Amazon Cognito garante que uma identidade antiga mantenha o mesmo identificador exclusivo da nova e que os dados do perfil sejam mesclados automaticamente.

`NSNotificationCenter` informa seu aplicativo sobre uma mesclagem de perfil:

```
[[NSNotificationCenter defaultCenter] addObserver:self
                                      selector:@selector(identityIdDidChange:)
                                      name:AWSCognitoIdentityIdChangedNotification
                                      object:nil];

-(void)identityDidChange:(NSNotification*)notification {
    NSDictionary *userInfo = notification.userInfo;
```

```
    NSLog(@"identity changed from %@ to %@",
        [userInfo objectForKey:AWSCognitoNotificationPreviousId],
        [userInfo objectForKey:AWSCognitoNotificationNewId]);
}
```

iOS - swift

Os usuários podem fazer login em seu aplicativo como convidados não autenticados. Eventualmente, eles podem decidir fazer login usando um dos compatíveis IdPs. O Amazon Cognito garante que uma identidade antiga mantenha o mesmo identificador exclusivo da nova e que os dados do perfil sejam mesclados automaticamente.

`NSNotificationCenter` informa seu aplicativo sobre uma mesclagem de perfil:

```
[NSNotificationCenter.defaultCenter().addObserver(observer: self
   selector:"identityDidChange"
   name:AWSCognitoIdentityIdChangedNotification
   object:nil)

func identityDidChange(notification: NSNotification!) {
  if let userInfo = notification.userInfo as? [String: AnyObject] {
    print("identity changed from: \(userInfo[AWSCognitoNotificationPreviousId])
    to: \(userInfo[AWSCognitoNotificationNewId])")
  }
}
```

JavaScript

Usuário inicialmente não autenticado

Os usuários geralmente começam com a função não autenticada. Para essa função, você define a propriedade de credenciais de seu objeto de configuração sem uma propriedade de logins. Neste caso, sua configuração padrão pode parecer com o seguinte:

```

// set the default config object
var creds = new AWS.CognitoIdentityCredentials({
    IdentityPoolId: 'us-east-1:1699ebc0-7900-4099-b910-2df94f52a030'
});
AWS.config.credentials = creds;
```

Alternar para usuário autenticado

Quando um usuário autenticado se conecta a um IdP e você tem um token, você pode mudar o usuário de não autenticado para autenticado chamando uma função personalizada que atualiza o objeto de credenciais e adiciona o token de logins:

```
// Called when an identity provider has a token for a logged in user
function userLoggedIn(providerName, token) {
    creds.params.Logins = creds.params.Logins || {};
    creds.params.Logins[providerName] = token;

    // Expire credentials to refresh them on the next request
    creds.expired = true;
}
```

Você também pode criar um objeto `CognitoIdentityCredentials`. Se fizer isso, você deverá redefinir as propriedades das credenciais dos objetos de serviço existentes para refletir as informações de configuração das credenciais atualizadas. Consulte Usar objeto de configuração global.

Para obter mais informações sobre o `CognitoIdentityCredentials` objeto, consulte AWS.CognitoIdentityCredentialsna Referência da AWS SDK para JavaScript API.

Unity

Os usuários podem fazer login em seu aplicativo como convidados não autenticados. Eventualmente, eles podem decidir fazer login usando um dos compatíveis IdPs. O Amazon Cognito garante que uma identidade antiga mantenha o mesmo identificador exclusivo da nova e que os dados do perfil sejam mesclados automaticamente.

Você pode se inscrever no `IdentityChangedEvent` para ser notificado sobre mesclagens de perfil:

```
credentialsProvider.IdentityChangedEvent += delegate(object sender,
 CognitoAWSCredentials.IdentityChangedArgs e)
{
    // handle the change
    Debug.log("Identity changed from " + e.OldIdentityId + " to " + e.NewIdentityId);
};
```

Xamarin

Os usuários podem fazer login em seu aplicativo como convidados não autenticados. Eventualmente, eles podem decidir fazer login usando um dos compatíveis IdPs. O Amazon Cognito garante que uma identidade antiga mantenha o mesmo identificador exclusivo da nova e que os dados do perfil sejam mesclados automaticamente.

```
credentialsProvider.IdentityChangedEvent += delegate(object sender,
 CognitoAWSCredentials.IdentityChangedArgs e){
    // handle the change
    Console.WriteLine("Identity changed from " + e.OldIdentityId + " to " +
 e.NewIdentityId);
};
```

Amazon Cognito Sync

> ⚠ Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Assim como o Amazon Cognito Sync, AWS AppSync é um serviço para sincronizar dados de aplicativos entre dispositivos.
> Ele permite que dados do usuário, como preferências de aplicações ou estado de jogos, sejam sincronizados. Ele também amplia essas capacidades ao permitir que vários usuários sincronizem e colaborem em tempo real com dados compartilhados.

O Amazon Cognito Sync é uma AWS service (Serviço da AWS) biblioteca de clientes que possibilita a sincronização de dados de usuários relacionados a aplicativos em vários dispositivos. O Amazon Cognito pode sincronizar dados de perfil do usuário entre dispositivos móveis e a Web sem precisar usar seu próprio backend. As bibliotecas de cliente armazenam dados em cache localmente para que a aplicação possa ler e gravar dados, independentemente do status de conectividade do dispositivo. Quando o dispositivo estiver online, você poderá sincronizar dados. Se você configurar a sincronização por push, poderá notificar outros dispositivos imediatamente de que uma atualização está disponível.

Para obter informações sobre a disponibilidade de regiões do Amazon Cognito, consulte Disponibilidade de regiões de serviço da AWS.

Para saber mais sobre o Amazon Cognito Sync, consulte os tópicos a seguir.

Tópicos

Conceitos básicos do Amazon Cognito Sync

> ⚠ Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Assim como o Amazon Cognito Sync, AWS AppSync é um serviço para sincronizar dados de aplicativos entre dispositivos.
> Ele permite que dados do usuário, como preferências de aplicações ou estado de jogos, sejam sincronizados. Ele também amplia essas capacidades ao permitir que vários usuários sincronizem e colaborem em tempo real com dados compartilhados.

O Amazon Cognito Sync é um AWS serviço e uma biblioteca de clientes que permitem a sincronização entre dispositivos de dados de usuários relacionados a aplicativos. Você pode usá-lo para sincronizar dados de perfil de usuário entre dispositivos móveis e aplicativos web. As bibliotecas de cliente armazenam dados em cache localmente para que o aplicativo possa ler e gravar dados, independentemente do status de conectividade do dispositivo. Quando o dispositivo está online, você pode sincronizar dados e, se configurar a sincronização por push, pode notificar outros dispositivos imediatamente de que uma atualização está disponível.

Configurar um grupo de identidades no Amazon Cognito

O Amazon Cognito Sync requer um grupo de identidades do Amazon Cognito para fornecer identidades de usuário. Antes de usar a sincronização do Amazon Cognito, é necessário primeiro configurar um banco de identidades. Para criar um grupo de identidades e instalar o SDK, consulte Conceitos básicos dos bancos de identidades do Amazon Cognito.

Armazenar e sincronizar dados

Depois que você tiver configurado o grupo de identidades e instalado o SDK, poderá começar a armazenar e sincronizar dados entre dispositivos. Para obter mais informações, consulte Como sincronizar dados entre clientes.

Como sincronizar dados entre clientes

> ⚠ Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Assim como o Amazon Cognito Sync, AWS AppSync é um serviço para sincronizar dados de aplicativos entre dispositivos.

Ele permite que dados do usuário, como preferências de aplicações ou estado de jogos, sejam sincronizados. Ele também amplia essas capacidades ao permitir que vários usuários sincronizem e colaborem em tempo real com dados compartilhados.

Com o Amazon Cognito, é possível salvar dados de usuários em conjuntos de dados que contêm pares de chave-valor. O Amazon Cognito associa esses dados a uma identidade em seu grupo de identidades para que sua aplicação possa acessá-los entre logins e dispositivos. Para sincronizar esses dados entre o serviço do Amazon Cognito e os dispositivos de um usuário final, invoque o método de sincronização. Cada conjunto de dados pode ter um tamanho máximo de 1 MB. Você pode associar até 20 conjuntos de dados a uma identidade.

O cliente do Amazon Cognito Sync cria um cache local para os dados de identidade. Quando sua aplicação lê e grava chaves, ela se comunica com esse cache local. Essa comunicação garante que todas as alterações feitas no dispositivo sejam imediatamente disponibilizadas no dispositivo, mesmo quando você estiver offline. Quando o método de sincronização é chamado, as alterações do serviço são recebidas no dispositivo, e quaisquer alterações locais são enviadas ao serviço. Nesse momento, as alterações são disponibilizadas para outros dispositivos para sincronização.

Como inicializar o cliente do Amazon Cognito Sync

Para inicializar o cliente do Amazon Cognito Sync, primeiro você precisa criar um provedor de credenciais. O provedor de credenciais adquire AWS credenciais temporárias para possibilitar que seu aplicativo acesse seus recursos. AWS Você também deve importar os arquivos de cabeçalho necessários. Use as seguintes etapas para inicializar o cliente do Amazon Cognito Sync:

Android

1. Crie um provedor de credenciais, seguindo as instruções em Como obter credenciais.
2. Importe o pacote do Amazon Cognito da seguinte forma: `import com.amazonaws.mobileconnectors.cognito.*;`
3. Inicialize o Amazon Cognito Sync. Transmita o contexto da aplicação Android, o ID do grupo de identidades, uma Região da AWS e um provedor de credenciais inicializado do Amazon Cognito da seguinte forma:

```
CognitoSyncManager client = new CognitoSyncManager(
    getApplicationContext(),
    Regions.YOUR_REGION,
```

```
    credentialsProvider);
```

iOS – Objective-C

1. Crie um provedor de credenciais, seguindo as instruções em Como obter credenciais.
2. Importe `AWSCore` e `Cognito` e inicialize o `AWSCognito` da seguinte forma:

```
#import <AWSiOSSDKv2/AWSCore.h>
#import <AWSCognitoSync/Cognito.h>

AWSCognito *syncClient = [AWSCognito defaultCognito];
```

3. Se você estiver usando CocoaPods, `<AWSiOSSDKv2/AWSCore.h>` substitua por`AWSCore.h`. Siga a mesma sintaxe para a importação do Amazon Cognito.

iOS – Swift

1. Crie um provedor de credenciais, seguindo as instruções em Como obter credenciais.
2. Importe e inicialize o `AWSCognito` da seguinte forma:

```
import AWSCognito
let syncClient = AWSCognito.default()!
```

JavaScript

1. Baixe o Amazon Cognito Sync Manager para. JavaScript
2. Inclua a biblioteca do Sync Manager no projeto.
3. Crie um provedor de credenciais, seguindo as instruções em Como obter credenciais.
4. Inicialize o Sync Manager da seguinte forma:

```
var syncManager = new AWS.CognitoSyncManager();
```

Unity

1. Crie uma instância de `CognitoAWSCredentials` seguindo as instruções em Como obter credenciais.
2. Crie uma instância de `CognitoSyncManager`. Transmita o objeto `CognitoAwsCredentials` e um `AmazonCognitoSyncConfig` e inclua pelo menos o conjunto de regiões da seguinte forma:

```
AmazonCognitoSyncConfig clientConfig = new AmazonCognitoSyncConfig { RegionEndpoint =
 REGION };
CognitoSyncManager syncManager = new CognitoSyncManager(credentials, clientConfig);
```

Xamarin

1. Crie uma instância de `CognitoAWSCredentials` seguindo as instruções em Como obter credenciais.
2. Crie uma instância de `CognitoSyncManager`. Transmita o objeto `CognitoAwsCredentials` e um `AmazonCognitoSyncConfig` e inclua pelo menos o conjunto de regiões da seguinte forma:

```
AmazonCognitoSyncConfig clientConfig = new AmazonCognitoSyncConfig { RegionEndpoint =
 REGION };
CognitoSyncManager syncManager = new CognitoSyncManager(credentials, clientConfig);
```

Noções básicas sobre conjuntos de dados

O Amazon Cognito organiza os dados de perfil do usuário em conjuntos de dados. Cada conjunto de dados pode conter até 1 MB de dados na forma de pares de chave-valor. Um conjunto de dados é a entidade mais granular na qual você pode realizar uma operação de sincronização. As operações de leitura e gravação realizadas em um conjunto de dados só afetam o repositório local enquanto o método de sincronização não é invocado. O Amazon Cognito identifica um conjunto de dados por meio de uma string exclusiva. Você pode criar um conjunto de dados ou abrir um existente, como mostrado a seguir.

Android

```
Dataset dataset = client.openOrCreateDataset("datasetname");
```

Para excluir um conjunto de dados, primeiro chame o método para removê-lo do armazenamento local e, depois, chame o método `synchronize` da forma a seguir para excluir o conjunto de dados do Amazon Cognito:

```
dataset.delete();
dataset.synchronize(syncCallback);
```

iOS – Objective-C

```
AWSCognitoDataset *dataset = [syncClient openOrCreateDataset:@"myDataSet"];
```

Para excluir um conjunto de dados, primeiro chame o método para removê-lo do armazenamento local e, depois, chame o método `synchronize` da forma a seguir para excluir o conjunto de dados do Amazon Cognito:

```
[dataset clear];
[dataset synchronize];
```

iOS – Swift

```
let dataset = syncClient.openOrCreateDataset("myDataSet")!
```

Para excluir um conjunto de dados, primeiro chame o método para removê-lo do armazenamento local e, depois, chame o método `synchronize` da forma a seguir para excluir o conjunto de dados do Amazon Cognito:

```
dataset.clear()
dataset.synchronize()
```

JavaScript

```
syncManager.openOrCreateDataset('myDatasetName', function(err, dataset) {
   // ...
});
```

Unity

```
string myValue = dataset.Get("myKey");
```

```
dataset.Put("myKey", "newValue");
```

Para excluir uma chave de um conjunto de dados, use `Remove` da seguinte forma:

```
dataset.Remove("myKey");
```

Xamarin

```
Dataset dataset = syncManager.OpenOrCreateDataset("myDatasetName");
```

Para excluir um conjunto de dados, primeiro chame o método para removê-lo do armazenamento local e, depois, chame o método `synchronize` da forma a seguir para excluir o conjunto de dados do Amazon Cognito:

```
dataset.Delete();
dataset.SynchronizeAsync();
```

Leitura e gravação de dados em conjuntos de dados

Os conjuntos de dados do Amazon Cognito funcionam como dicionários, com valores acessíveis por chave. Você pode ler, adicionar ou modificar as chaves e os valores de um conjunto de dados como se ele fosse um dicionário, conforme mostrado nos exemplos a seguir.

Observe que os valores gravados em um conjunto de dados afetam a cópia de dados armazenada em cache local somente enquanto você não chamar o método de sincronização.

Android

```
String value = dataset.get("myKey");
dataset.put("myKey", "my value");
```

iOS – Objective-C

```
[dataset setString:@"my value" forKey:@"myKey"];
NSString *value = [dataset stringForKey:@"myKey"];
```

iOS – Swift

```
dataset.setString("my value", forKey:"myKey")
```

```
let value = dataset.stringForKey("myKey")
```

JavaScript

```
dataset.get('myKey', function(err, value) {
  console.log('myRecord: ' + value);
});

dataset.put('newKey', 'newValue', function(err, record) {
  console.log(record);
});

dataset.remove('oldKey', function(err, record) {
  console.log(success);
});
```

Unity

```
string myValue = dataset.Get("myKey");
dataset.Put("myKey", "newValue");
```

Xamarin

```
//obtain a value
string myValue = dataset.Get("myKey");

// Create a record in a dataset and synchronize with the server
dataset.OnSyncSuccess += SyncSuccessCallback;
dataset.Put("myKey", "myValue");
dataset.SynchronizeAsync();

void SyncSuccessCallback(object sender, SyncSuccessEventArgs e) {
  // Your handler code here
}
```

Android

Para remover chaves de um conjunto de dados, use o método `remove` da seguinte forma:

```
dataset.remove("myKey");
```

iOS – Objective-C

Para excluir uma chave de um conjunto de dados, use `removeObjectForKey` da seguinte forma:

```
[dataset removeObjectForKey:@"myKey"];
```

iOS – Swift

Para excluir uma chave de um conjunto de dados, use `removeObjectForKey` da seguinte forma:

```
dataset.removeObjectForKey("myKey")
```

Unity

Para excluir uma chave de um conjunto de dados, use `Remove` da seguinte forma:

```
dataset.Remove("myKey");
```

Xamarin

Você pode usar `Remove` para excluir uma chave de um conjunto de dados:

```
dataset.Remove("myKey");
```

Como sincronizar dados locais com o armazenamento de sincronização

Android

O método `synchronize` compara os dados armazenados em cache local com os dados armazenados no armazenamento do repositório do Amazon Cognito Sync. As alterações remotas são recebidas do armazenamento do Amazon Cognito Sync. A resolução conflitante será invocada se ocorrer algum conflito e os valores atualizados no dispositivo serão enviados ao serviço. Para sincronizar um conjunto de dados, chame seu método `synchronize`:

```
dataset.synchronize(syncCallback);
```

O método `synchronize` recebe uma implementação da interface `SyncCallback`, assunto abordado a seguir.

O método `synchronizeOnConnectivity()` tenta realizar a sincronização quando a conectividade está disponível. Se a conectividade for disponibilizada imediatamente, `synchronizeOnConnectivity()` se comportará como `synchronize()`. Caso contrário, ele monitorará as alterações de conectividade e executará uma sincronização quando a conectividade for disponibilizada. Se `synchronizeOnConnectivity()`for chamado várias vezes, apenas a última solicitação de sincronização será mantida e apenas o último retorno de chamada será acionado. Se o conjunto de dados ou o retorno de chamada for descartado, esse método não executará uma sincronização e o retorno de chamada não será acionado.

Para saber mais sobre a sincronização de conjunto de dados e os diferentes retornos de chamada, consulte Como manipular retornos de chamada de eventos.

iOS – Objective-C

O método `synchronize` compara os dados armazenados em cache local com os dados armazenados no armazenamento do repositório do Amazon Cognito Sync. As alterações remotas são recebidas do armazenamento do Amazon Cognito Sync. A resolução conflitante será invocada se ocorrer algum conflito e os valores atualizados no dispositivo serão enviados ao serviço. Para sincronizar um conjunto de dados, chame seu método `synchronize`:

O método `synchronize` é assíncrono e retorna um objeto `AWSTask` para manipular a resposta:

```
[[dataset synchronize] continueWithBlock:^id(AWSTask *task) {
    if (task.isCancelled) {
        // Task cancelled.
    } else if (task.error) {
        // Error while executing task.
    } else {
        // Task succeeded. The data was saved in the sync store.
    }
    return nil;
}];
```

O método `synchronizeOnConnectivity` tenta realizar a sincronização quando o dispositivo tem conectividade. Primeiro, `synchronizeOnConnectivity` verifica se há conectividade e, se o dispositivo estiver online, invocará imediatamente synchronize e retornará o objeto `AWSTask` associado à tentativa.

Se o dispositivo estiver offline, `synchronizeOnConnectivity` 1) programará uma sincronização para a próxima vez que o dispositivo ficar online e 2) retornará um `AWSTask` com um resultado de zero. A sincronização programada é válida somente para o ciclo de vida do objeto do conjunto de dados. Os dados não serão sincronizados se o aplicativo for encerrado antes que a conectividade seja recuperada. Se você quiser ser notificado quando os eventos ocorrerem durante a sincronização programada, adicione observadores das notificações encontradas em `AWSCognito`.

Para saber mais sobre a sincronização de conjunto de dados e os diferentes retornos de chamada, consulte Como manipular retornos de chamada de eventos.

iOS – Swift

O método `synchronize` compara os dados armazenados em cache local com os dados armazenados no armazenamento do repositório do Amazon Cognito Sync. As alterações remotas são recebidas do armazenamento do Amazon Cognito Sync. A resolução conflitante será invocada se ocorrer algum conflito e os valores atualizados no dispositivo serão enviados ao serviço. Para sincronizar um conjunto de dados, chame seu método `synchronize`:

O método `synchronize` é assíncrono e retorna um objeto `AWSTask` para manipular a resposta:

```
dataset.synchronize().continueWith(block: { (task) -> AnyObject? in

            if task.isCancelled {
                // Task cancelled.
            } else if task.error != nil {
                // Error while executing task
            } else {
                // Task succeeded. The data was saved in the sync store.
            }
            return task
})
```

O método `synchronizeOnConnectivity` tenta realizar a sincronização quando o dispositivo tem conectividade. Primeiro, `synchronizeOnConnectivity` verifica se há conectividade e, se o dispositivo estiver online, invocará imediatamente `synchronize` e retornará o objeto `AWSTask` associado à tentativa.

Se o dispositivo estiver offline, `synchronizeOnConnectivity` 1) programará uma sincronização para a próxima vez que o dispositivo ficar online e 2) retornará um objeto `AWSTask` com um resultado de zero. A sincronização programada é válida somente para o ciclo de vida do objeto do

conjunto de dados. Os dados não serão sincronizados se o aplicativo for encerrado antes que a conectividade seja recuperada. Se você quiser ser notificado quando os eventos ocorrerem durante a sincronização programada, adicione observadores das notificações encontradas em `AWSCognito`.

Para saber mais sobre a sincronização de conjunto de dados e os diferentes retornos de chamada, consulte Como manipular retornos de chamada de eventos.

JavaScript

O método `synchronize` compara os dados armazenados em cache local com os dados armazenados no armazenamento do repositório do Amazon Cognito Sync. As alterações remotas são recebidas do armazenamento do Amazon Cognito Sync. A resolução conflitante será invocada se ocorrer algum conflito e os valores atualizados no dispositivo serão enviados ao serviço. Para sincronizar um conjunto de dados, chame seu método `synchronize`:

```
dataset.synchronize();
```

Para saber mais sobre a sincronização de conjunto de dados e os diferentes retornos de chamada, consulte Como manipular retornos de chamada de eventos.

Unity

O método synchronize compara os dados armazenados em cache local com os dados armazenados no repositório do Amazon Cognito Sync. As alterações remotas são recebidas do armazenamento do Amazon Cognito Sync. A resolução conflitante será invocada se ocorrer algum conflito e os valores atualizados no dispositivo serão enviados ao serviço. Para sincronizar um conjunto de dados, chame seu método `synchronize`:

```
dataset.Synchronize();
```

A sincronização será executada de forma assíncrona e acabará chamando um dos vários retornos de chamada que você pode especificar no conjunto de dados.

Para saber mais sobre a sincronização de conjunto de dados e os diferentes retornos de chamada, consulte Como manipular retornos de chamada de eventos.

Xamarin

O método `synchronize` compara os dados armazenados em cache local com os dados armazenados no armazenamento do repositório do Amazon Cognito Sync. As alterações remotas

são recebidas do armazenamento do Amazon Cognito Sync. A resolução conflitante será invocada se ocorrer algum conflito e os valores atualizados no dispositivo serão enviados ao serviço. Para sincronizar um conjunto de dados, chame seu método `synchronize`:

```
dataset.SynchronizeAsync();
```

Para saber mais sobre a sincronização de conjunto de dados e os diferentes retornos de chamada, consulte Como manipular retornos de chamada de eventos.

Como manipular retornos de chamada de eventos

> ⚠ Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Assim como o Amazon Cognito Sync, AWS AppSync é um serviço para sincronizar dados de aplicativos entre dispositivos.
> Ele permite que dados do usuário, como preferências de aplicações ou estado de jogos, sejam sincronizados. Ele também amplia essas capacidades ao permitir que vários usuários sincronizem e colaborem em tempo real com dados compartilhados.

Como desenvolvedor do Amazon Cognito Sync, você pode implementar vários retornos de chamada para lidar com diferentes eventos e cenários de sincronização. A interface `SyncCallback` no SDK do Android configura notificações sobre a sincronização do conjunto de dados, incluindo `onSuccess()` quando um conjunto de dados é baixado com sucesso, `onFailure()` quando ocorre uma exceção e `onConflict()` para resolver conflitos entre dados locais e remotos.

No SDK do iOS, você pode se registrar para receber notificações como `AWSCognitoDidStartSynchronizeNotification` e definir manipuladores como o `AWSCognitoRecordConflictHandler` para resolução de conflitos. As JavaScript plataformas Unity e Xamarin têm mecanismos de retorno de chamada análogos. Quando você implementa esses retornos de chamada, a aplicação pode lidar perfeitamente com os vários eventos e cenários de sincronização que podem ocorrer ao usar o Amazon Cognito Sync.

Android

SyncCallback Interface

Ao implementar a interface `SyncCallback`, você poderá receber notificações sobre a sincronização de conjuntos de dados em seu aplicativo. O aplicativo poderá, então, tomar decisões ativas sobre a

exclusão de dados locais, mesclando perfis autenticados e não autenticados e resolvendo conflitos de sincronização. Você deverá implementar os seguintes métodos, que são exigidos pela interface:

- `onSuccess()`
- `onFailure()`
- `onConflict()`
- `onDatasetDeleted()`
- `onDatasetsMerged()`

Observe que, se você não especificar todos os retornos de chamada, também poderá usar a classe `DefaultSyncCallback`, que fornece implementações vazias padrão para todos eles.

onSuccess

O retorno de chamada `onSuccess()` é acionado quando um conjunto de dados é obtido por download no repositório de sincronização.

```
@Override
public void onSuccess(Dataset dataset, List<Record> newRecords) {
}
```

onFailure

onFailure () será chamado se ocorrer uma exceção durante a sincronização.

```
@Override
public void onFailure(DataStorageException dse) {
}
```

onConflict

Podem surgir conflitos se a mesma chave for modificada no repositório local e no repositório de sincronização. O método `onConflict()` se encarrega da resolução de conflitos. Se você não implementar esse método, o cliente do Amazon Cognito Sync adotará como comportamento padrão o uso da alteração mais recente.

```
@Override
public boolean onConflict(Dataset dataset, final List<SyncConflict> conflicts) {
```

```
    List<Record> resolvedRecords = new ArrayList<Record>();
    for (SyncConflict conflict : conflicts) {
        /* resolved by taking remote records */
        resolvedRecords.add(conflict.resolveWithRemoteRecord());

        /* alternately take the local records */
        // resolvedRecords.add(conflict.resolveWithLocalRecord());

        /* or customer logic, say concatenate strings */
        // String newValue = conflict.getRemoteRecord().getValue()
        //     + conflict.getLocalRecord().getValue();
        // resolvedRecords.add(conflict.resolveWithValue(newValue);
    }
    dataset.resolve(resolvedRecords);

    // return true so that synchronize() is retried after conflicts are resolved
    return true;
}
```

onDatasetDeleted

Quando um conjunto de dados é excluído, o cliente do Amazon Cognito usa a interface `SyncCallback` para confirmar se a cópia do conjunto de dados armazenada em cache local também será excluída. Implemente o método `onDatasetDeleted()` para informar ao SDK do cliente o que fazer com os dados locais.

```
@Override
public boolean onDatasetDeleted(Dataset dataset, String datasetName) {
    // return true to delete the local copy of the dataset
    return true;
}
```

onDatasetMerged

Quando duas identidades anteriormente não conectadas são vinculadas, todos os conjuntos de dados serão mesclados. Os aplicativos são notificados da mesclagem por meio do método `onDatasetsMerged()`:

```
@Override
public boolean onDatasetsMerged(Dataset dataset, List<String> datasetNames) {
    // return false to handle Dataset merge outside the synchronization callback
    return false;
```

```
}
```

iOS – Objective-C

Notificações de sincronização

O cliente do Amazon Cognito emitirá diversos eventos de `NSNotification` durante uma chamada de sincronização. Você pode se registrar para monitorar essas notificações por meio do `NSNotificationCenter` padrão:

```
[NSNotificationCenter defaultCenter]
  addObserver:self
  selector:@selector(myNotificationHandler:)
  name:NOTIFICATION_TYPE
  object:nil];
```

O Amazon Cognito é compatível com os cinco tipos de notificação listados a seguir.

AWSCognitoDidStartSynchronizeNotification

Chamado quando uma operação de sincronização está iniciando. O `userInfo` conterá o conjunto de dados de chaves, que equivale ao conjunto de dados que está sendo sincronizado.

AWSCognitoDidEndSynchronizeNotification

Chamado quando uma operação de sincronização é concluída (seja ela bem-sucedida ou não). O `userInfo` conterá o conjunto de dados de chaves, que equivale ao conjunto de dados que está sendo sincronizado.

AWSCognitoDidFailToSynchronizeNotification

Chamado quando uma operação de sincronização apresenta falha. `userInfo` conterá o conjunto de dados de chaves, que equivale ao conjunto de dados que está sendo sincronizado, e o erro de chave, que conterá o erro que ocasionou a falha.

AWSCognitoDidChangeRemoteValueNotification

Chamado quando alterações locais são enviadas com êxito ao Amazon Cognito. Eles `userInfo` conterão o conjunto de dados chave, que é o nome do conjunto de dados que está sendo sincronizado, e as chaves principais, que conterão as chaves NSArray de registro enviadas.

AWSCognitoDidChangeLocalValueFromRemoteNotification

Chamado quando um valor local é alterado devido a uma operação de sincronização. Eles `userInfo` conterão o conjunto de dados chave, que é o nome do conjunto de dados que está sendo sincronizado, e as chaves principais, que conterão uma NSArray das chaves de registro que foram alteradas.

Handler de resolução de conflitos

Durante uma operação de sincronização, poderão surgir conflitos se a mesma chave for modificada no repositório local e no repositório de sincronização. Se você não definir um handler de resolução de conflitos, o Amazon Cognito assumirá como comportamento padrão a atualização mais recente.

Ao implementar e atribuir um, AWSCognito RecordConflictHandler você pode alterar a resolução padrão de conflitos. O AWSCognito conflito do parâmetro de entrada Conflict contém um objeto AWSCognito Record para os dados em cache locais e para o registro conflitante no armazenamento de sincronização. Usando o AWSCognito Conflito, você pode resolver o conflito com o registro local: [resolveWithLocalregistro de conflito], o registro remoto: [registro de conflito resolveWithRemote] ou um valor totalmente novo: [resolveWithValueconflito:valor]. O resultado nil retornado por esse método impede que a sincronização continue, e os conflitos serão apresentados novamente na próxima vez que o processo de sincronização for iniciado.

Você pode definir o handler de resolução de conflitos no nível do cliente:

```
client.conflictHandler = ^AWSCognitoResolvedConflict* (NSString *datasetName,
 AWSCognitoConflict *conflict) {
    // always choose local changes
    return [conflict resolveWithLocalRecord];
};
```

Ou no nível do conjunto de dados:

```
dataset.conflictHandler = ^AWSCognitoResolvedConflict* (NSString *datasetName,
 AWSCognitoConflict *conflict) {
    // override and always choose remote changes
    return [conflict resolveWithRemoteRecord];
};
```

Handler de conjunto de dados excluído

Quando um conjunto de dados é excluído, o cliente do Amazon Cognito usa o `AWSCognitoDatasetDeletedHandler` para confirmar se a cópia do conjunto

de dados armazenada em cache local também será excluída. Se nenhum `AWSCognitoDatasetDeletedHandler` for implementado, os dados locais serão removidos automaticamente. Implemente um `AWSCognitoDatasetDeletedHandler` se quiser manter uma cópia dos dados locais antes da limpeza ou os próprios dados locais.

Você pode definir o handler de conjunto de dados excluído no nível do cliente:

```
client.datasetDeletedHandler = ^BOOL (NSString *datasetName) {
    // make a backup of the data if you choose
    ...
    // delete the local data (default behavior)
    return YES;
};
```

Ou no nível do conjunto de dados:

```
dataset.datasetDeletedHandler = ^BOOL (NSString *datasetName) {
    // override default and keep the local data
    return NO;
};
```

Handler de mesclagem do conjunto de dados

Quando duas identidades anteriormente não conectadas são vinculadas, todos os conjuntos de dados serão mesclados. Os aplicativos são notificados da mesclagem por meio do método `DatasetMergeHandler`. O handler receberá o nome do conjunto de dados raiz, bem como uma matriz de nomes de conjunto de dados que são marcados como mesclagens do conjunto de dados raiz.

Se nenhum `DatasetMergeHandler` for implementado, esses conjuntos de dados serão ignorados, mas continuarão utilizando o espaço dos 20 conjuntos de dados da identidade.

Você pode definir o handler de mesclagem no nível do cliente:

```
client.datasetMergedHandler = ^(NSString *datasetName, NSArray *datasets) {
    // Blindly delete the datasets
    for (NSString *name in datasets) {
        AWSCognitoDataset *merged = [[AWSCognito defaultCognito]
 openOrCreateDataset:name];
       [merged clear];
```

```
        [merged synchronize];
    }
};
```

Ou no nível do conjunto de dados:

```
dataset.datasetMergedHandler = ^(NSString *datasetName, NSArray *datasets) {
    // Blindly delete the datasets
    for (NSString *name in datasets) {
        AWSCognitoDataset *merged = [[AWSCognito defaultCognito]
 openOrCreateDataset:name];
        // do something with the data if it differs from existing dataset
        ...
        // now delete it
        [merged clear];
        [merged synchronize];
    }
};
```

iOS – Swift

Notificações de sincronização

O cliente do Amazon Cognito emitirá diversos eventos de `NSNotification` durante uma chamada de sincronização. Você pode se registrar para monitorar essas notificações por meio do `NSNotificationCenter` padrão:

```
NSNotificationCenter.defaultCenter().addObserver(observer: self,
   selector: "myNotificationHandler",
   name:NOTIFICATION_TYPE,
   object:nil)
```

O Amazon Cognito é compatível com os cinco tipos de notificação listados a seguir.

AWSCognitoDidStartSynchronizeNotification

Chamado quando uma operação de sincronização está iniciando. O `userInfo` conterá o conjunto de dados de chaves, que equivale ao conjunto de dados que está sendo sincronizado.

AWSCognitoDidEndSynchronizeNotification

Chamado quando uma operação de sincronização é concluída (seja ela bem-sucedida ou não). O `userInfo` conterá o conjunto de dados de chaves, que equivale ao conjunto de dados que está sendo sincronizado.

AWSCognitoDidFailToSynchronizeNotification

Chamado quando uma operação de sincronização apresenta falha. `userInfo` conterá o conjunto de dados de chaves, que equivale ao conjunto de dados que está sendo sincronizado, e o erro de chave, que conterá o erro que ocasionou a falha.

AWSCognitoDidChangeRemoteValueNotification

Chamado quando alterações locais são enviadas com êxito ao Amazon Cognito. Eles `userInfo` conterão o conjunto de dados chave, que é o nome do conjunto de dados que está sendo sincronizado, e as chaves principais, que conterão as chaves NSArray de registro enviadas.

AWSCognitoDidChangeLocalValueFromRemoteNotification

Chamado quando um valor local é alterado devido a uma operação de sincronização. Eles `userInfo` conterão o conjunto de dados chave, que é o nome do conjunto de dados que está sendo sincronizado, e as chaves principais, que conterão uma NSArray das chaves de registro que foram alteradas.

Handler de resolução de conflitos

Durante uma operação de sincronização, poderão surgir conflitos se a mesma chave for modificada no repositório local e no repositório de sincronização. Se você não definir um handler de resolução de conflitos, o Amazon Cognito assumirá como comportamento padrão a atualização mais recente.

Ao implementar e atribuir um `AWSCognitoRecordConflictHandler`, você pode alterar a resolução de conflitos padrão. O parâmetro de entrada `AWSCognitoConflict` contém um objeto `AWSCognitoRecord` para os dados armazenados em cache local e para o registro conflitante no repositório de sincronização. Usando o, `AWSCognitoConflict` você pode resolver o conflito com o registro local: [conflict resolveWithLocal Record], o registro remoto: [conflict resolveWithRemote Record] ou um valor totalmente novo: [resolveWithValueconflict:value]. O resultado nil retornado por esse método impede que a sincronização continue, e os conflitos serão apresentados novamente na próxima vez que o processo de sincronização for iniciado.

Você pode definir o handler de resolução de conflitos no nível do cliente:

```
client.conflictHandler = {
    (datasetName: String?, conflict: AWSCognitoConflict?) ->
 AWSCognitoResolvedConflict? in
    return conflict.resolveWithLocalRecord()
}
```

Ou no nível do conjunto de dados:

```
dataset.conflictHandler = {
    (datasetName: String?, conflict: AWSCognitoConflict?) ->
 AWSCognitoResolvedConflict? in
    return conflict.resolveWithLocalRecord()
}
```

Handler de conjunto de dados excluído

Quando um conjunto de dados é excluído, o cliente do Amazon Cognito usa o `AWSCognitoDatasetDeletedHandler` para confirmar se a cópia do conjunto de dados armazenada em cache local também será excluída. Se nenhum `AWSCognitoDatasetDeletedHandler` for implementado, os dados locais serão removidos automaticamente. Implemente um `AWSCognitoDatasetDeletedHandler` se quiser manter uma cópia dos dados locais antes da limpeza ou os próprios dados locais.

Você pode definir o handler de conjunto de dados excluído no nível do cliente:

```
client.datasetDeletedHandler = {
      (datasetName: String!) -> Bool in
      // make a backup of the data if you choose
      ...
      // delete the local data (default behaviour)
      return true
}
```

Ou no nível do conjunto de dados:

```
dataset.datasetDeletedHandler = {
      (datasetName: String!) -> Bool in
      // make a backup of the data if you choose
      ...
      // delete the local data (default behaviour)
```

```
      return true
}
```

Manipulador de mesclagem do conjunto de dados

Quando duas identidades anteriormente não conectadas são vinculadas, todos os conjuntos de dados serão mesclados. Os aplicativos são notificados da mesclagem por meio do método `DatasetMergeHandler`. O handler receberá o nome do conjunto de dados raiz, bem como uma matriz de nomes de conjunto de dados que são marcados como mesclagens do conjunto de dados raiz.

Se nenhum `DatasetMergeHandler` for implementado, esses conjuntos de dados serão ignorados, mas continuarão utilizando o espaço dos 20 conjuntos de dados da identidade.

Você pode definir o handler de mesclagem no nível do cliente:

```
client.datasetMergedHandler = {
    (datasetName: String!, datasets: [AnyObject]!) -> Void in
    for nameObject in datasets {
        if let name = nameObject as? String {
            let merged = AWSCognito.defaultCognito().openOrCreateDataset(name)
            merged.clear()
            merged.synchronize()
        }
    }
}
```

Ou no nível do conjunto de dados:

```
dataset.datasetMergedHandler = {
    (datasetName: String!, datasets: [AnyObject]!) -> Void in
    for nameObject in datasets {
        if let name = nameObject as? String {
            let merged = AWSCognito.defaultCognito().openOrCreateDataset(name)
            // do something with the data if it differs from existing dataset
              ...
              // now delete it
            merged.clear()
            merged.synchronize()
        }
    }
```

```
}
```

JavaScript

Retornos de chamada de sincronização

Ao executar um synchronize() em um conjunto de dados, você poderá especificar retornos de chamada para lidar com cada um dos estados a seguir:

```
dataset.synchronize({

   onSuccess: function(dataset, newRecords) {
      //...
   },

   onFailure: function(err) {
      //...
   },

   onConflict: function(dataset, conflicts, callback) {
      //...
   },

   onDatasetDeleted: function(dataset, datasetName, callback) {
      //...
   },

  onDatasetMerged: function(dataset, datasetNames, callback) {
      //...
  }

});
```

onSuccess()

O retorno de chamada `onSuccess()` é acionado quando um conjunto de dados é atualizado com êxito no repositório de sincronização. Se você não definir um retorno de chamada, a sincronização será concluída com êxito silenciosamente.

```
onSuccess: function(dataset, newRecords) {
   console.log('Successfully synchronized ' + newRecords.length + ' new records.');
}
```

onFailure()

`onFailure()` será chamado se ocorrer uma exceção durante a sincronização. Se você não definir um retorno de chamada, a sincronização apresentará falha silenciosamente.

```
onFailure: function(err) {
   console.log('Synchronization failed.');
   console.log(err);
}
```

onConflict()

Podem surgir conflitos se a mesma chave for modificada no repositório local e no repositório de sincronização. O método `onConflict()` se encarrega da resolução de conflitos. Se você não implementar esse método, a sincronização será anulada quando houver um conflito.

```
onConflict: function(dataset, conflicts, callback) {

   var resolved = [];

   for (var i=0; i<conflicts.length; i++) {

      // Take remote version.
      resolved.push(conflicts[i].resolveWithRemoteRecord());

      // Or... take local version.
      // resolved.push(conflicts[i].resolveWithLocalRecord());

      // Or... use custom logic.
      // var newValue = conflicts[i].getRemoteRecord().getValue() +
 conflicts[i].getLocalRecord().getValue();
      // resolved.push(conflicts[i].resovleWithValue(newValue);

   }

   dataset.resolve(resolved, function() {
      return callback(true);
   });

   // Or... callback false to stop the synchronization process.
   // return callback(false);
```

```
}
```

onDatasetDeleted()

Quando um conjunto de dados é excluído, o cliente do Amazon Cognito usa o retorno de chamada `onDatasetDeleted()` para decidir se a cópia do conjunto de dados armazenada em cache local também será excluída. Por padrão, o conjunto de dados não será excluído.

```
onDatasetDeleted: function(dataset, datasetName, callback) {

   // Return true to delete the local copy of the dataset.
   // Return false to handle deleted datasets outside the synchronization callback.

   return callback(true);

}
```

onDatasetMerged()

Quando duas identidades anteriormente não conectadas são vinculadas, todos os conjuntos de dados serão mesclados. Os aplicativos são notificados da mesclagem por meio do retorno de chamada `onDatasetsMerged()`.

```
onDatasetMerged: function(dataset, datasetNames, callback) {

   // Return true to continue the synchronization process.
   // Return false to handle dataset merges outside the synchronization callback.

   return callback(false);

}
```

Unity

Depois que você abrir ou criar um conjunto de dados, poderá definir diferentes retornos de chamada para ele, que serão acionados quando você usar o método synchronize. Essa é a maneira de registrar os retornos de chamada neles:

```
dataset.OnSyncSuccess += this.HandleSyncSuccess;
dataset.OnSyncFailure += this.HandleSyncFailure;
dataset.OnSyncConflict = this.HandleSyncConflict;
```

```
dataset.OnDatasetMerged = this.HandleDatasetMerged;
dataset.OnDatasetDeleted = this.HandleDatasetDeleted;
```

Observe que `SyncSuccess` e `SyncFailure` usam **+=**, em vez de **=**, para que você possa inscrever mais de um retorno de chamada neles.

OnSyncSuccess

O retorno de chamada `OnSyncSuccess` é acionado quando um conjunto de dados é atualizado com êxito na nuvem. Se você não definir um retorno de chamada, a sincronização será concluída com êxito silenciosamente.

```
private void HandleSyncSuccess(object sender, SyncSuccessEvent e)
{
    // Continue with your game flow, display the loaded data, etc.
}
```

OnSyncFailure

`OnSyncFailure` será chamado se ocorrer uma exceção durante a sincronização. Se você não definir um retorno de chamada, a sincronização apresentará falha silenciosamente.

```
private void HandleSyncFailure(object sender, SyncFailureEvent e)
{
    Dataset dataset = sender as Dataset;
    if (dataset.Metadata != null) {
        Debug.Log("Sync failed for dataset : " + dataset.Metadata.DatasetName);
    } else {
        Debug.Log("Sync failed");
    }
    // Handle the error
    Debug.LogException(e.Exception);
}
```

OnSyncConflict

Podem surgir conflitos se a mesma chave for modificada no repositório local e no repositório de sincronização. O retorno de chamada `OnSyncConflict` se encarrega da resolução de conflitos. Se você não implementar esse método, a sincronização será anulada quando houver um conflito.

```
private bool HandleSyncConflict(Dataset dataset, List < SyncConflict > conflicts)
```

```
{
  if (dataset.Metadata != null) {
    Debug.LogWarning("Sync conflict " + dataset.Metadata.DatasetName);
  } else {
    Debug.LogWarning("Sync conflict");
  }
  List < Amazon.CognitoSync.SyncManager.Record > resolvedRecords = new List <
 Amazon.CognitoSync.SyncManager.Record > ();
  foreach(SyncConflict conflictRecord in conflicts) {
    // SyncManager provides the following default conflict resolution methods:
    //      ResolveWithRemoteRecord - overwrites the local with remote records
    //      ResolveWithLocalRecord - overwrites the remote with local records
    //      ResolveWithValue - to implement your own logic
    resolvedRecords.Add(conflictRecord.ResolveWithRemoteRecord());
  }
  // resolves the conflicts in local storage
  dataset.Resolve(resolvedRecords);
  // on return true the synchronize operation continues where it left,
  //      returning false cancels the synchronize operation
  return true;
}
```

OnDatasetDeleted

Quando um conjunto de dados é excluído, o cliente do Amazon Cognito usa o retorno de chamada `OnDatasetDeleted` para decidir se a cópia do conjunto de dados armazenada em cache local também será excluída. Por padrão, o conjunto de dados não será excluído.

```
private bool HandleDatasetDeleted(Dataset dataset)
  {
      Debug.Log(dataset.Metadata.DatasetName + " Dataset has been deleted");
      // Do clean up if necessary
      // returning true informs the corresponding dataset can be purged in the local
 storage and return false retains the local dataset
      return true;
  }
```

OnDatasetMerged

Quando duas identidades anteriormente não conectadas são vinculadas, todos os conjuntos de dados serão mesclados. Os aplicativos são notificados da mesclagem por meio do retorno de chamada `OnDatasetsMerged`.

```
public bool HandleDatasetMerged(Dataset localDataset, List<string> mergedDatasetNames)
{
    foreach (string name in mergedDatasetNames)
    {
        Dataset mergedDataset = syncManager.OpenOrCreateDataset(name);
        //Lambda function to delete the dataset after fetching it
        EventHandler<SyncSuccessEvent> lambda;
        lambda = (object sender, SyncSuccessEvent e) => {
            ICollection<string> existingValues = localDataset.GetAll().Values;
            ICollection<string> newValues = mergedDataset.GetAll().Values;

            //Implement your merge logic here

            mergedDataset.Delete(); //Delete the dataset locally
            mergedDataset.OnSyncSuccess -= lambda; //We don't want this callback to be
 fired again
            mergedDataset.OnSyncSuccess += (object s2, SyncSuccessEvent e2) => {
                localDataset.Synchronize(); //Continue the sync operation that was
 interrupted by the merge
            };
            mergedDataset.Synchronize(); //Synchronize it as deleted, failing to do so
 will leave us in an inconsistent state
        };
        mergedDataset.OnSyncSuccess += lambda;
        mergedDataset.Synchronize(); //Asnchronously fetch the dataset
    }

    // returning true allows the Synchronize to continue and false stops it
    return false;
}
```

Xamarin

Depois que você abrir ou criar um conjunto de dados, poderá definir diferentes retornos de chamada para ele, que serão acionados quando você usar o método synchronize. Essa é a maneira de registrar os retornos de chamada neles:

```
dataset.OnSyncSuccess += this.HandleSyncSuccess;
dataset.OnSyncFailure += this.HandleSyncFailure;
dataset.OnSyncConflict = this.HandleSyncConflict;
dataset.OnDatasetMerged = this.HandleDatasetMerged;
dataset.OnDatasetDeleted = this.HandleDatasetDeleted;
```

Observe que `SyncSuccess` e `SyncFailure` usam **+=**, em vez de **=**, para que você possa inscrever mais de um retorno de chamada neles.

OnSyncSuccess

O retorno de chamada `OnSyncSuccess` é acionado quando um conjunto de dados é atualizado com êxito na nuvem. Se você não definir um retorno de chamada, a sincronização será concluída com êxito silenciosamente.

```
private void HandleSyncSuccess(object sender, SyncSuccessEventArgs e)
{
    // Continue with your game flow, display the loaded data, etc.
}
```

OnSyncFailure

`OnSyncFailure` será chamado se ocorrer uma exceção durante a sincronização. Se você não definir um retorno de chamada, a sincronização apresentará falha silenciosamente.

```
private void HandleSyncFailure(object sender, SyncFailureEventArgs e)
{
    Dataset dataset = sender as Dataset;
    if (dataset.Metadata != null) {
        Console.WriteLine("Sync failed for dataset : " + dataset.Metadata.DatasetName);
    } else {
        Console.WriteLine("Sync failed");
    }
}
```

OnSyncConflict

Podem surgir conflitos se a mesma chave for modificada no repositório local e no repositório de sincronização. O retorno de chamada `OnSyncConflict` se encarrega da resolução de conflitos. Se você não implementar esse método, a sincronização será anulada quando houver um conflito.

```
private bool HandleSyncConflict(Dataset dataset, List < SyncConflict > conflicts)
{
  if (dataset.Metadata != null) {
    Console.WriteLine("Sync conflict " + dataset.Metadata.DatasetName);
  } else {
```

```
    Console.WriteLine("Sync conflict");
  }
  List < Amazon.CognitoSync.SyncManager.Record > resolvedRecords = new List <
 Amazon.CognitoSync.SyncManager.Record > ();
  foreach(SyncConflict conflictRecord in conflicts) {
    // SyncManager provides the following default conflict resolution methods:
    //      ResolveWithRemoteRecord - overwrites the local with remote records
    //      ResolveWithLocalRecord - overwrites the remote with local records
    //      ResolveWithValue - to implement your own logic
    resolvedRecords.Add(conflictRecord.ResolveWithRemoteRecord());
  }
  // resolves the conflicts in local storage
  dataset.Resolve(resolvedRecords);
  // on return true the synchronize operation continues where it left,
  //      returning false cancels the synchronize operation
  return true;
}
```

OnDatasetDeleted

Quando um conjunto de dados é excluído, o cliente do Amazon Cognito usa o retorno de chamada `OnDatasetDeleted` para decidir se a cópia do conjunto de dados armazenada em cache local também será excluída. Por padrão, o conjunto de dados não será excluído.

```
private bool HandleDatasetDeleted(Dataset dataset)
{
    Console.WriteLine(dataset.Metadata.DatasetName + " Dataset has been deleted");
    // Do clean up if necessary
    // returning true informs the corresponding dataset can be purged in the local
 storage and return false retains the local dataset
    return true;
}
```

OnDatasetMerged

Quando duas identidades anteriormente não conectadas são vinculadas, todos os conjuntos de dados serão mesclados. Os aplicativos são notificados da mesclagem por meio do retorno de chamada `OnDatasetsMerged`.

```
public bool HandleDatasetMerged(Dataset localDataset, List<string> mergedDatasetNames)
{
    foreach (string name in mergedDatasetNames)
```

```
    {
        Dataset mergedDataset = syncManager.OpenOrCreateDataset(name);

            //Implement your merge logic here

        mergedDataset.OnSyncSuccess += lambda;
        mergedDataset.SynchronizeAsync(); //Asnchronously fetch the dataset
    }

    // returning true allows the Synchronize to continue and false stops it
    return false;
}
```

Como implementar a sincronização por push

> Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Assim como o Amazon Cognito Sync, AWS AppSync é um serviço para sincronizar dados de aplicativos entre dispositivos.
> Ele permite que dados do usuário, como preferências de aplicações ou estado de jogos, sejam sincronizados. Ele também amplia essas capacidades ao permitir que vários usuários sincronizem e colaborem em tempo real com dados compartilhados.

O Amazon Cognito rastreia automaticamente a associação entre identidade e dispositivos. Usando o recurso de sincronização por push, você pode garantir que cada instância de uma identidade específica será notificada quando os dados da identidade forem alterados. A sincronização por push garante que, sempre que os dados do repositório de sincronização forem alterados para uma identidade específica, todos os dispositivos associados a essa identidade receberão uma notificação por push silenciosa informando-os da alteração.

> **Note**
>
> A sincronização push não é compatível com Unity ou Xamarin. JavaScript

Para que você possa usar a sincronização por push, primeiro configure a conta da sincronização por push e habilite a sincronização por push no console do Amazon Cognito.

Criar uma aplicação do Amazon Simple Notification Service (Amazon SNS)

Crie e configure uma aplicação do Amazon SNS para suas plataformas compatíveis, conforme descrito no Guia do desenvolvedor do SNS.

Habilitar a sincronização por push no console do Amazon Cognito

Você pode habilitar a sincronização por push por meio do console do Amazon Cognito. Na página inicial do console:

1. Clique no nome do grupo de identidades para o qual deseja habilitar a sincronização por push. A página Dashboard (Painel) do grupo de identidades será exibida.
2. No canto superior direito da página Dashboard (Painel), clique em Manage Identity Pools (Gerenciar grupos de identidades). A página Federated Identities (Identidades federadas) é exibida.
3. Role para baixo e clique em Push synchronization para expandi-lo.
4. No menu suspenso Service role, selecione a função do IAM que concede ao Cognito permissão para enviar uma notificação SNS. Clique em Create role (Criar função) para criar ou modificar as funções associadas ao grupo de identidades no console do AWS IAM.
5. Selecione um aplicativo de plataforma e clique em Save Changes.
6. Concessão de acesso ao SNS ao aplicativo

No AWS Identity and Access Management console, configure suas funções do IAM para ter acesso total ao Amazon SNS ou crie uma nova função que tenha acesso total ao Amazon SNS. A política de confiança da função do exemplo a seguir concede ao Amazon Cognito Sync uma capacidade limitada de assumir uma função do IAM. O Amazon Cognito Sync só pode assumir a função quando fizer isso em nome de ambos o grupo de identidades na condição `aws:SourceArn` e a conta na condição `aws:SourceAccount`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Principal": {
                "Service": "cognito-sync.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "123456789012"
                },
                "ArnLike": {
                    "AWS:SourceArn": "arn:aws:cognito-identity:us-
east-1:123456789012:identitypool/us-east-1:177a950c-2c08-43f0-9983-28727EXAMPLE"
                }
            }
        }
    ]
}
```

Para saber mais sobre as funções do IAM, consulte Funções (delegação e federação).

Usar sincronização por push em sua aplicação: Android

O aplicativo precisará importar o Google Play serviços. Você pode fazer download da versão mais recente do Google Play SDK por meio do Android SDK Manager. Siga a documentação do Android sobre a implementação do Android para registrar seu aplicativo e receber um ID de registro do GCM. Assim que você tiver o ID do registro, será necessário registrar o dispositivo no Amazon Cognito, conforme mostrado no trecho abaixo:

```
String registrationId = "MY_GCM_REGISTRATION_ID";
try {
    client.registerDevice("GCM", registrationId);
} catch (RegistrationFailedException rfe) {
    Log.e(TAG, "Failed to register device for silent sync", rfe);
} catch (AmazonClientException ace) {
    Log.e(TAG, "An unknown error caused registration for silent sync to fail", ace);
}
```

Agora você pode inscrever um dispositivo para receber atualizações de um conjunto de dados específico:

```
Dataset trackedDataset = client.openOrCreateDataset("myDataset");
if (client.isDeviceRegistered()) {
```

```
    try {
        trackedDataset.subscribe();
    } catch (SubscribeFailedException sfe) {
        Log.e(TAG, "Failed to subscribe to datasets", sfe);
    } catch (AmazonClientException ace) {
        Log.e(TAG, "An unknown error caused the subscription to fail", ace);
    }
}
```

Para interromper o recebimento de notificações por push de um conjunto de dados, basta chamar o método unsubscribe. Para inscrever-se em todos os conjuntos de dados (ou em um subconjunto específico) do objeto `CognitoSyncManager`, use `subscribeAll()`:

```
if (client.isDeviceRegistered()) {
    try {
        client.subscribeAll();
    } catch (SubscribeFailedException sfe) {
        Log.e(TAG, "Failed to subscribe to datasets", sfe);
    } catch (AmazonClientException ace) {
        Log.e(TAG, "An unknown error caused the subscription to fail", ace);
    }
}
```

Na implementação do BroadcastReceiver objeto Android, você pode verificar a versão mais recente do conjunto de dados modificado e decidir se seu aplicativo precisa ser sincronizado novamente:

```
@Override
public void onReceive(Context context, Intent intent) {

    PushSyncUpdate update = client.getPushSyncUpdate(intent);

    // The update has the source (cognito-sync here), identityId of the
    // user, identityPoolId in question, the non-local sync count of the
    // data set and the name of the dataset. All are accessible through
    // relevant getters.

    String source = update.getSource();
    String identityPoolId = update.getIdentityPoolId();
    String identityId = update.getIdentityId();
    String datasetName = update.getDatasetName;
    long syncCount = update.getSyncCount;
```

```
    Dataset dataset = client.openOrCreateDataset(datasetName);

    // need to access last sync count. If sync count is less or equal to
    // last sync count of the dataset, no sync is required.

    long lastSyncCount = dataset.getLastSyncCount();
    if (lastSyncCount < syncCount) {
        dataset.synchronize(new SyncCallback() {
            // ...
        });
    }

}
```

As chaves a seguir estão disponíveis na carga útil de notificação por push:

- `source`: cognito-sync. Pode atuar como um fator de diferenciação entre notificações.
- `identityPoolId`: o ID do grupo de identidades. Pode ser usado para validação ou informações adicionais, embora não seja parte integrante do ponto de vista do receptor.
- `identityId`: o ID da identidade no grupo.
- `datasetName`: o nome do conjunto de dados atualizado. Isso está disponível para fins de chamada do openOrCreate conjunto de dados.
- `syncCount`: a contagem de sincronização do conjunto de dados remoto. Você pode usar esse recurso como certificar-se de que o conjunto de dados local está desatualizado e a sincronização de entrada é nova.

Usar sincronização por push em sua aplicação: iOS - Objective-C

Para obter um token de dispositivo para o aplicativo, siga a documentação da Apple sobre registro de notificações remotas. Depois de receber o token do dispositivo como um NSData objeto do APNs, você precisará registrar o dispositivo no Amazon Cognito usando o `registerDevice:` método do cliente de sincronização, conforme mostrado abaixo:

```
AWSCognito *syncClient = [AWSCognito defaultCognito];
    [[syncClient registerDevice: devToken] continueWithBlock:^id(AWSTask *task) {
        if(task.error){
            NSLog(@"Unable to registerDevice: %@", task.error);
        } else {
            NSLog(@"Successfully registered device with id: %@", task.result);
```

```
        }
        return nil;
      }
    ];
```

No modo de depuração, seu dispositivo será registrado na APNs sandbox; no modo de lançamento, ele será registrado com. APNs Para receber atualizações de um conjunto de dados específico, use o método `subscribe`:

```
[[[syncClient openOrCreateDataset:@"MyDataset"] subscribe]
 continueWithBlock:^id(AWSTask *task) {
        if(task.error){
            NSLog(@"Unable to subscribe to dataset: %@", task.error);
        } else {
            NSLog(@"Successfully subscribed to dataset: %@", task.result);
        }
        return nil;
      }
    ];
```

Para interromper o recebimento de notificações por push de um conjunto de dados, basta chamar o método `unsubscribe`:

```
[[[syncClient openOrCreateDataset:@”MyDataset”] unsubscribe]
 continueWithBlock:^id(AWSTask *task) {
        if(task.error){
            NSLog(@"Unable to unsubscribe from dataset: %@", task.error);
        } else {
            NSLog(@"Successfully unsubscribed from dataset: %@", task.result);
        }
        return nil;
      }
    ];
```

Para inscrever-se em todos os conjuntos de dados do objeto `AWSCognito`, chame `subscribeAll`:

```
[[syncClient subscribeAll] continueWithBlock:^id(AWSTask *task) {
        if(task.error){
            NSLog(@"Unable to subscribe to all datasets: %@", task.error);
        } else {
            NSLog(@"Successfully subscribed to all datasets: %@", task.result);
```

```
        }
        return nil;
      }
    ];
```

Antes de chamar `subscribeAll`, sincronize-se pelo menos uma vez em cada conjunto de dados, para que estes passem a existir no servidor.

Para reagir às notificações por push, é necessário implementar o método `didReceiveRemoteNotification` no aplicativo delegado:

```
- (void)application:(UIApplication *)application didReceiveRemoteNotification:
(NSDictionary *)userInfo
    {
        [[NSNotificationCenter defaultCenter]
 postNotificationName:@"CognitoPushNotification" object:userInfo];
    }
```

Se você publicar uma notificação usando o handler de notificação, poderá responder à notificação em qualquer outro lugar do aplicativo no qual há um handler para o conjunto de dados. Se você inscrever-se na notificação desta forma...

```
[[NSNotificationCenter defaultCenter] addObserver:self
 selector:@selector(didReceivePushSync:)
    name: :@"CognitoPushNotification" object:nil];
```

...poderá reagir à notificação desta forma:

```
- (void)didReceivePushSync:(NSNotification*)notification
    {
        NSDictionary * data = [(NSDictionary *)[notification object]
 objectForKey:@"data"];
        NSString * identityId = [data objectForKey:@"identityId"];
        NSString * datasetName = [data objectForKey:@"datasetName"];
        if([self.dataset.name isEqualToString:datasetName] && [self.identityId
 isEqualToString:identityId]){
            [[self.dataset synchronize] continueWithBlock:^id(AWSTask *task) {
                if(!task.error){
                    NSLog(@"Successfully synced dataset");
                }
                return nil;
```

```
            }];
        }
    }
```

As chaves a seguir estão disponíveis na carga útil de notificação por push:

- `source`: cognito-sync. Pode atuar como um fator de diferenciação entre notificações.
- `identityPoolId`: o ID do grupo de identidades. Pode ser usado para validação ou informações adicionais, embora não seja parte integrante do ponto de vista do receptor.
- `identityId`: o ID da identidade no grupo.
- `datasetName`: o nome do conjunto de dados atualizado. É disponibilizado graças à chamada de `openOrCreateDataset`.
- `syncCount`: a contagem de sincronização do conjunto de dados remoto. Você pode usar esse recurso como certificar-se de que o conjunto de dados local está desatualizado e a sincronização de entrada é nova.

Usar sincronização por push em sua aplicação: iOS - Swift

Para obter um token de dispositivo para o aplicativo, siga a documentação da Apple sobre registro de notificações remotas. Depois de receber o token do dispositivo como um NSData objeto do APNs, você precisará registrar o dispositivo no Amazon Cognito usando o método registerDevice: do cliente de sincronização, conforme mostrado abaixo:

```
let syncClient = AWSCognito.default()
syncClient.registerDevice(devToken).continueWith(block: { (task: AWSTask!) ->
 AnyObject! in
    if (task.error != nil) {
        print("Unable to register device: " + task.error.localizedDescription)

    } else {
        print("Successfully registered device with id: \(task.result)")
    }
    return task
})
```

No modo de depuração, seu dispositivo será registrado na APNs sandbox; no modo de lançamento, ele será registrado com. APNs Para receber atualizações de um conjunto de dados específico, use o método `subscribe`:

```
syncClient.openOrCreateDataset("MyDataset").subscribe().continueWith(block: { (task:
 AWSTask!) -> AnyObject! in
  if (task.error != nil) {
      print("Unable to subscribe to dataset: " + task.error.localizedDescription)

  } else {
      print("Successfully subscribed to dataset: \(task.result)")
  }
  return task
})
```

Para interromper o recebimento de notificações por push de um conjunto de dados, chame o método `unsubscribe`:

```
syncClient.openOrCreateDataset("MyDataset").unsubscribe().continueWith(block: { (task:
 AWSTask!) -> AnyObject! in
  if (task.error != nil) {
      print("Unable to unsubscribe to dataset: " + task.error.localizedDescription)

  } else {
      print("Successfully unsubscribed to dataset: \(task.result)")
  }
  return task
})
```

Para inscrever-se em todos os conjuntos de dados do objeto `AWSCognito`, chame `subscribeAll`:

```
syncClient.openOrCreateDataset("MyDataset").subscribeAll().continueWith(block: { (task:
 AWSTask!) -> AnyObject! in
  if (task.error != nil) {
      print("Unable to subscribe to all datasets: " + task.error.localizedDescription)

  } else {
      print("Successfully subscribed to all datasets: \(task.result)")
  }
  return task
})
```

Antes de chamar `subscribeAll`, sincronize-se pelo menos uma vez em cada conjunto de dados, para que estes passem a existir no servidor.

Para reagir às notificações por push, é necessário implementar o método `didReceiveRemoteNotification` no aplicativo delegado:

```
func application(application: UIApplication, didReceiveRemoteNotification userInfo:
 [NSObject : AnyObject],
  fetchCompletionHandler completionHandler: (UIBackgroundFetchResult) -> Void) {

 NSNotificationCenter.defaultCenter().postNotificationName("CognitoPushNotification",
 object: userInfo)
})
```

Se você publicar uma notificação usando o handler de notificação, poderá responder à notificação em qualquer outro lugar do aplicativo no qual há um handler para o conjunto de dados. Se você inscrever-se na notificação desta forma...

```
NSNotificationCenter.defaultCenter().addObserver(observer:self,
   selector:"didReceivePushSync:",
   name:"CognitoPushNotification",
   object:nil)
```

...poderá reagir à notificação desta forma:

```
func didReceivePushSync(notification: NSNotification) {
    if let data = (notification.object as! [String: AnyObject])["data"] as? [String:
 AnyObject] {
        let identityId = data["identityId"] as! String
        let datasetName = data["datasetName"] as! String

        if self.dataset.name == datasetName && self.identityId == identityId {
          dataset.synchronize().continueWithBlock {(task) -> AnyObject! in
              if task.error == nil {
                print("Successfully synced dataset")
              }
              return nil
          }
        }
    }
}
```

As chaves a seguir estão disponíveis na carga útil de notificação por push:

- `source`: cognito-sync. Pode atuar como um fator de diferenciação entre notificações.

- `identityPoolId`: o ID do grupo de identidades. Pode ser usado para validação ou informações adicionais, embora não seja parte integrante do ponto de vista do receptor.
- `identityId`: o ID da identidade no grupo.
- `datasetName`: o nome do conjunto de dados atualizado. É disponibilizado graças à chamada de `openOrCreateDataset`.
- `syncCount`: a contagem de sincronização do conjunto de dados remoto. Você pode usar esse recurso como certificar-se de que o conjunto de dados local está desatualizado e a sincronização de entrada é nova.

Como implementar o Amazon Cognito Sync Streams

> ⚠ Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Assim como o Amazon Cognito Sync, AWS AppSync é um serviço para sincronizar dados de aplicativos entre dispositivos.
> Ele permite que dados do usuário, como preferências de aplicações ou estado de jogos, sejam sincronizados. Ele também amplia essas capacidades ao permitir que vários usuários sincronizem e colaborem em tempo real com dados compartilhados.

O Amazon Cognito Streams oferece aos desenvolvedores controle e insight sobre os dados armazenados no Amazon Cognito. Agora, os desenvolvedores podem configurar um fluxo do Kinesis para receber eventos à medida que os dados forem atualizados e sincronizados. O Amazon Cognito pode enviar cada alteração de conjunto de dados a um fluxo do Kinesis de sua propriedade em tempo real.

Usando o Amazon Cognito Streams, você pode mover todos os dados de sincronização para o Kinesis, que poderão, em seguida, ser transmitidos para uma ferramenta de data warehouse, como o Amazon Redshift, para análise posterior. Para saber mais sobre o Kinesis, consulte Conceitos básicos do uso do Amazon Kinesis.

Como configurar transmissões

Você pode configurar o Amazon Cognito Streams no console do Amazon Cognito. Para habilitar o Amazon Cognito Streams no console do Amazon Cognito, você precisa selecionar o fluxo do Kinesis no qual será realizada a publicação e uma função do IAM que concede ao Amazon Cognito permissão para colocar eventos no fluxo selecionado.

Na página inicial do console:

1. Clique no nome do grupo de identidades para o qual você deseja configurar o Amazon Cognito Streams. A página Dashboard (Painel) do grupo de identidades será exibida.
2. No canto superior direito da página Dashboard (Painel), clique em Manage Identity Pools (Gerenciar grupos de identidades). A página Manage Federated Identities (Gerenciar identidades federadas) é exibida.
3. Role para baixo e clique em Cognito Streams para expandi-lo.
4. No menu suspenso Stream name, selecione o nome de um fluxo existente do Kinesis. Se desejar, clique em Create stream para criar um fluxo, informando um nome de fluxo e o número de estilhaços. Para saber mais sobre fragmentos e para obter ajuda sobre como estimar o número de fragmentos necessários para seu fluxo, consulte o Guia do desenvolvedor do Kinesis.
5. No menu suspenso Publish role (Publicar função), selecione a função do IAM que concede ao Amazon Cognito permissão para publicar seu fluxo. Clique em Create role (Criar função) para criar ou modificar as funções associadas ao grupo de identidades no console do AWS IAM.
6. No menu suspenso Stream status, selecione Enabled para habilitar as atualizações de fluxo. Clique em Salvar alterações

Depois que você tiver configurado com êxito os fluxos do Amazon Cognito, todas as atualizações subsequentes dos conjuntos de dados nesse grupo de identidades serão enviadas ao fluxo.

Conteúdo de transmissão

Cada registro enviado ao fluxo representa uma única sincronização. Este é um exemplo de um registro enviado ao fluxo:

```
{
    "identityPoolId": "Pool Id",
    "identityId": "Identity Id",
    "dataSetName": "Dataset Name",
    "operation": "(replace|remove)",
    "kinesisSyncRecords": [
        {
            "key": "Key",
            "value": "Value",
            "syncCount": 1,
            "lastModifiedDate": 1424801824343,
            "deviceLastModifiedDate": 1424801824343,
```

```
            "op": "(replace|remove)"
        },
        ...
    ],
    "lastModifiedDate": 1424801824343,
    "kinesisSyncRecordsURL": "S3Url",
    "payloadType": "(S3Url|Inline)",
    "syncCount": 1
}
```

Para atualizações maiores que o tamanho máximo de carga útil de 1 MB do Kinesis, o Amazon Cognito incluirá um URL pré-assinado do Amazon S3 com o conteúdo completo da atualização.

Depois de configurar fluxos do Amazon Cognito, se você excluir o fluxo do Kinesis ou alterar a permissão de confiança de função para que não possa mais ser presumida pelo Amazon Cognito Sync, os fluxos do Amazon Cognito serão desabilitados. Você deve recriar o fluxo do Kinesis ou corrigir a função e, depois, ativar o fluxo novamente.

Publicação em massa

Após ter configurado os fluxos do Amazon Cognito, você poderá executar uma operação de publicação em massa dos dados existentes no grupo de identidades. Depois de iniciar uma operação de publicação em massa, por meio do console ou diretamente por meio da API, o Amazon Cognito começará a publicar esses dados no mesmo fluxo que está recebendo as atualizações.

O Amazon Cognito não garante a exclusividade dos dados enviados ao fluxo durante o uso da operação de publicação em massa. Você pode receber a mesma atualização como uma atualização e como parte de uma publicação em massa. Mantenha isso em mente ao processar os registros do seu fluxo.

Para realizar uma publicação em massa de todos os fluxos, siga as etapas de 1 a 6 em Configuração de fluxos e clique em Start bulk publish. Você está limitado a uma operação de publicação em massa contínua a qualquer momento e a uma solicitação de publicação em massa bem-sucedida a cada 24 horas.

Como personalizar fluxos de trabalho com o Amazon Cognito Events

Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Assim como o Amazon Cognito Sync, AWS AppSync é um serviço para sincronizar dados de aplicativos entre dispositivos.
Ele permite que dados do usuário, como preferências de aplicações ou estado de jogos, sejam sincronizados. Ele também amplia essas capacidades ao permitir que vários usuários sincronizem e colaborem em tempo real com dados compartilhados.

O Amazon Cognito Events permite que você execute uma AWS Lambda função em resposta a eventos importantes no Amazon Cognito. O Amazon Cognito gera o evento Sync Trigger quando um conjunto de dados é sincronizado. Você pode usar o evento Sync Trigger para executar uma ação quando um usuário atualizar dados. A função pode avaliar e, opcionalmente, manipular os dados antes de serem armazenados na nuvem e sincronizados nos outros dispositivos do usuário. Isso será útil para validar os dados provenientes do dispositivo antes que eles sejam sincronizados com outros dispositivos do usuário ou para atualizar outros valores no conjunto de dados com base nos dados de entrada, como emitir um prêmio quando um jogador atinge um novo nível.

As etapas a seguir orientarão você durante a configuração de uma função Lambda executada toda vez que um conjunto de dados do Amazon Cognito for sincronizado.

Note

Ao usar eventos do Amazon Cognito, você só pode utilizar as credenciais obtidas no Amazon Cognito Identity. Se você tiver uma função Lambda associada, mas ligar `UpdateRecords` com credenciais da AWS conta (credenciais de desenvolvedor), sua função Lambda não será invocada.

Criando uma função em AWS Lambda

Para integrar o Lambda ao Amazon Cognito, primeiro é necessário criar uma função no Lambda. Para fazer isso:

Selecionar a função Lambda no Amazon Cognito

1. Abra o console do lambda.
2. Clique em Create a Lambda function (Criar uma função Lambda).
3. Na tela Selecionar modelo, pesquise e selecione ""cognito-sync-trigger.
4. Na tela Configure event sources, deixe Event source type definido como "Cognito Sync Trigger" e selecione o grupo de identidades. Clique em Next.

> **Note**
>
> Ao configurar um acionador do Amazon Cognito Sync fora do console, você deve adicionar permissões baseadas em recursos do Lambda para permitir que o Amazon Cognito invoque a função. Você pode adicionar essa permissão no console Lambda (consulte Uso de políticas baseadas em recursos para AWS Lambda) ou usando a operação Lambda. AddPermission
> Exemplo de política baseada em recursos do Lambda
> A seguinte política baseada em recursos do AWS Lambda concede ao Amazon Cognito uma capacidade limitada de invocar uma função Lambda. O Amazon Cognito só pode invocar a função em nome do grupo de identidades na condição `aws:SourceArn` e da conta na condição `aws:SourceAccount`.
>
> JSON

```
{
    "Version":"2012-10-17",
    "Id": "default",
    "Statement": [
        {
            "Sid": "lambda-allow-cognito-my-function",
            "Effect": "Allow",
            "Principal": {
                "Service": "cognito-sync.amazonaws.com"
            },
            "Action": "lambda:InvokeFunction",
            "Resource": "arn:aws:lambda:us-
east-1:111122223333:function:MyFunction",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "111122223333"
                },
```

```
                "ArnLike": {
                    "AWS:SourceArn": "arn:aws:cognito-
identity:us-east-1:111122223333:identitypool/us-
east-1:abcdefg-1234-5678-910a-0e8443553f95"
                }
            }
        }
    ]
}
```

5. Na tela Configure function, insira um nome e uma descrição para a função. Deixe Runtime definido como "Node.js". Deixe o código inalterado no nosso exemplo. O exemplo padrão não faz as alterações nos dados que estão sendo sincronizados. Ele só registra o fato de que o evento Sync Trigger do Amazon Cognito ocorreu. Deixe Handler name definido como "index.handler". Em Role (Função), selecione uma função do IAM que conceda permissão de código para acessar o AWS Lambda. Para modificar funções, consulte o console do IAM. Deixe a opção Advanced settings inalterada. Clique em Next.
6. Na tela Review, revise os detalhes e clique em Create function. A próxima página exibe a nova função Lambda.

Agora que você tem uma função apropriada gravada no Lambda, precisa escolher essa função como handler do evento Sync Trigger do Amazon Cognito. As etapas a seguir percorrerão esse processo.

Na página inicial do console:

1. Clique no nome do grupo de identidades para o qual você deseja configurar os eventos do Amazon Cognito. A página Dashboard (Painel) do grupo de identidades será exibida.
2. No canto superior direito da página Dashboard, clique em Manage Federated Identities. A página Manage Federated Identities (Gerenciar identidades federadas) é exibida.
3. Role a tela para baixo e clique em Cognito Events para expandi-lo.
4. No menu suspenso Sync Trigger, selecione a função Lambda que você quer acionar quando ocorre um evento Sync.
5. Clique em Salvar alterações

Agora, a função Lambda será executada todas as vezes que um conjunto de dados for sincronizado. A próxima seção abordará como ler e modificar os dados na função enquanto eles estão sendo sincronizados.

Como criar uma função Lambda para acionadores de sincronização

Os acionadores de sincronização seguem o padrão de programação que as interfaces do provedor de serviços usam. O Amazon Cognito fornece a entrada na função do Lambda no formato JSON a seguir.

```
{
  "version": 2,
  "eventType": "SyncTrigger",
  "region": "us-east-1",
  "identityPoolId": "identityPoolId",
  "identityId": "identityId",
  "datasetName": "datasetName",
  "datasetRecords": {
    "SampleKey1": {
      "oldValue": "oldValue1",
      "newValue": "newValue1",
      "op": "replace"
    },
    "SampleKey2": {
      "oldValue": "oldValue2",
      "newValue": "newValue2",
      "op": "replace"
    },..
  }
}
```

O Amazon Cognito espera o valor de retorno da função no mesmo formato da entrada.

Ao gravar funções para o evento Sync Trigger, observe o seguinte:

- Quando o Amazon Cognito chama sua função Lambda durante UpdateRecords, a função deve responder em 5 segundos. Se isso não ocorrer, o serviço Amazon Cognito Sync lançará uma exceção `LambdaSocketTimeoutException`. Você não pode aumentar esse valor de tempo limite.
- Se você receber uma exceção `LambdaThrottledException`, tente a operação de sincronização novamente para atualizar os registros.
- O Amazon Cognito fornece todos os registros presentes no conjunto de dados como entrada para a função.
- Os registros que o usuário da aplicação atualiza têm o campo op definido como `replace`. Os registros excluídos têm o campo op definido como `remove`.

- Você poderá modificar qualquer registro, mesmo se o usuário da aplicação não o atualizar.
- Todos os campos, exceto datasetRecords, são somente leitura. Não os altere. Se você alterar esses campos, não poderá atualizar os registros.
- Para modificar o valor de um registro, atualize o valor e defina op como `replace`.
- Para remover um registro, defina op como `remove` ou defina o valor como null.
- Para adicionar um registro, adicione um novo registro à matriz datasetRecords.
- O Amazon Cognito ignora qualquer registro omitido na resposta quando o Amazon Cognito o atualiza.

Amostra de função Lambda

O exemplo de função do Lambda a seguir mostra como acessar, modificar e remover os dados.

```
console.log('Loading function');

exports.handler = function(event, context) {
    console.log(JSON.stringify(event, null, 2));

    //Check for the event type
    if (event.eventType === 'SyncTrigger') {

        //Modify value for a key
        if('SampleKey1' in event.datasetRecords){
            event.datasetRecords.SampleKey1.newValue = 'ModifyValue1';
            event.datasetRecords.SampleKey1.op = 'replace';
        }

        //Remove a key
        if('SampleKey2' in event.datasetRecords){
            event.datasetRecords.SampleKey2.op = 'remove';
        }

        //Add a key
        if(!('SampleKey3' in event.datasetRecords)){
            event.datasetRecords.SampleKey3={'newValue':'ModifyValue3', 'op' :
 'replace'};
        }

    }
    context.done(null, event);
```

```
};
```

Segurança no Amazon Cognito

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de uma arquitetura de data center e rede criada para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O modelo de responsabilidade compartilhada descreve isso como a segurança da nuvem e a segurança na nuvem:

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços na AWS nuvem. AWS também fornece serviços que você pode usar com segurança. Auditores terceirizados testam e verificam regularmente a eficácia de nossa segurança como parte dos Programas de Conformidade Programas de AWS de . Para saber mais sobre os programas de conformidade que se aplicam ao Amazon Cognito, consulte AWS Serviços em Escopo por Programa de Conformidade AWS Serviços em Escopo por Programa .
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade dos dados, os requisitos da empresa e as leis e os regulamentos aplicáveis

Esta documentação ajuda você a entender como aplicar o modelo de responsabilidade compartilhada ao usar o Amazon Cognito. Ela mostra como configurar o Amazon Cognito para atender aos seus objetivos de segurança e compatibilidade. Você também aprenderá a usar outros AWS serviços que ajudam a monitorar e proteger seus recursos do Amazon Cognito.

Conteúdo

Proteção de dados no Amazon Cognito

O modelo de responsabilidade AWS compartilhada O modelo se aplica à proteção de dados no Amazon Cognito (Amazon Cognito). Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa toda a AWS nuvem. Você é responsável por manter o controle sobre o conteúdo hospedado nessa infraestrutura. Esse conteúdo inclui as tarefas de configuração e gerenciamento de segurança dos AWS serviços que você usa. Para obter mais informações sobre a privacidade de dados, consulte as Perguntas frequentes sobre privacidade de dados.

Para fins de proteção de dados, recomendamos que você proteja as credenciais da AWS conta e configure contas de usuário individuais com 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 AWS os recursos.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail.
- Use soluções AWS de criptografia, juntamente com todos os controles de segurança padrão nos AWS serviços.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados pessoais armazenados no Amazon S3.

É altamente recomendável que você nunca coloque informações de identificação confidenciais, como números de conta dos seus clientes, em campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com o Amazon Cognito ou outros AWS serviços usando o console, a API ou. AWS CLI AWS SDKs Todos os dados inseridos no Amazon Cognito ou em outros serviços poderão ser selecionados para inclusão em logs de diagnóstico. Ao fornecer um URL para um servidor externo, não inclua informações de credenciais no URL para validar a solicitação a esse servidor.

Criptografia de dados

A criptografia de dados geralmente se encaixa em duas categorias: criptografia em repouso e criptografia em trânsito.

Criptografia em repouso

Os dados no Amazon Cognito são criptografados em repouso de acordo com os padrões do setor.

O Amazon Cognito oferece suporte à confidencialidade, integridade e disponibilidade de informações de identificação pessoal em pesquisas de atributos do usuário com criptografia pesquisável. Essas funções de Código de Autenticação de Mensagens (HMAC) baseadas em hash, com desempenho otimizado para conjuntos de dados de grupos de usuários, mapeiam entre o texto simples e os valores criptografados dos atributos do usuário. O Amazon Cognito calcula valores de HMAC com a chave KMS que criptografa seu grupo de usuários. Essa proteção se aplica aos seguintes atributos:

- sub
- email
- phone_number
- given_name
- family_name
- name
- username
- preferred_username
- cognito:user_status

Criptografia em trânsito

Como um serviço gerenciado, o Amazon Cognito é protegido pela segurança de rede AWS global. Para obter informações sobre serviços AWS de segurança e como AWS proteger a infraestrutura, consulte AWS Cloud Security. Para projetar seu AWS ambiente usando as melhores práticas de segurança de infraestrutura, consulte Proteção de infraestrutura no Security Pillar AWS Well-Architected Framework.

Você usa chamadas de API AWS publicadas para acessar o Amazon Cognito pela rede. Os clientes devem oferecer compatibilidade com:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com perfect forward secrecy (PFS) como DHE (Ephemeral Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Os grupos de usuários e os bancos de identidades do Amazon Cognito têm operações de API autenticadas, não autenticadas e autorizadas por token pelo IAM. As operações de API não autenticadas e autorizadas por token devem ser utilizadas por seus clientes, os usuários finais da sua aplicação. As operações de API não autenticadas e autorizadas por tokens são criptografadas em repouso e em trânsito. Para obter mais informações, consulte Lista de operações de API agrupadas por modelo de autorização.

> **Note**
>
> O Amazon Cognito criptografa o conteúdo internamente e não é compatível com chaves fornecidas pelo cliente.

Gerenciamento de identidade e acesso para o Amazon Cognito

AWS Identity and Access Management (IAM) é uma ferramenta AWS service (Serviço da AWS) que ajuda o administrador a controlar com segurança o acesso aos AWS recursos. Os administradores do IAM controlam quem pode ser autenticado (conectado) e autorizado (ter permissões) a usar os recursos do Amazon Cognito. O IAM é um AWS service (Serviço da AWS) que você pode usar sem custo adicional.

Tópicos

- Público
- Autenticação com identidades
- Gerenciar o acesso usando políticas
- Como o Amazon Cognito funciona com o IAM
- Exemplos de políticas baseadas em identidade para o Amazon Cognito
- Solução de problemas de identidade e acesso do Amazon Cognito
- Como usar funções vinculadas a serviço para o Amazon Cognito

Público

A forma como você usa AWS Identity and Access Management (IAM) difere com base na sua função:

- Usuário do serviço: solicite permissões ao seu administrador se você não conseguir acessar os atributos (consulte Solução de problemas de identidade e acesso do Amazon Cognito).
- Administrador do serviço: determine o acesso do usuário e envie solicitações de permissão (consulte Como o Amazon Cognito funciona com o IAM)
- Administrador do IAM: escreva políticas para gerenciar o acesso (consulte Exemplos de políticas baseadas em identidade para o Amazon Cognito)

Autenticação com identidades

A autenticação é a forma como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado como usuário do IAM ou assumindo uma função do IAM. Usuário raiz da conta da AWS

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

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

Conta da AWS usuário root

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

Identidade federada

Como prática recomendada, exija que os usuários humanos usem a federação com um provedor de identidade para acessar Serviços da AWS usando credenciais temporárias.

Uma identidade federada é um usuário do seu diretório corporativo, provedor de identidade da web ou Directory Service que acessa Serviços da AWS usando credenciais de uma fonte de identidade. As identidades federadas assumem funções que oferecem credenciais temporárias.

Para o gerenciamento de acesso centralizado, recomendamos Centro de Identidade do AWS IAM. Para saber mais, consulte O que é o IAM Identity Center? no Guia do usuário do Centro de Identidade do AWS IAM .

Usuários e grupos do IAM

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

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

Perfis do IAM

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

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

Gerenciar o acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política define permissões quando associada a uma identidade ou recurso. AWS avalia essas políticas quando um diretor faz uma solicitação. A maioria das políticas é armazenada AWS como documentos JSON. Para ter mais informações sobre documentos de política JSON, consulte Visão geral das políticas JSON no Guia do usuário do IAM.

Por meio de políticas, os administradores especificam quem tem acesso a que, definindo qual entidade principal pode realizar ações em quais recursos e sob quais condições.

Por padrão, usuários e perfis não têm permissões. Um administrador do IAM cria políticas do IAM e as adiciona aos perfis, os quais os usuários podem então assumir. As políticas do IAM definem permissões, independentemente do método usado para realizar a operação.

Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissão JSON que você anexa a uma identidade (usuário, grupo ou perfil). Essas políticas controlam quais ações as identidades podem realizar, em quais recursos e sob quais condições. Para saber como criar uma política baseada em identidade, consulte Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente no Guia do Usuário do IAM.

As políticas baseadas em identidade podem ser políticas em linha (incorporadas diretamente em uma única identidade) ou políticas gerenciadas (políticas autônomas anexadas a várias identidades). Para saber como escolher entre uma política gerenciada e políticas em linha, consulte Escolher entre políticas gerenciadas e políticas em linha no Guia do usuário do IAM.

Políticas baseadas em recursos

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. Entre os exemplos estão políticas de confiança de perfil do IAM e políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. É necessário especificar uma entidade principal em uma política baseada em recursos.

Políticas baseadas em recursos são políticas em linha localizadas nesse serviço. Você não pode usar políticas AWS gerenciadas do IAM em uma política baseada em recursos.

Outros tipos de política

AWS oferece suporte a tipos de políticas adicionais que podem definir o máximo de permissões concedidas por tipos de políticas mais comuns:

- Limites de permissões: definem o número máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM. Para saber mais sobre limites de permissões, consulte Limites de permissões para identidades do IAM no Guia do usuário do IAM.
- Políticas de controle de serviço (SCPs) — Especifique as permissões máximas para uma organização ou unidade organizacional em AWS Organizations. Para saber mais, consulte Políticas de controle de serviço no Guia do usuário do AWS Organizations .

- Políticas de controle de recursos (RCPs) — Defina o máximo de permissões disponíveis para recursos em suas contas. Para obter mais informações, consulte Políticas de controle de recursos (RCPs) no Guia AWS Organizations do usuário.
- Políticas de sessão: políticas avançadas transmitidas como um parâmetro durante a criação de uma sessão temporária para um perfil ou um usuário federado. Para saber mais, consulte Políticas de sessão no Guia do usuário do IAM.

Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como AWS determinar se uma solicitação deve ser permitida quando vários tipos de políticas estão envolvidos, consulte Lógica de avaliação de políticas no Guia do usuário do IAM.

Como o Amazon Cognito funciona com o IAM

Antes de usar o IAM para gerenciar o acesso ao Amazon Cognito, saiba quais atributos do IAM estão disponíveis para uso com o Amazon Cognito.

Recursos do IAM que você pode usar com o Amazon Cognito

Recurso do IAM	Suporte ao Amazon Cognito
Políticas baseadas em identidade	Sim
Políticas baseadas em recurso	Não
Ações de políticas	Sim
Recursos de políticas	Sim
Chaves de condição de políticas	Sim
ACLs	Não
ABAC (tags em políticas)	Parcial
Credenciais temporárias	Sim

Recurso do IAM	Suporte ao Amazon Cognito
Permissões de entidade principal	Não
Perfis de serviço	Sim
Perfis vinculados a serviço	Sim

Para ter uma visão de alto nível de como o Amazon Cognito e AWS outros serviços funcionam com a maioria dos recursos do IAM, AWS consulte os serviços que funcionam com o IAM no Guia do usuário do IAM.

Políticas baseadas em identidade do Amazon Cognito

Compatível com políticas baseadas em identidade: sim

As políticas baseadas em identidade são documentos de políticas de permissões JSON que podem ser anexados a uma identidade, como usuário do IAM, grupo de usuários ou perfil. Essas políticas controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente no Guia do Usuário do IAM.

Com as políticas baseadas em identidade do IAM, é possível especificar ações e recursos permitidos ou negados, assim como as condições sob as quais as ações são permitidas ou negadas. Para saber mais sobre todos os elementos que podem ser usados em uma política JSON, consulte Referência de elemento de política JSON do IAM no Guia do usuário do IAM.

Exemplos de políticas baseadas em identidade para o Amazon Cognito

Para ver exemplos de políticas baseadas em identidade do Amazon Cognito, consulte Exemplos de políticas baseadas em identidade para o Amazon Cognito.

Políticas baseadas em recursos no Amazon Cognito

Compatibilidade com políticas baseadas em recursos: não

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos,

os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o atributo ao qual a política está anexada, a política define quais ações uma entidade principal especificado pode executar nesse atributo e em que condições. É necessário especificar uma entidade principal em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Para permitir o acesso entre contas, é possível especificar uma conta inteira ou as entidades do IAM em outra conta como a entidade principal em uma política baseada em recursos. Consulte mais informações em Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Ações de políticas para o Amazon Cognito

Compatível com ações de políticas: sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento `Action` de uma política JSON descreve as ações que podem ser usadas para permitir ou negar acesso em uma política. Incluem ações em uma política para conceder permissões para executar a operação associada.

Para ver uma lista das ações do Amazon Cognito, consulte Ações definidas pelo Amazon Cognito na Referência de autorização do serviço.

As ações de políticas no Amazon Cognito usam o seguinte prefixo antes da ação:

```
cognito-identity
```

Para especificar várias ações em uma única declaração, separe-as com vírgulas.

```
"Action": [
      "cognito-identity:action1",
      "cognito-identity:action2"
         ]
```

Assinado versus não assinado APIs

Ao assinar solicitações da API do Amazon Cognito com AWS credenciais, você pode restringi-las em uma política AWS Identity and Access Management (IAM). As solicitações de API que devem ser assinadas com credenciais da AWS incluem login do lado do servidor com `AdminInitiateAuth` e

ações que criam, visualizam ou modificam recursos do Amazon Cognito, como `UpdateUserPool`. Para obter mais informações sobre solicitações de API assinadas, consulte Assinatura de solicitações de AWS API.

Como o Amazon Cognito é um produto de identidade do consumidor para aplicativos que você deseja disponibilizar ao público, você tem acesso aos seguintes itens não assinados. APIs Sua aplicação faz essas solicitações de API aos seus usuários e usuários em potencial. Alguns não APIs exigem autorização prévia, como `InitiateAuth` iniciar uma nova sessão de autenticação. Alguns APIs usam tokens de acesso ou chaves de sessão para autorização, como `VerifySoftwareToken` para concluir a configuração de MFA para um usuário que já tem uma sessão autenticada. Uma API de grupos de usuários do Amazon Cognito autorizada, não assinada, aceita um parâmetro `Session` ou `AccessToken` na sintaxe da solicitação, conforme exibido na Referência de API do Amazon Cognito. Uma API não assinada do Amazon Cognito Identity aceita um parâmetro `IdentityId`, conforme exibido na Referência de API de identidades federadas do Amazon Cognito.

Para ter mais informações sobre os modelos de autorização e funções de operações da API de grupos de usuários do Amazon Cognito, consulte Lista de operações de API agrupadas por modelo de autorização.

Operações da API de bancos de identidades do Amazon Cognito

- `GetId`
- `GetOpenIdToken`
- `GetCredentialsForIdentity`
- `UnlinkIdentity`

Operações da API de grupos de usuários do Amazon Cognito

- `AssociateSoftwareToken`
- `ChangePassword`
- `ConfirmDevice`
- `ConfirmForgotPassword`
- `ConfirmSignUp`
- `DeleteUser`
- `DeleteUserAttributes`
- `ForgetDevice`

- `ForgotPassword`
- `GetDevice`
- `GetUser`
- `GetUserAttributeVerificationCode`
- `GlobalSignOut`
- `InitiateAuth`
- `ListDevices`
- `ResendConfirmationCode`
- `RespondToAuthChallenge`
- `RevokeToken`
- `SetUserMFAPreference`
- `SetUserSettings`
- `SignUp`
- `UpdateAuthEventFeedback`
- `UpdateDeviceStatus`
- `UpdateUserAttributes`
- `VerifySoftwareToken`
- `VerifyUserAttribute`

Para ver exemplos de políticas baseadas em identidade do Amazon Cognito, consulte Exemplos de políticas baseadas em identidade para o Amazon Cognito.

Recursos de políticas para o Amazon Cognito

Compatível com recursos de políticas: sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento de política JSON `Resource` especifica o objeto ou os objetos aos quais a ação se aplica. Como prática recomendada, especifique um recurso usando seu nome do recurso da Amazon (ARN). Para ações que não oferecem compatibilidade com permissões em nível de recurso, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*"
```

Nomes de recursos da Amazon (ARNs)

ARNs para identidades federadas do Amazon Cognito

Nos grupos de identidades do Amazon Cognito (identidades federadas), é possível restringir o acesso de um usuário do IAM a um determinado grupo de identidades usando o formato de nome do recurso da Amazon (ARN), como no exemplo a seguir. Para obter mais informações sobre ARNs, consulte Identificadores do IAM.

```
arn:aws:cognito-identity:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID
```

ARNs para Amazon Cognito Sync

Na sincronização do Amazon Cognito, os clientes também podem restringir o acesso por ID de grupo de identidades, ID de identidade e nome do conjunto de dados.

Para APIs operar em um grupo de identidades, o formato ARN do grupo de identidades é o mesmo das Identidades Federadas do Amazon Cognito, exceto que o nome do serviço é em vez de: `cognito-sync` `cognito-identity`

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID
```

Para APIs isso, opere em uma única identidade, como`RegisterDevice`, por exemplo, você pode se referir à identidade individual pelo seguinte formato de ARN:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID/
identity/IDENTITY_ID
```

Para APIs isso, opere em conjuntos de dados, como `UpdateRecords` e`ListRecords`, você pode consultar o conjunto de dados individual usando o seguinte formato ARN:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID/
identity/IDENTITY_ID/dataset/DATASET_NAME
```

ARNs para grupos de usuários do Amazon Cognito

Para seus grupos de usuários do Amazon Cognito, é possível restringir o acesso de um usuário a um grupo de usuários específico, usando o seguinte formato de ARN:

```
arn:aws:cognito-idp:REGION:ACCOUNT_ID:userpool/USER_POOL_ID
```

Para ver uma lista dos tipos de recursos do Amazon Cognito e seus ARNs, consulte Recursos definidos pelo Amazon Cognito na Referência de autorização de serviço. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte Ações definidas pelo Amazon Cognito.

Para ver exemplos de políticas baseadas em identidade do Amazon Cognito, consulte Exemplos de políticas baseadas em identidade para o Amazon Cognito.

Chaves de condição de política do Amazon Cognito

Compatível com chaves de condição de política específicas de serviço: sim

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

O elemento `Condition` especifica quando as instruções são executadas com base em critérios definidos. É possível criar expressões condicionais que usem agentes de condição, como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação. Para ver todas as chaves de condição AWS globais, consulte as chaves de contexto de condição AWS global no Guia do usuário do IAM.

Para ver uma lista de chaves de condição do Amazon Cognito, consulte Chaves de condição do Amazon Cognito na Referência de autorização do serviço. Para saber com quais ações e recursos você pode usar a chave de condição, consulte Ações definidas pelo Amazon Cognito.

Para ver exemplos de políticas baseadas em identidade do Amazon Cognito, consulte Exemplos de políticas baseadas em identidade para o Amazon Cognito.

Listas de controle de acesso (ACLs) no Amazon Cognito

Suportes ACLs: Não

As listas de controle de acesso (ACLs) controlam quais diretores (membros da conta, usuários ou funções) têm permissões para acessar um recurso. ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

Controle de acesso por atributo (ABAC) com o Amazon Cognito

Compatível com ABAC (tags em políticas): parcial

O controle de acesso por atributo (ABAC) é uma estratégia de autorização que define permissões com base em atributos chamados de tags. Você pode anexar tags a entidades e AWS recursos do IAM e, em seguida, criar políticas ABAC para permitir operações quando a tag do diretor corresponder à tag no recurso.

Para controlar o acesso baseado em tags, forneça informações sobre as tags no elemento de condição de uma política usando as `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou chaves de condição `aws:TagKeys`.

Se um serviço for compatível com as três chaves de condição para cada tipo de recurso, o valor será Sim para o serviço. Se um serviço for compatível com as três chaves de condição somente para alguns tipos de recursos, o valor será Parcial

Para saber mais sobre o ABAC, consulte Definir permissões com autorização do ABAC no Guia do usuário do IAM. Para visualizar um tutorial com etapas para configurar o ABAC, consulte Usar controle de acesso por atributo (ABAC) no Guia do usuário do IAM.

Usar credenciais temporárias com o Amazon Cognito

Compatível com credenciais temporárias: sim

As credenciais temporárias fornecem acesso de curto prazo aos AWS recursos e são criadas automaticamente quando você usa a federação ou troca de funções. AWS recomenda que você gere credenciais temporárias dinamicamente em vez de usar chaves de acesso de longo prazo. Para obter mais informações, consulte Credenciais de segurança temporárias no IAM e Serviços da AWS que funcionam com o IAM no Guia do usuário do IAM.

Permissões de entidade principal entre serviços para o Amazon Cognito

Compatível com o recurso de encaminhamento de sessões de acesso (FAS): Não

As sessões de acesso direto (FAS) usam as permissões do principal chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) de fazer solicitações aos serviços posteriores. Para obter detalhes da política ao fazer solicitações de FAS, consulte Sessões de acesso direto.

Perfis de serviço para o Amazon Cognito

Compatível com perfis de serviço: sim

O perfil de serviço é um perfil do IAM que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para saber mais, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.

Para obter detalhes sobre os perfis de serviço do Amazon Cognito, consulte Ativar a sincronização por push e Como implementar a sincronização por push.

> **Warning**
>
> A alteração das permissões de um perfil de serviço pode interromper a funcionalidade do Amazon Cognito. Edite perfis de serviço somente quando o Amazon Cognito fornecer orientação para isso.

Como usar perfis vinculados ao serviço para o Amazon Cognito

Compatibilidade com perfis vinculados a serviços: sim

Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode assumir o perfil de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para perfis vinculados ao serviço.

Para obter detalhes sobre como criar ou gerenciar perfis vinculadas ao serviço do Amazon Cognito, consulte Como usar funções vinculadas a serviço para o Amazon Cognito.

Exemplos de políticas baseadas em identidade para o Amazon Cognito

Por padrão, usuários e perfis não têm permissão para criar nem modificar recursos do Amazon Cognito. Para conceder permissão aos usuários para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM.

Para aprender a criar uma política baseada em identidade do IAM ao usar esses documentos de política em JSON de exemplo, consulte Criar políticas do IAM (console) no Guia do usuário do IAM.

Para obter detalhes sobre ações e tipos de recursos definidos pelo Amazon Cognito, incluindo o formato de cada um dos ARNs tipos de recursos, consulte Ações, recursos e chaves de condição para o Amazon Cognito na Referência de autorização de serviço.

Tópicos

Práticas recomendadas de política

As políticas baseadas em identidade determinam se alguém pode criar, acessar ou excluir recursos do Amazon Cognito em sua conta. Essas ações podem incorrer em custos para sua Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas AWS gerenciadas e avance para as permissões de privilégios mínimos — Para começar a conceder permissões aos seus usuários e cargas de trabalho, use as políticas AWS gerenciadas que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis no seu Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo AWS cliente que sejam específicas para seus casos de uso. Para saber mais, consulte Políticas gerenciadas pela AWS ou Políticas gerenciadas pela AWS para funções de trabalho no Guia do usuário do IAM.
- 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. Para saber mais sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no IAM no Guia do usuário do IAM.
- Use condições nas políticas do IAM para restringir ainda mais o acesso: é possível adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, é possível escrever uma condição de política para especificar que todas as solicitações devem ser enviadas usando SSL. Você também pode usar condições para conceder acesso às ações de serviço se elas forem usadas por meio de uma ação específica AWS service (Serviço da AWS), como CloudFormation. Para saber mais, consulte Elementos da política JSON do IAM: condição no Guia do usuário do IAM.
- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais: o IAM Access Analyzer valida as políticas novas e existentes para que elas

sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de cem verificações de política e recomendações práticas para ajudar a criar políticas seguras e funcionais. Para saber mais, consulte Validação de políticas do IAM Access Analyzer no Guia do Usuário do IAM.

- Exigir autenticação multifator (MFA) — Se você tiver um cenário que exija usuários do IAM ou um usuário root, ative Conta da AWS a MFA para obter segurança adicional. Para exigir MFA quando as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para saber mais, consulte Configuração de acesso à API protegido por MFA no Guia do Usuário do IAM.

Para saber mais sobre as práticas recomendadas do IAM, consulte Práticas recomendadas de segurança no IAM no Guia do usuário do IAM.

> **Note**
>
> As versões original e nova do console do Amazon Cognito têm um comportamento subjacente diferente quando você visualiza e modifica seus recursos do Amazon Cognito. Se conceder permissão para ações com o prefixo de serviço `cognito-idp` somente quando a condição `aws:ViaAWSService` for verdadeira, a entidade principal afetada do IAM poderá ser eficaz para os recursos do Amazon Cognito no console original, mas não no novo. Para trabalhar no console do Amazon Cognito, não defina uma condição `aws:ViaAWSService` nas permissões do Amazon Cognito em sua política do IAM.

Como usar o console do Amazon Cognito

Para acessar o console da Amazon Cognito, você deve ter um conjunto mínimo de permissões. Essas permissões devem permitir que você liste e visualize detalhes sobre os recursos do Amazon Cognito em seu. Conta da AWS Caso crie uma política baseada em identidade mais restritiva que as permissões mínimas necessárias, o console não funcionará como pretendido para entidades (usuários ou perfis) com essa política.

Você não precisa permitir permissões mínimas do console para usuários que estão fazendo chamadas somente para a API AWS CLI ou para a AWS API. Em vez disso, permita o acesso somente a ações que correspondam à operação de API que estiverem tentando executar.

Para garantir que usuários e funções ainda possam usar o console do Amazon Cognito, anexe também o Amazon `ConsoleAccess` Cognito `ReadOnly` AWS ou a política gerenciada às entidades. Para obter informações, consulte Adicionar permissões a um usuário no Guia do usuário do IAM.

Permitir que os usuários visualizem suas próprias permissões

Este exemplo mostra como criar uma política que permita que os usuários do IAM visualizem as políticas gerenciadas e em linha anexadas a sua identidade de usuário. Essa política inclui permissões para concluir essa ação no console ou programaticamente usando a API AWS CLI ou AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Como restringir o acesso do console a um grupo específico de identidades

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cognito-identity:ListIdentityPools"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cognito-identity:*"
            ],
            "Resource": "arn:aws:cognito-identity:us-
east-1:111122223333:identitypool/us-east-1:1a1a1a1a-ffff-1111-9999-12345678"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cognito-sync:*"
            ],
            "Resource": "arn:aws:cognito-sync:us-
east-1:111122223333:identitypool/us-east-1:1a1a1a1a-ffff-1111-9999-12345678"
        }
    ]
}
```

Como permitir o acesso a um conjunto de dados específico para todas as identidades em um grupo

JSON

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cognito-sync:ListRecords",
                "cognito-sync:UpdateRecords"
            ],
            "Resource": "arn:aws:cognito-sync:us-
east-1:111122223333:identitypool/us-east-1:1a1a1a1a-ffff-1111-9999-12345678/
identity/*/dataset/UserProfile"
        }
    ]
}
```

Solução de problemas de identidade e acesso do Amazon Cognito

Use as seguintes informações para ajudar a diagnosticar e corrigir problemas comuns que podem ser encontrados ao trabalhar com o Amazon Cognito e o IAM:

Tópicos

- Não tenho autorização para executar uma ação no Amazon Cognito
- Não estou autorizado a realizar iam: PassRole
- Sou administrador e desejo permitir que outras pessoas tenham acesso ao Amazon Cognito
- Quero permitir que pessoas fora da minha AWS conta acessem meus recursos do Amazon Cognito

Não tenho autorização para executar uma ação no Amazon Cognito

Se você receber uma mensagem de erro informando que não tem autorização para executar uma ação, suas políticas deverão ser atualizadas para permitir que você realize a ação.

O erro do exemplo a seguir ocorre quando o usuário do IAM `mateojackson` tenta usar o console para visualizar detalhes sobre um atributo *`my-example-widget`* fictício, mas não tem as permissões `cognito-identity:`*`GetWidget`* fictícias.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 cognito-identity:GetWidget on resource: my-example-widget
```

Nesse caso, a política do usuário `mateojackson` deve ser atualizada para permitir o acesso ao recurso *`my-example-widget`* usando a ação `cognito-identity:`*`GetWidget`*.

Se precisar de ajuda, entre em contato com seu AWS administrador. Seu administrador é a pessoa que forneceu suas credenciais de login.

Não estou autorizado a realizar iam: PassRole

Se receber uma mensagem de erro informando que você não tem autorização para executar a ação `iam:PassRole`, suas políticas deverão ser atualizadas para permitir a transmissão de um perfil ao Amazon Cognito.

Alguns Serviços da AWS permitem que você passe uma função existente para esse serviço em vez de criar uma nova função de serviço ou uma função vinculada ao serviço. Para fazer isso, é preciso ter permissões para passar o perfil para o serviço.

O erro do exemplo a seguir ocorre quando uma usuária do IAM chamada `marymajor` tenta usar o console para executar uma ação no Amazon Cognito. No entanto, a ação exige que o serviço tenha permissões concedidas por um perfil de serviço. Mary não tem permissões para passar o perfil para o serviço.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Nesse caso, as políticas de Mary devem ser atualizadas para permitir que ela realize a ação `iam:PassRole`.

Se precisar de ajuda, entre em contato com seu AWS administrador. Seu administrador é a pessoa que forneceu suas credenciais de login.

Sou administrador e desejo permitir que outras pessoas tenham acesso ao Amazon Cognito

Para permitir que outras pessoas acessem o Amazon Cognito, você deve conceder permissão às pessoas ou aplicações que precisam de acesso. Se você estiver usando o Centro de

Identidade do AWS IAM para gerenciar pessoas e aplicações, atribua conjuntos de permissões a usuários ou grupos para definir o nível de acesso. Os conjuntos de permissões criam e atribuem automaticamente políticas do IAM aos perfis do IAM associados à pessoa ou aplicação. Para ter mais informações, consulte Conjuntos de permissões no Guia do usuário do Centro de Identidade do AWS IAM .

Se você não estiver usando o Centro de Identidade do IAM, deverá criar entidades do IAM (usuários ou perfis) para as pessoas ou aplicações que precisam de acesso. Você deve anexar uma política à entidade que concede a elas as permissões corretas no Amazon Cognito. Depois que as permissões forem concedidas, forneça as credenciais ao usuário ou desenvolvedor da aplicação. Eles usarão essas credenciais para acessar AWS. Para saber mais sobre como criar grupos, políticas, permissões e usuários do IAM, consulte Identidades do IAM e Políticas e permissões no IAM no Guia do usuário do IAM.

Quero permitir que pessoas fora da minha AWS conta acessem meus recursos do Amazon Cognito

É possível criar um perfil que os usuários de outras contas ou pessoas fora da organização podem usar para acessar seus recursos. É possível especificar quem é confiável para assumir o perfil. Para serviços que oferecem suporte a políticas baseadas em recursos ou listas de controle de acesso (ACLs), você pode usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte:

- Para saber se o Amazon Cognito é compatível esses atributos, consulte Como o Amazon Cognito funciona com o IAM.
- Para saber como fornecer acesso aos seus recursos em todos os Contas da AWS que você possui, consulte Como fornecer acesso a um usuário do IAM em outro Conta da AWS que você possui no Guia do usuário do IAM.
- Para saber como fornecer acesso aos seus recursos a terceiros Contas da AWS, consulte Como fornecer acesso Contas da AWS a terceiros no Guia do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte Conceder acesso a usuários autenticados externamente (federação de identidades) no Guia do usuário do IAM.
- Para saber a diferença entre perfis e políticas baseadas em recurso para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.

Como usar funções vinculadas a serviço para o Amazon Cognito

O Amazon Cognito usa funções vinculadas a serviços AWS Identity and Access Management (IAM). Uma função vinculada ao serviço é um tipo exclusivo de função do IAM com uma política de confiança que permite que um assuma AWS service (Serviço da AWS) a função. As funções vinculadas ao serviço são predefinidas pelo Amazon Cognito e incluem todas as permissões que o serviço exige para chamar outros AWS serviços em seu nome.

Uma função vinculada ao serviço facilita a configuração do Amazon Cognito porque você não precisa adicionar as permissões necessárias manualmente. O Amazon Cognito define as permissões das funções vinculadas ao serviço e, exceto se definido de outra forma, somente o Amazon Cognito pode assumir suas funções. As permissões definidas incluem a política de confiança e a política de permissões, que não pode ser anexada a nenhuma outra entidade do IAM.

Um perfil vinculado ao serviço poderá ser excluído somente após excluir seus atributos relacionados. Isso protege seus recursos do Amazon Cognito, pois você não pode remover por engano as permissões para acessar os recursos.

Para obter informações sobre outros serviços compatíveis com perfis vinculados a serviços, consulte Serviços da AWS compatíveis com o IAM e procure os serviços que contenham Sim na coluna Service-Linked Role. Escolha um Sim com um link para visualizar a documentação do perfil vinculado para esse serviço.

Permissões de função vinculada a serviço para o Amazon Cognito

O Amazon Cognito usa funções vinculadas ao serviço:

- AWSServiceRoleForAmazonCognitoIdpEmailService— Permite que o serviço de grupos de usuários do Amazon Cognito use suas identidades do Amazon SES para enviar e-mails.

- AWSServiceRoleForAmazonCognitoIdp— Permite que grupos de usuários do Amazon Cognito publiquem eventos e configurem endpoints para seus projetos do Amazon Pinpoint.

AWSServiceRoleForAmazonCognitoIdpEmailService

O perfil vinculado ao serviço `AWSServiceRoleForAmazonCognitoIdpEmailService` confia nos seguintes serviços para aceitar o perfil:

- `email.cognito-idp.amazonaws.com`

A política de permissões da função permite que o Amazon Cognito conclua as seguintes ações nos recursos especificados:

Ações permitidas para AWSServiceRoleForAmazonCognitoIdpEmailService:

- Ação: `ses:SendEmail` e `ses:SendRawEmail`
- Recurso: *

A política nega ao Amazon Cognito a capacidade de realizar as seguintes ações nos recursos especificados:

Ações negadas

- Ação: `ses:List*`
- Recurso: *

Com essas permissões, o Amazon Cognito pode usar seus endereços de e-mail verificados no Amazon SES apenas para enviar e-mails aos seus usuários. O Amazon Cognito envia e-mails aos seus usuários quando eles executam determinadas ações na aplicação cliente para um grupo de usuários, como cadastramento ou redefinição de uma senha.

Você deve configurar permissões para que uma entidade do IAM (por exemplo, um usuário, grupo ou função) crie, edite ou exclua um perfil vinculado a serviço. Para obter mais informações, consulte Permissões de perfil vinculado a serviços no Guia do usuário do IAM.

AWSServiceRoleForAmazonCognitoIdp

A função AWSService RoleForAmazonCognitoIdp vinculada ao serviço confia nos seguintes serviços para assumir a função:

- `email.cognito-idp.amazonaws.com`

A política de permissões da função permite que o Amazon Cognito conclua as seguintes ações nos recursos especificados:

Ações permitidas para AWSService RoleForAmazonCognitoIdp

- Ação: `cognito-idp:Describe`

- Recurso: *

Com essa permissão, o Amazon Cognito pode chamar `Describe` operações de API do Amazon Cognito para você.

> **Note**
>
> Quando você integrar o Amazon Cognito ao Amazon Pinpoint usando `createUserPoolClient` e `updateUserPoolClient`, as permissões de recursos serão adicionadas ao SLR como uma política em linha. A política em linha fornecerá permissões `mobiletargeting:UpdateEndpoint` e `mobiletargeting:PutEvents`. Com essas permissões, o Amazon Cognito publica eventos e configura endpoints para projetos do Pinpoint que você integra ao Cognito.

Como criar uma função vinculada a serviço para o Amazon Cognito

Não é necessário criar manualmente um perfil vinculado ao serviço. Quando você configura um grupo de usuários para usar sua configuração do Amazon SES para lidar com a entrega de e-mail na Console de gerenciamento da AWS, na ou na API do Amazon Cognito AWS CLI, o Amazon Cognito cria a função vinculada ao serviço para você.

Se excluir esse perfil vinculado ao serviço e precisar criá-lo novamente, será possível usar esse mesmo processo para recriar o perfil em sua conta. Quando você configura um grupo de usuários para usar sua configuração do Amazon SES para lidar com a entrega de e-mails, o Amazon Cognito cria a função vinculada ao serviço para você novamente.

Antes que o Amazon Cognito possa criar essa função, as permissões do IAM que você usa para configurar o grupo de usuários devem incluir a ação `iam:CreateServiceLinkedRole`. Para obter mais informações sobre como atualizar permissões no IAM, consulte Alterar permissões para um usuário do IAM no Guia do usuário do IAM.

Como editar uma função vinculada a serviço para o Amazon Cognito

Você não pode editar as funções AmazonCognitoIdp ou funções AmazonCognitoIdpEmailService vinculadas ao serviço em. AWS Identity and Access Management Depois que você criar um perfil vinculado ao serviço, não poderá alterar o nome do perfil, pois várias entidades podem fazer referência ao perfil. No entanto, você poderá editar a descrição do perfil usando o IAM. Para saber mais, consulte Editar um perfil vinculado ao serviço no Guia do usuário do IAM.

Como excluir uma função vinculada a serviço para o Amazon Cognito

Se você não precisar mais usar um atributo ou serviço que requer um perfil vinculado ao serviço, é recomendável excluí-lo. Se você excluir a função, só reterá entidades que o Amazon Cognito monitora ou mantém ativamente. Antes de excluir AmazonCognitoIdp ou AmazonCognitoIdpEmailService vincular funções ao serviço, você deve fazer o seguinte para cada grupo de usuários que usa a função:

- Exclua o grupo de usuários.
- Atualize as configurações de e-mail no grupo de usuários para usar a funcionalidade de e-mail padrão. A configuração padrão não usa a função vinculada ao serviço.

Lembre-se de realizar a ação em cada um Região da AWS com um grupo de usuários que usa a função.

> **Note**
>
> Se o serviço do Amazon Cognito 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 um grupo de usuários do Amazon Cognito

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Cognito em. https://console.aws.amazon.com/cognito
2. Selecione Manage User Pools.
3. Na página Your User Pools (Seus grupos de usuários), escolha o grupo de usuários que você quer excluir.
4. Escolha Excluir grupo.
5. Na janela Delete user pool (Excluir grupo de usuários), digite **`delete`** e escolha Delete pool (Excluir grupo).

Para atualizar um grupo de usuários do Amazon Cognito para usar a funcionalidade de e-mail padrão

1. Faça login no Console de gerenciamento da AWS e abra o console do Amazon Cognito em. https://console.aws.amazon.com/cognito
2. Selecione Manage User Pools.
3. Na página Your User Pools (Seus grupos de usuários), escolha o grupo de usuários que você quer atualizar.
4. No menu de navegação à esquerda, escolha Message customizations (Personalizações de mensagens).
5. Em Do you want to send emails through your Amazon SES Configuration? (Deseja enviar e-mails por meio da configuração do Amazon SES?), escolha No - Use Cognito (Default) (Não - Usar o Cognito (padrão)).
6. Quando terminar de definir as opções da conta de e-mail, escolha Save changes (Salvar alterações).

Como excluir manualmente o perfil vinculado ao serviço usando o IAM

Use o console do IAM AWS CLI, o ou a AWS API para excluir funções AmazonCognitoIdp AmazonCognitoIdpEmailService vinculadas ao serviço. Para saber mais, consulte Excluir um perfil vinculado ao serviço no Guia do usuário do IAM.

Regiões compatíveis com funções vinculadas a serviço do Amazon Cognito

O Amazon Cognito oferece suporte a funções vinculadas a serviços em todos os Regiões da AWS lugares em que o serviço está disponível. Para obter mais informações, consulte Regiões da AWS e endpoints.

Como registrar em log e monitorar no Amazon Cognito

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e desempenho do Amazon Cognito e de suas outras AWS soluções. Atualmente, o Amazon Cognito é compatível com os Serviços da AWS a seguir, para que você possa monitorar sua organização e a atividade que acontece dentro dela.

- AWS CloudTrail — Com CloudTrail você pode capturar chamadas de API do console do Amazon Cognito e de chamadas de código para as operações de API do Amazon Cognito. Por exemplo, quando um usuário se autentica, CloudTrail pode registrar detalhes como o endereço IP na solicitação, quem fez a solicitação e quando ela foi feita.

- Amazon CloudWatch Logs — Com o CloudWatch Logs, você pode enviar registros detalhados da atividade do usuário para um grupo de registros. Por exemplo, é possível revisar os logs detalhados de atividades dos usuários para solucionar problemas na entrega de mensagens de e-mail e de SMS aos seus usuários.
- Amazon CloudWatch Metrics — Com CloudWatch métricas, você pode monitorar, relatar e realizar ações automáticas no caso de um evento quase em tempo real. Por exemplo, você pode criar CloudWatch painéis nas métricas fornecidas para monitorar seus grupos de usuários do Amazon Cognito ou CloudWatch criar alarmes nas métricas fornecidas para notificá-lo sobre a violação de um limite definido.
- Amazon CloudWatch Logs Insights — Com o CloudWatch Logs Insights, você pode configurar o CloudTrail envio de eventos CloudWatch para monitorar os arquivos de CloudTrail log do Amazon Cognito.

Tópicos

Monitorar e gerenciar custos

Como com qualquer outro AWS service (Serviço da AWS), é importante entender o efeito da configuração e do uso do Amazon Cognito na sua AWS fatura. Como parte de seus preparativos para a implantação de grupos de usuários na produção, configure o monitoramento e as proteções para o consumo de atividades e recursos. Quando você sabe onde procurar e quais ações geram custos adicionais, pode configurar precauções contra surpresas na fatura.

O Amazon Cognito cobra pelas seguintes dimensões de utilização:

- Grupo de usuários ativos mensais (MAUs) — a taxa varia de acordo com o plano de recursos
- Grupo de usuários MAUs conectado com a federação OIDC ou SAML
- Volume de solicitações para autorização máquina a máquina (M2M) com concessões de credenciais do cliente
- Uso comprado acima das cotas padrão para algumas categorias de grupos de usuários APIs

Além disso, recursos do seu grupo de usuários, como mensagens de e-mail, mensagens SMS e gatilhos Lambda, podem gerar custos em serviços dependentes. Para uma visão geral completa, consulte Preço do Amazon Cognito.

Visualizar e prever custos

Eventos de alto volume, como lançamentos de produtos e abertura para novas bases de usuários, podem aumentar sua contagem de MAUs e afetar os custos. Estime a contagem de novos usuários com antecedência e observe regularmente as atividades. Você pode acomodar o volume com a compra de capacidade de cota adicional ou controlar o volume com medidas de segurança adicionais.

Você pode visualizar e relatar seus AWS custos no Gerenciamento de Faturamento e Custos da AWS console. As cobranças mais recentes do Amazon Cognito estão na seção Faturamento e pagamentos. Em Faturas, Cobranças por serviço, filtre o `Cognito` para ver a utilização. Para obter mais informações, consulte Exibição da sua fatura no Guia do usuário do AWS Billing .

Para monitorar as taxas de solicitação de API, consulte a métrica Utilização no console do Service Quotas. Por exemplo, as solicitações de credenciais do cliente são exibidas como Taxa de ClientAuthentication solicitações. Na fatura, essas solicitações estão associadas ao cliente da aplicação que as produziu. Com essas informações, você pode alocar custos de forma equitativa aos locatários em uma arquitetura de multilocatários.

Para obter uma contagem das solicitações M2M por um período de tempo, você também pode enviar AWS CloudTrail eventos ao CloudWatch Logs para análise. Consulte seus CloudTrail eventos para `Token_POST` eventos com uma concessão de credenciais de cliente. A consulta do CloudWatch Insights a seguir retorna essa contagem.

```
filter eventName = "Token_POST" and @message like '"grant_type":["client_credentials"]'
 | stats count(*)
```

Gerenciar custos

O Amazon Cognito cobra com base na contagem de usuários, no uso de recursos e no volume de solicitações. Veja a seguir algumas dicas para gerenciar custos no Amazon Cognito:

Não ative usuários inativos

Operações típicas que tornam um usuário ativo são login, inscrição e redefinição de senha. Para obter uma lista mais completa, consulte Usuários ativos mensalmente. O Amazon Cognito não

contabiliza usuários inativos na fatura. Evite qualquer operação que deixe um usuário ativo. Em vez da operação AdminGetUserda API, consulte os usuários com a ListUsersoperação. Não faça testes administrativos de alto volume de operações de grupos de usuários com usuários inativos.

Vincule usuários federados

Usuários que fazem login com um provedor de identidades SAML 2.0 ou OpenID Connect (OIDC) têm um custo mais alto do que usuários locais. Você pode vincular esses usuários a um perfil de usuário local. Um usuário vinculado pode fazer login como usuário local com os atributos e o acesso fornecidos com seu usuário federado. Os usuários do SAML ou do OIDC IdPs que, ao longo de um mês, só fazem login com uma conta local vinculada são cobrados como usuários locais.

Gerencie as taxas de solicitação

Se seu grupo de usuários estiver se aproximando do limite máximo de sua cota, recomenda-se comprar capacidade adicional para lidar com o volume. Talvez você consiga reduzir o volume de solicitações na aplicação. Para obter mais informações, consulte Otimizar as taxas de solicitação para limites de cota.

Solicite um novo token somente quando precisar de um

A autorização máquina a máquina (M2M) com concessões de credenciais de clientes pode atingir um alto volume de solicitações de token. Cada nova solicitação de token afeta sua cota de taxa de solicitação e o tamanho da fatura. Para otimizar o custo, inclua configurações de expiração e tratamento de tokens no design de suas aplicações.

- Armazene os tokens de acesso em cache para que, quando sua aplicação solicitar um novo token, ela receba uma versão em cache de um token emitido anteriormente. Quando você implementa esse método, seu proxy de cache age como uma proteção contra aplicações que solicitam tokens de acesso sem conhecer a expiração dos tokens adquiridos anteriormente. O armazenamento em cache de tokens é ideal para microsserviços de curta duração, como funções do Lambda e contêineres do Docker.
- Implemente mecanismos de tratamento de tokens em suas aplicações que levem em conta a expiração do token. Não solicite um novo token até que os tokens anteriores estejam prestes a expirar. A prática recomendada é atualizar os tokens em cerca de 75% da vida útil do token. Essa prática maximiza a duração do token e, ao mesmo tempo, garante a continuidade do usuário em sua aplicação.

  Avalie as necessidades de confidencialidade e disponibilidade de cada aplicação e configure o cliente da aplicação do grupo de usuários para emitir tokens de acesso com um período de

validade apropriado. A duração personalizada do token funciona melhor com servidores de longa duração APIs que podem gerenciar persistentemente a frequência das solicitações de credenciais.

ListUsers, não AdminGetUser

Para consultar os atributos dos usuários em seu grupo de usuários, use a operação da ListUsersAPI e os métodos SDK associados sempre que possível. AdminGetUsermarca um usuário como ativo no mês e contribui para os usuários ativos mensais (MAUs) que são usados para calcular sua fatura por grupos de usuários.

Gerenciar planos de recursos

Quando você escolhe um plano de recursos em um grupo de usuários, a taxa de cobrança se aplica a todos MAUs no grupo de usuários. Se você tiver usuários que não precisam de recursos que vêm com um plano de recursos de nível superior, separe-os em outro grupo de usuários.

Exportação de logs dos grupos de usuários do Amazon Cognito

Você pode configurar seu grupo de usuários para enviar registros detalhados de alguma atividade adicional para outra pessoa AWS service (Serviço da AWS), como um grupo de CloudWatch registros. Esses registros têm uma granularidade mais fina do que os registrados e podem ser úteis para solucionar problemas do grupo de usuários e analisar a atividade de login do usuário com proteção contra ameaças. AWS CloudTrail Quando você quiser transmitir registros de erros de notificação por SMS e e-mail, seu grupo de usuários envia registros em `ERROR` nível de CloudWatch registro para um grupo de registros. Quando você quiser transmitir logs da atividade de login do usuário, seu grupo de usuários enviará logs em nível de `INFO` para um grupo de logs, um stream do Amazon Data Firehose ou um bucket do Amazon S3. É possível combinar as duas opções em um grupo de usuários.

Tópicos

- O que é importante saber sobre exportação de logs
- Exportar erros de entrega de e-mails e mensagens SMS
- Exportar logs de atividade de usuários de proteção contra ameaças

O que é importante saber sobre exportação de logs

Impacto do custo

O Amazon Data Firehose, o Amazon S3 e o Logs incorrem em custos de ingestão CloudWatch e recuperação de dados. Sua configuração de registro pode afetar sua AWS fatura. Para saber mais, consulte:

- Logs vendidos nos CloudWatch preços da Amazon.
- Definição de preços do Amazon Data Firehose
- Definição de preços do Amazon S3

As exportações de logs de atividades do usuário contêm avaliações de segurança e são uma função da proteção contra ameaças do grupo de usuários. O Amazon Cognito só gera esses logs quando a proteção contra ameaças está no modo Somente auditoria ou Função completa e o grupo de usuários está no plano de recursos Plus.

Logs de atividades do usuário são de nível **`INFO`**

Os logs de atividade do usuário exportados estão somente no nível de erro `INFO` e fornecem informações para análise estatística e de segurança da atividade de autenticação. Mensagens nos níveis de erro `WARNING` e `ERROR`, por exemplo, erros de controle de utilização, não estão incluídas nos logs exportados.

Entrega do melhor esforço

A entrega de logs do Amazon Cognito é o melhor esforço. O volume de registros que seu grupo de usuários fornece e suas cotas de serviço para CloudWatch Logs, Amazon S3 e Firehose podem afetar a entrega de registros.

Os logs externos existentes não são afetados

Essas opções de logs não substituem nem alteram as seguintes funções de log dos grupos de usuários:

1. CloudTrail registros de atividades rotineiras do usuário, como inscrição e login.
2. Análise da atividade do usuário em grande escala com CloudWatch métricas.

Separadamente, você também pode encontrar registros de Visualizando os resultados da importação do grupo de usuários no CloudWatch console e Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda em CloudWatch Registros. O Amazon Cognito e o Lambda armazenam esses logs em grupos de logs diferentes daqueles que você especifica para logs de atividades do usuário.

Aplica-se somente aos grupos de usuários

Não existem recursos de exportação de logs para bancos de identidades.

Requer permissões de usuário e perfil vinculado ao serviço

O AWS diretor que configura a exportação de registros deve ter permissões para modificar os recursos de destino, conforme descrito nos tópicos a seguir. O Amazon Cognito cria um perfil vinculado ao serviço em seu nome e assume a função de entregar logs ao recurso de destino.

Para obter mais informações sobre o modelo de autorização para envio de registros do Amazon Cognito, consulte Habilitar o registro Serviços da AWS no Guia do usuário do Amazon CloudWatch Logs.

O nível de log é exclusivo para o tipo de log

Os logs de entrega de mensagens são do tipo `userNotification` e do nível de erro `ERROR`. Os logs de atividades do usuário de segurança avançada são do tipo `userAuthEvents` e do nível de erro `INFO`. Você pode combinar dois membros do`LogConfigurations`, um `userNotification` para CloudWatch Logs e outro `userAuthEvents` para Firehose, Amazon S3 ou Logs. CloudWatch

Você não pode enviar logs de atividades do usuário para vários destinos. Você não pode enviar registros de notificação do usuário para nenhum destino que não seja o CloudWatch Logs.

Opções de configuração diferentes

Você só pode configurar logs de notificação de usuário com a API de grupos de usuários do Amazon Cognito ou um AWS SDK. Você pode configurar logs de atividades do usuário de segurança avançada com a API ou no console do Amazon Cognito. Para definir ambos, use a API conforme demonstrado na solicitação de exemplo em SetLogDeliveryConfiguration.

Configuração adicional necessária com grandes políticas baseadas em recursos

Para enviar logs a grupos de logs com uma política de recursos de tamanho maior que 5120 caracteres, configure um grupo de logs com um caminho que comece com `/aws/vendedlogs`. Para obter mais informações, consulte Habilitar o registro de determinados AWS serviços.

Criação automática de uma pasta no Amazon S3

Ao configurar a exportação do log de proteção contra ameaças para um bucket do Amazon S3, o Amazon Cognito pode criar uma pasta `AWSLogs` no seu bucket. Essa pasta não é criada em todos os casos, e a configuração pode ser bem-sucedida mesmo sem sua criação.

Exportar erros de entrega de e-mails e mensagens SMS

Para erros de entrega de mensagens de e-mail e SMS, você pode entregar logs de notificação de usuário no nível de Erro do grupo de usuários. Ao ativar esse atributo, é possível escolher o grupo de logs para o qual você deseja que o Amazon Cognito envie logs. O log de notificações do usuário é útil quando você deseja descobrir o status das mensagens de e-mail e SMS que o grupo de usuários entregou com o Amazon SNS e o Amazon SES. Essa opção de exportação de log, diferentemente da exportação de atividades do usuário, não exige o plano de recursos Plus.

Você pode configurar registros de notificação detalhados com a API de grupos de usuários do Amazon Cognito em uma solicitação de SetLogDeliveryConfigurationAPI. Você pode ver a configuração de registro de um grupo de usuários em uma solicitação de GetLogDeliveryConfigurationAPI. Veja a seguir um exemplo de corpo da solicitação .

```
{
   "LogConfigurations": [
      {
         "CloudWatchLogsConfiguration": {
            "LogGroupArn": "arn:aws:logs:us-west-2:123456789012:log-group:example-user-
pool-exported"
         },
         "EventSource": "userNotification",
         "LogLevel": "ERROR"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Você deve autorizar essas solicitações com AWS credenciais que tenham as seguintes permissões.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
            {
            "Sid": "ManageUserPoolLogs",
            "Action": [
                "cognito-idp:SetLogDeliveryConfiguration",
                "cognito-idp:GetLogDeliveryConfiguration"
            ],
```

```
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Sid": "CognitoLog",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Sid": "CognitoLoggingCWL",
            "Action": [
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

Veja a seguir um exemplo de evento de um grupo de usuários. Esse esquema de logs está sujeito a alterações. Alguns campos podem ser registrados em log com valores nulos.

```
{
    "eventTimestamp": "1687297330677",
    "eventSource": "USER_NOTIFICATION",
    "logLevel": "ERROR",
    "message": {
```

```
        "details": "String"
    },
    "logSourceId": {
        "userPoolId": "String"
    }
}
```

Exportar logs de atividade de usuários de proteção contra ameaças

Grupos de usuários com o plano de recursos Plus e proteção contra ameaças registram em log os eventos de atividade de usuários: os detalhes e a avaliação de segurança do login e logout do usuário e de outras operações de autenticação com seu grupo de usuários. Talvez você queira revisar os logs de atividades do usuário em seu próprio sistema de gerenciamento de logs ou criar um arquivo. Você pode exportar esses dados para um grupo de CloudWatch logs do Amazon Logs, um stream do Amazon Data Firehose ou um bucket do Amazon Simple Storage Service (Amazon S3). A partir daí, você pode ingerir esses dados em outros sistemas que analisam, normalizam ou processam dados de forma a ajustá-los aos seus processos operacionais. Para exportar dados desse tipo, o grupo de usuários deve estar no plano de recursos Plus e a proteção contra ameaças deve estar ativa no grupo de usuários.

Com as informações nesses logs de atividades do usuário, você pode ver um perfil das atividades de login e gerenciamento de contas do usuário. Por padrão, o Amazon Cognito captura esses eventos para um armazenamento baseado em seu grupo de usuários. O exemplo a seguir é de um evento de um usuário que fez login e recebeu uma avaliação de ausência de fatores de risco. Você pode recuperar essas informações com a operação da API `AdminListUserAuthEvents`. Veja a seguir um exemplo de saída:

```
{
    "AuthEvents": [
        {
            "EventId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "EventType": "SignIn",
            "CreationDate": "2024-06-27T10:49:59.139000-07:00",
            "EventResponse": "Pass",
            "EventRisk": {
                "RiskDecision": "NoRisk",
                "CompromisedCredentialsDetected": false
            },
            "ChallengeResponses": [
                {
```

```
                    "ChallengeName": "Password",
                    "ChallengeResponse": "Success"
                }
            ],
            "EventContextData": {
                "IpAddress": "192.0.2.1",
                "DeviceName": "Chrome 126, Windows 10",
                "Timezone": "-07:00",
                "City": "null",
                "Country": "United States"
            }
        }
    ],
    "NextToken": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222#2024-06-27T17:49:59.139Z"
}
```

Você pode ativar a exportação de log para a atividade do usuário no console do Amazon Cognito ou com a operação da SetLogDeliveryConfigurationAPI.

Console de gerenciamento da AWS

1. Se você ainda não tem um que queira usar, crie um bucket do S3, um stream do Firehose CloudWatch ou um grupo de registros.
2. Faça login no console do Amazon Cognito.
3. Escolha User Pools (Grupos de usuários).
4. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
5. Escolha a guia Segurança avançada. Localize Exportar logs de atividades do usuário e escolha Editar
6. Em Status de registro em log, marque a caixa de seleção ao lado de Ativar exportação do log de atividades do usuário.
7. Em Logging destination, escolha o AWS service (Serviço da AWS) que você deseja manipular com seus registros: grupo de CloudWatch registros, stream do Amazon Data Firehose ou bucket do S3.
8. Sua seleção preencherá o seletor de recursos com o tipo de recurso correspondente. Selecione um grupo de logs, stream ou bucket na lista. Você também pode selecionar o botão Criar para navegar até o serviço selecionado e criar um novo recurso. Console de gerenciamento da AWS
9. Selecione Salvar alterações.

API

Escolha um tipo de destino para seus logs de atividades do usuário.

Veja a seguir um exemplo de corpo de solicitação `SetLogDeliveryConfiguration` que define um stream do Firehose como o destino do log.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "FirehoseConfiguration": {
            "StreamArn": "arn:aws:firehose:us-west-2:123456789012:deliverystream/
example-user-pool-activity-exported"
         },
         "LogLevel": "INFO"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Veja a seguir um exemplo de corpo de solicitação `SetLogDeliveryConfiguration` que define um bucket do Amazon S3 como o destino do log.

```
{
   "LogConfigurations": [
      {
         "EventSource": "userAuthEvents",
         "S3Configuration": {
            "BucketArn": "arn:aws:s3:::amzn-s3-demo-logging-bucket"
         },
         "LogLevel": "INFO"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

Veja a seguir um exemplo de corpo de `SetLogDeliveryConfiguration` solicitação que define um grupo de CloudWatch registros como o destino do registro.

```
{
   "LogConfigurations": [
```

```
      {
         "EventSource": "userAuthEvents",
         "CloudWatchLogsConfiguration": {
            "LogGroupArn": "arn:aws:logs:us-west-2:123456789012:log-group:DOC-
EXAMPLE-LOG-GROUP"
         },
         "LogLevel": "INFO"
      }
   ],
   "UserPoolId": "us-west-2_EXAMPLE"
}
```

O usuário que configura a entrega de logs deve ser administrador do grupo de usuários e ter as seguintes permissões adicionais:

Amazon S3

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageUserPoolLogs",
            "Action": [
                "cognito-idp:SetLogDeliveryConfiguration",
                "cognito-idp:GetLogDeliveryConfiguration"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Sid": "ManageLogsS3",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogDelivery",
                "s3:PutBucketPolicy",
                "s3:GetBucketPolicy"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

CloudWatch Logs

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageUserPoolLogs",
            "Action": [
                "cognito-idp:SetLogDeliveryConfiguration",
                "cognito-idp:GetLogDeliveryConfiguration"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Sid": "ManageLogsCWL",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries",
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        }
    ]
```

```
}
```

Amazon Data Firehose

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ManageUserPoolLogs",
            "Action": [
                "cognito-idp:SetLogDeliveryConfiguration",
                "cognito-idp:GetLogDeliveryConfiguration"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Sid": "ManageUserPoolLogsFirehose",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogDelivery",
                "iam:CreateServiceLinkedRole",
                "firehose:TagDeliveryStream"
            ],
            "Resource": "*"
        }
    ]
}
```

Veja a seguir um exemplo de evento de um grupo de usuários. Esse esquema de logs está sujeito a alterações. Alguns campos podem ser registrados em log com valores nulos.

```
{
    "eventTimestamp": "1687297330677",
    "eventSource": "USER_ACTIVITY",
    "logLevel": "INFO",
```

```
    "message": {
        "version": "1",
        "eventId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "eventType": "SignUp",
        "userSub": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "userName": "test-user",
        "userPoolId": "us-west-2_EXAMPLE",
        "clientId": "1example23456789",
        "creationDate": "Wed Jul 17 17:25:55 UTC 2024",
        "eventResponse": "InProgress",
        "riskLevel": "",
        "riskDecision": "PASS",
        "challenges": [],
        "deviceName": "Other, Other",
        "ipAddress": "192.0.2.1",
        "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
        "idpName": "",
        "compromisedCredentialDetected": "false",
        "city": "Seattle",
        "country": "United States",
        "eventFeedbackValue": "",
        "eventFeedbackDate": "",
        "eventFeedbackProvider": "",
        "hasContextData": "true"
    },
    "logSourceId": {
        "userPoolId": "us-west-2_EXAMPLE"
    }
}
```


Rastreamento de cotas e uso em CloudWatch e Service Quotas

Você pode monitorar grupos de usuários do Amazon Cognito usando a Amazon CloudWatch ou usando Cotas de Serviço. Você também pode monitorar o uso de grupos de identidades em Service Quotas. CloudWatch coleta dados brutos e os processa em métricas legíveis, quase em tempo real. Em CloudWatch, você pode definir alarmes que observem determinados limites e enviar notificações ou realizar ações quando esses limites forem atingidos. Para criar um CloudWatch alarme para uma cota de serviço, consulte Criar um CloudWatch alarme. As métricas do Amazon Cognito são disponibilizadas em intervalos de cinco minutos. Para obter mais informações sobre períodos de retenção em CloudWatch, visite a página de CloudWatch perguntas frequentes da Amazon.

É possível utilizar o Service Quotas para visualizar e gerenciar a utilização de cotas de grupos de usuários e de bancos de identidades do Amazon Cognito. O console do Service Quotas tem três recursos: visualizar cotas de serviço, solicitar um aumento da cota de serviço e visualizar a utilização atual. É possível usar o primeiro recurso para visualizar cotas e ver se a cota é ajustável. Você pode usar o segundo recurso para solicitar um aumento do Service Quotas. Você pode usar o último recurso para visualizar a utilização da cota. Esse recurso só estará disponível depois que sua conta estiver ativa por algum tempo. Para obter mais informações sobre como visualizar cotas no console do Service Quotas, consulte Visualizar Service Quotas.

> **Note**
>
> As métricas do Amazon Cognito são disponibilizadas em intervalos de cinco minutos. Para obter mais informações sobre períodos de retenção em CloudWatch, visite a página de CloudWatch perguntas frequentes da Amazon.

Se você estiver conectado a uma Conta da AWS conta configurada como uma conta de monitoramento na observabilidade CloudWatch entre contas, poderá usar essa conta de monitoramento para visualizar cotas de serviço e definir alarmes para métricas nas contas de origem vinculadas a essa conta de monitoramento. Para obter mais informações, consulte Observabilidade entre contas do CloudWatch .

Tópicos

- Métricas do grupo de usuários em CloudWatch
- Métricas no Service Quotas

Métricas do grupo de usuários em CloudWatch

Os grupos de usuários relatam as estatísticas da atividade do usuário CloudWatch como métricas. A partir de CloudWatch, você pode analisar o volume da atividade de autenticação e o uso da cota em seus grupos de usuários. Com as informações nessas métricas, você pode definir alarmes para eventos que merecem destaque e ajustar a configuração do grupo de usuários conforme necessário. Onde o registro de atividades do usuário tem registros detalhados da atividade do usuário em seus grupos de usuários, CloudWatch as métricas têm estatísticas agregadas e indicadores de desempenho.

A tabela a seguir lista as métricas disponíveis para grupos de usuários do Amazon Cognito. O Amazon Cognito publica métricas nos namespaces `AWS/Cognito` e `AWS/Usage`. Para obter mais informações, consulte Namespaces no Guia CloudWatch do usuário da Amazon.

Para obter mais informações sobre rastreamento de cotas e uso, consulte Rastrear o uso da cota e Rastreie usuários ativos mensais (MAUs).

> **Note**
>
> As métricas que não tiverem tido novos pontos de dados nas últimas duas semanas não serão exibidas no console. Elas também não são exibidas quando você insere o nome da métrica ou os nomes de dimensão na caixa de pesquisa na guia All metrics (Todas as métricas) no console. Além disso, elas não são retornados nos resultados de um comando list-metrics. A melhor maneira de recuperar essas métricas é com os `get-metric-statistics` comandos `get-metric-data` or na AWS CLI.

Métrica	Description	Namespace
`SignUpSuccesses`	Fornece o número total de solicitações bem-sucedidas de registro de usuário feitas ao grupo de usuários do Amazon Cognito. Uma solicitação de registro de usuário bem-sucedida produz um valor de 1, enquanto uma solicitaç ão malsucedida produz um valor de 0. Uma solicitação com controle de utilização também é considerada uma solicitação malsucedida e, portanto, também produzirá uma contagem de 0. Para descobrir a porcentag em de solicitações de registro de usuário bem-sucedidas,	`AWS/Cognito`

Métrica	Description	Namespace
	use a estatística `Average` nessa métrica. Para contar o número total de solicitações de registro de usuário, use a estatística `Sample Count` nessa métrica. Para contar o número total de solicitaç ões de registro de usuário bem-sucedidas, use a estatísti ca `Sum` nessa métrica. Para contar o número total de solicitações de registro de usuários que falharam, use a CloudWatch `Math` expressão e subtraia a `Sum` estatística da `Sample Count` estatística. Essa métrica é publicada por grupo de usuários, para cada cliente de grupo de usuários. Caso o registro de usuário seja realizado por um administrador, a métrica será publicada com o cliente de grupo de usuários como `Admin`. Observe que essa métrica não é emitida para casos de Importação de usuários e Migração de usuários. Dimensão da métrica: `UserPool`, `UserPoolC lient`	

Métrica	Description	Namespace
	Unidades: contagem	
`SignUpThrottles`	Fornece o número total de solicitações com controle de utilização de registro de usuário feitas ao grupo de usuários do Amazon Cognito. Uma contagem de 1 é publicada sempre que uma solicitação de registro de usuário é limitada. Para contar o número total de solicitações de registro de usuário limitadas, use a estatística `Sum` para essa métrica. Essa métrica é publicada para cada grupo de usuários para cada cliente. Caso a solicitação com controle de utilização tenha sido feita por um administrador, a métrica será publicada com o cliente de grupo de usuários como `Admin`. Dimensão da métrica: `UserPool`, `UserPoolClient` Unidades: contagem	`AWS/Cognito`

Métrica	Description	Namespace
`SignInSuccesses`	Fornece o número total de solicitações bem-sucedidas de autenticação de usuário feitas ao grupo de usuários do Amazon Cognito. Uma autenticação de usuário é considerada bem-sucedida quando o token de autentica ção é emitido para o usuário. Uma autenticação bem-sucedida produz um valor de 1, enquanto uma solicitaç ão malsucedida produz um valor de 0. Uma solicitação com controle de utilização também é considerada uma solicitação malsucedida e, portanto, também produzirá uma contagem de 0. Para descobrir a porcentag em de solicitações de autenticação de usuário bem-sucedidas, use a estatísti ca `Average` nessa métrica. Para contar o número total de solicitações de autentica ção de usuário, use a estatísti ca `Sample Count` nessa métrica. Para contar o número total de solicitações de autenticação de usuário bem-sucedidas, use a estatísti ca `Sum` nessa métrica. Para contar o número total de	`AWS/Cognito`

Métrica	Description	Namespace
	solicitações de autenticação de usuário com falha, use a CloudWatch `Math` expressão e subtraia a `Sum` estatística da `Sample Count` estatística. Essa métrica é publicada para cada grupo de usuários para cada cliente. Caso um cliente de grupo de usuários inválido seja fornecido com uma solicitação, o valor de cliente de grupo de usuários correspondente na métrica conterá um valor fixo `Invalid` em vez do valor inválido real enviado na solicitação. Observe que as solicitações para atualizar o token do Amazon Cognito não estão incluídas nessa métrica. Há uma métrica separada para fornecer estatísticas de token `Refresh`. Dimensão da métrica: `UserPool`, `UserPoolClient` Unidades: contagem	

Métrica	Description	Namespace
`SignInThrottles`	Fornece o número total de solicitações com controle de utilização de autenticação de usuário feitas ao grupo de usuários do Amazon Cognito. Uma contagem de 1 é publicada sempre que uma solicitação de autenticação de usuário é limitada. Para contar o número total de solicitações de autentica ção de usuário limitadas, use a estatística `Sum` para essa métrica. Essa métrica é publicada para cada grupo de usuários para cada cliente. Caso um cliente de grupo de usuários inválido seja fornecido com uma solicitação, o valor de cliente de grupo de usuários correspondente na métrica conterá um valor fixo `Invalid` em vez do valor inválido real enviado na solicitação. Solicitações para atualizar o token do Amazon Cognito não estão incluídas nessa métrica. Há uma métrica separada para fornecer estatísticas de token `Refresh`.	`AWS/Cognito`

Métrica	Description	Namespace
	Dimensão da métrica: `UserPool`, `UserPoolClient` Unidades: contagem	

Métrica	Description	Namespace
`TokenRefreshSuccesses`	Fornece o número total de solicitações bem-sucedidas para atualizar um token do Amazon Cognito que foram feitas ao grupo de usuários do Amazon Cognito. Uma solicitação bem-sucedida de atualização do token do Amazon Cognito produz um valor de 1, enquanto uma solicitação malsucedida produz um valor de 0. Uma solicitação com controle de utilização também é considerada uma solicitação malsucedida e, portanto, também produzirá uma contagem de 0. Para descobrir a porcentagem de solicitações bem-sucedidas de atualização de um token do Amazon Cognito, use a estatística `Average` nessa métrica. Para contar o número total de solicitações de atualização de um token do Amazon Cognito, use a estatística `Sample Count` nessa métrica. Para contar o número total de solicitações bem-sucedidas de atualização de um token do Amazon Cognito, use a estatística `Sum` nessa métrica. Para contar o número total de solicitações	`AWS/Cognito`

Métrica	Description	Namespace
	malsucedidas para atualizar um token do Amazon Cognito, use CloudWatch `Math` a expressão e subtraia `Sum` a estatística da estatística. `Sample Count` Essa métrica é publicada para cada cliente de grupo de usuários. Se o cliente de um grupo de usuários inválido estiver em uma solicitação, o valor do cliente do grupo de usuários conterá um valor fixo de `Invalid`. Dimensão da métrica: `UserPool`, `UserPoolClient` Unidades: contagem	

Métrica	Description	Namespace
`TokenRefreshThrottles`	Fornece o número total de solicitações com controle de utilização de atualização de um token do Amazon Cognito que foram feitas para o grupo de usuários do Amazon Cognito. Uma contagem de 1 é publicada sempre que uma solicitação de atualização de token do Amazon Cognito tem controle de utilização. Para contar o número total de solicitações com controle de utilização para atualizar um token do Amazon Cognito, use a estatística `Sum` para essa métrica. Essa métrica é publicada para cada grupo de usuários para cada cliente. Caso um cliente de grupo de usuários inválido seja fornecido com uma solicitação, o valor de cliente de grupo de usuários correspondente na métrica conterá um valor fixo `Invalid` em vez do valor inválido real enviado na solicitação. Dimensão da métrica: `UserPool`, `UserPoolClient`	`AWS/Cognito`

Métrica	Description	Namespace
	Unidades: contagem	

Métrica	Description	Namespace
`FederationSuccesses`	Fornece o número total de solicitações bem-sucedidas de federação de identidades feitas ao grupo de usuários do Amazon Cognito. Uma federação de identidades é considerada bem-sucedida quando o Amazon Cognito emite tokens de autenticação para o usuário. Uma solicitação de federação de identidades bem-sucedida produz um valor de 1, enquanto uma solicitação malsucedida produz um valor de 0. Solicitações com controle de utilização e solicitações que geram um código de autorização, mas nenhum token, produzem um valor de 0. Para descobrir a porcentagem de solicitações de federação de identidades bem-sucedidas, use a estatística `Average` nessa métrica. Para contar o número total de solicitações de federação de identidades, use a estatística `Sample Count` nessa métrica. Para contar o número total de solicitações bem-sucedidas de federação de identidades, use a estatística `Sum` nessa métrica. Para	`AWS/Cognito`

Métrica	Description	Namespace
	contar o número total de solicitações de federação de identidade com falha, use a CloudWatch `Math` expressão e subtraia a `Sum` estatística da `Sample Count` estatística. Dimensão de métrica: `UserPool`, `UserPoolClient` , `IdentityProvider` Unidades: contagem	
`FederationThrottles`	Fornece o número total de solicitações limitadas de federação de identidades feitas ao grupo de usuários do Amazon Cognito. Uma contagem de 1 é publicada sempre que uma solicitação de federação de identidades tem controle de utilização. Para contar o número total de solicitações de federação de identidades limitadas, use a estatística `Sum` para essa métrica. Dimensão de métrica: `UserPool`, `UserPoolClient` , `IdentityProvider` Unidades: contagem	`AWS/Cognito`

Métrica	Description	Namespace
`CallCount`	Fornece o número total de chamadas feitas pelos clientes em relação a uma categoria . Essa métrica inclui todas as chamadas, como chamadas com controle de utilizaçã o, chamadas com falha e chamadas bem-sucedidas. A cota de categoria é aplicada para cada AWS conta em todos os grupos de usuários em uma conta e região. Você pode contar o número total de chamadas em uma categoria usando a estatística `Sum` para essa métrica. Dimensão métrica: Serviço, Tipo, Recurso, Classe Unidades: contagem	`AWS/Usage`

Métrica	Description	Namespace
`ThrottleCount`	Fornece o número total de chamadas com controle de utilização relacionadas a uma categoria. Essa métrica é publicada no nível da conta. Você pode contar o número total de chamadas em uma categoria usando a estatística `Sum` para essa métrica. Dimensão métrica: Serviço, Tipo, Recurso, Classe Unidades: contagem	`AWS/Usage`

Como exibir métricas de proteção contra ameaças

As métricas que seu grupo de usuários publica têm informações estatísticas sobre o efeito que as configurações de proteção contra ameaças têm na atividade de autenticação do usuário. Talvez você queira saber quantos usuários estão tentando entrar com credenciais comprometidas. Você também pode descobrir qual porcentagem da atividade de login foi avaliada por conter certo nível de risco. O Amazon Cognito publica métricas para recursos de proteção contra ameaças em sua conta na Amazon. CloudWatch O Amazon Cognito agrupa as métricas de proteção contra ameaças com base nos níveis de risco e no nível de solicitação.

Para adicionar contexto à sua análise de risco, você pode visualizar informações sobre tentativas individuais de login de usuários tanto no grupo de usuários como em uma fonte de dados exportada.

Para visualizar métricas no CloudWatch console

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, selecione Métricas.
3. Escolha Amazon Cognito.

4. Escolha um grupo de métricas agregadas, como By Risk Classification (Por classificação de risco).
5. A guia All metrics (Todas as métricas) exibe todas as métricas da opção escolhida. Você pode fazer o seguinte:
   - Para classificar a tabela, use o cabeçalho da coluna.
   - Para criar um gráfico de uma métrica, marque a caixa de seleção ao lado da métrica. Para selecionar todas as métricas, marque a caixa de seleção na linha de cabeçalho da tabela.
   - Para filtrar por recurso, escolha o ID do recurso e Add to search (Adicionar à pesquisa).
   - Para filtrar por métrica, escolha o nome da métrica e Add to search (Adicionar à pesquisa).

Métrica	Description	Dimensões da métrica	Namespace
CompromisedCredentialRisk	Solicitações em que o Amazon Cognito detectou credenciais comprometidas.	Operação: o tipo de operação. `PasswordChange`, `SignIn`, ou `SignUp` são as únicas dimensões. UserPoolId: o identificador do grupo de usuários. RiskLevel: alto (padrão), médio ou baixo.	`AWS/Cognito`
AccountTakeoverRisk	Solicitações em que o Amazon Cognito detectou risco de tomada de controle da conta.	Operação: o tipo de operação. `PasswordChange`, `SignIn`, ou `SignUp` são as únicas dimensões.	`AWS/Cognito`

Métrica	Description	Dimensões da métrica	Namespace
		UserPoolId: o identificador do grupo de usuários. RiskLevel: alto, médio ou baixo.	
OverrideBlock	Solicitações que o Amazon Cognito bloqueou por causa da configuração fornecida pelo desenvolvedor.	Operação: o tipo de operação. `PasswordChange` , `SignIn`, ou `SignUp` são as únicas dimensões. UserPoolId: o identificador do grupo de usuários. RiskLevel: alto, médio ou baixo.	`AWS/Cognito`
Risco	Solicitações que o Amazon Cognito marcou como arriscadas.	Operation: o tipo de operação, como `PasswordChange` , `SignIn` ou `SignUp`. UserPoolId: o identificador do grupo de usuários.	`AWS/Cognito`

Métrica	Description	Dimensões da métrica	Namespace
NoRisk	Solicitações em que o Amazon Cognito não identificou qualquer risco.	Operation: o tipo de operação, como `PasswordChange` , `SignIn` ou `SignUp`. UserPoolId: o identificador do grupo de usuários.	`AWS/Cognito`

O Amazon Cognito oferece dois grupos predefinidos de métricas para análise pronta. CloudWatch By Risk Classification (Por classificação de risco) identifica a granularidade do nível de risco para solicitações que o Amazon Cognito identifica como arriscadas. By Request Classification (Por classificação de solicitação) reflete métricas agregadas pelo nível de solicitação.

Grupo de métricas agregadas	Description
Por classificação de risco	Solicitações que o Amazon Cognito identifica como arriscadas.
Por classificação de solicitação	Métricas agregadas por solicitação.

Dimensões dos grupos de usuários do Amazon Cognito

As dimensões a seguir são usadas para refinar as métricas de uso publicadas pelo Amazon Cognito. As dimensões só se aplicam às métricas `CallCount` e `ThrottleCount` .

Dimensão	Descrição
Serviço	O nome do AWS serviço que contém o recurso. Para as métricas de uso do Amazon Cognito, o valor dessa dimensão é `Cognito user pool`.

Dimensão	Descrição
Tipo	O tipo de entidade que está sendo relatado. O único valor válido para métricas de uso do Amazon Cognito é API.
Recurso	O tipo de recurso que está em execução. O único valor válido é o nome da categoria.
Classe	A classe do recurso sob acompanhamento. O Amazon Cognito não usa a dimensão de classe.

Use o CloudWatch console para monitorar métricas

Você pode rastrear e coletar métricas de grupos de usuários do Amazon Cognito usando. CloudWatch O CloudWatch painel exibirá métricas sobre cada AWS serviço que você usa. Você pode usar CloudWatch para criar alarmes métricos. Os alarmes podem ser configurados para enviar a você notificações ou alterar um recurso específico que você está monitorando. Para visualizar as métricas da cota de serviço em CloudWatch, conclua as etapas a seguir.

1. Abra o console do CloudWatch.
2. No painel de navegação, escolha Metrics (Métricas).
3. Em All metrics (Todas as métricas), selecione uma métrica e uma dimensão.
4. Marque a caixa de seleção ao lado de uma métrica. As métricas serão exibidas no gráfico.

> **Note**
>
> As métricas que não tiverem tido novos pontos de dados nas últimas duas semanas não serão exibidas no console. Elas também não serão exibidas quando você digitar o nome da métrica ou os nomes de dimensão na caixa de pesquisa na guia All metrics (Todas as métricas) do console e não serão retornadas nos resultados de um comando list-metrics. A melhor maneira de recuperar essas métricas é com os comandos `get-metric-data` ou `get-metric-statistics` na CLI da AWS .

Crie um CloudWatch alarme para uma cota

O Amazon Cognito fornece métricas CloudWatch de uso que correspondem às cotas AWS de serviço para e. `CallCount` `ThrottleCount` APIs Para obter mais informações sobre o rastreamento do uso em CloudWatch, consulteRastrear o uso da cota.

No console do Service Quotas, é possível criar alarmes que alertarão você quando o uso se aproximar de uma cota de serviço. Para saber como configurar um CloudWatch alarme usando o console Service Quotas, consulte Service Quotas e alarmes. CloudWatch

Métricas no Service Quotas

É possível visualizar e gerenciar as cotas de grupos de usuários e de bancos de identidades do Amazon Cognito em um local central com o Service Quotas. É possível usar o console do Service Quotas para visualizar detalhes sobre uma cota específica, monitorar a utilização da cota e solicitar um aumento da cota. Para alguns tipos de cota, você pode criar um CloudWatch alarme para rastrear a utilização da cota. Para saber mais sobre quais métricas do Amazon Cognito você pode monitorar, consulte Rastrear o uso da cota.

Para visualizar a utilização de Service quotas de grupos de usuários e bancos de identidades do Amazon Cognito, conclua as etapas a seguir.

1. Abra o console do Service Quotas.
2. No painel de navegação, escolha Serviços da AWS .
3. Na lista de Serviços da AWS , pesquise e escolha Grupos de usuários do Amazon Cognito ou Identidades federadas do Amazon Cognito. A página de cota de serviço é exibida.
4. Selecione uma cota que ofereça suporte ao CloudWatch monitoramento. Por exemplo, escolha `Rate of UserAuthentication requests` em grupos de usuários do Amazon Cognito.
5. Role para baixo até Monitoring (Monitoramento). Essa seção aparece somente para cotas que oferecem suporte ao CloudWatch monitoramento.
6. Em Monitoring (Monitoramento), você pode visualizar a utilização atual da cota de serviço no gráfico.
7. Em Monitoring (Monitoramento), selecione uma hora, três horas, doze horas, um dia, três dias ou uma semana.
8. Selecione qualquer área dentro do gráfico para exibir a porcentagem de utilização da cota de serviço. A partir daqui, você pode adicionar o gráfico ao seu painel ou usar o menu de ação para selecionar Exibir em métricas, que o levará às métricas relacionadas no CloudWatch console.

Login no Amazon Cognito AWS CloudTrail

O Amazon Cognito é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço no Amazon Cognito. CloudTrail captura um subconjunto de chamadas de API para o Amazon Cognito como eventos, incluindo chamadas do console do Amazon Cognito e de chamadas de código para as operações da API do Amazon Cognito. Se você criar uma trilha, poderá optar por entregar CloudTrail eventos em um bucket do Amazon S3, incluindo eventos para o Amazon Cognito. Se você não configurar uma trilha, ainda poderá ver os eventos mais recentes no CloudTrail console no Histórico de eventos. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita ao Amazon Cognito, o endereço IP a partir do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais CloudTrail, inclusive como configurá-lo e ativá-lo, consulte o Guia AWS CloudTrail do usuário.

Você também pode criar CloudWatch alarmes da Amazon para CloudTrail eventos específicos. Por exemplo, você pode configurar o CloudWatch para acionar um alarme se uma configuração de grupo de identidades for alterada. Para obter mais informações, consulte Criação de CloudWatch alarmes para CloudTrail eventos: exemplos.

Tópicos

- Informações que o Amazon Cognito envia para CloudTrail
- Análise de CloudTrail eventos do Amazon Cognito com o Amazon Logs Insights CloudWatch
- Exemplo de eventos do Amazon Cognito

Informações que o Amazon Cognito envia para CloudTrail

CloudTrail é ativado quando você cria seu Conta da AWS. Quando uma atividade de evento suportada ocorre no Amazon Cognito, essa atividade é registrada em um CloudTrail evento junto com outros eventos de AWS serviço no histórico de eventos. Você pode visualizar, pesquisar e baixar eventos recentes em sua AWS conta. Para obter mais informações, consulte Visualização de eventos com histórico de CloudTrail eventos.

Para um registro contínuo de eventos em sua AWS conta, incluindo eventos para o Amazon Cognito, crie uma trilha. Uma CloudTrail trilha entrega arquivos de log para um bucket do Amazon S3. Por padrão, quando uma trilha é criada no console, a mesma é aplicada a todas as regiões da . A trilha registra eventos de todas as regiões na AWS partição e entrega os arquivos de log ao bucket do

Amazon S3 que você especificar. Além disso, você pode configurar outros AWS serviços para analisar e agir com base nos dados de eventos coletados nos CloudTrail registros. Para obter mais informações, consulte:

- Visão geral da criação de uma trilha
- CloudTrail serviços e integrações suportados
- Configurando notificações do Amazon SNS para CloudTrail
- Recebendo arquivos de CloudTrail log de várias regiões e Recebendo arquivos de CloudTrail log de várias contas

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário-raiz ou usuário do IAM.
- Se a solicitação foi feita com credenciais de segurança temporárias de um perfil ou de um usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte Elemento userIdentity do CloudTrail .

Dados confidenciais em AWS CloudTrail

Como grupos de usuários e grupos de identidades processam dados do usuário, o Amazon Cognito obscurece alguns campos privados em seus CloudTrail eventos com o valor. `HIDDEN_DUE_TO_SECURITY_REASONS` Para ver exemplos de campos que o Amazon Cognito não preenche para eventos, consulte Exemplo de eventos do Amazon Cognito. O Amazon Cognito oculta apenas alguns campos que normalmente contêm informações do usuário, como senhas e tokens. O Amazon Cognito não realiza nenhuma detecção ou mascaramento automático de informações de identificação pessoal que você preenche em campos não privados em suas solicitações de API.

Eventos de grupos de usuários

O Amazon Cognito suporta o registro em log de todas as ações listadas na página de ações do grupo de usuários como eventos em arquivos de CloudTrail log. O Amazon Cognito registra eventos do grupo de usuários CloudTrail como eventos de gerenciamento.

O `eventType` campo em uma CloudTrail entrada de grupos de usuários do Amazon Cognito informa se seu aplicativo fez a solicitação para a API de grupos de usuários do Amazon Cognito ou para um

endpoint que fornece recursos para OpenID Connect, SAML 2.0 ou páginas de login gerenciadas. As solicitações de API têm um `eventType` de `AwsApiCall` e as solicitações de endpoint têm um `eventType` de `AwsServiceEvent`.

O Amazon Cognito registra as seguintes solicitações em seus serviços de login gerenciados como eventos em. CloudTrail

Hosted UI (classic) events

Eventos de UI hospedados (clássicos) em CloudTrail

Operation	Description
`Login_GET` , `CognitoAuthentication`	Um usuário visualiza ou envia credenciais para o Endpoint de login.
`OAuth2_Authorize_GET` , `Beta_Authorize_GET`	Um usuário visualiza o Autorizar endpoint.
`OAuth2Response_GET` , `OAuth2Response_POST`	Um usuário envia um token de IdP ao endpoint `/oauth2/idpresponse` .
`SAML2Response_POST` , `Beta_SAML2Response_POST`	Um usuário envia uma afirmação SAML do IdP ao endpoint `/saml2/idpresponse` .
`Login_OIDC_SAML_POST`	Um usuário insere um nome de usuário no Endpoint de login e ele corresponde a um identificador IdP.
`Token_POST` , `Beta_Token_POST`	Um usuário envia um código de autorização ao Endpoint de token.
`Signup_GET` , `Signup_POST`	Um usuário envia informações de login ao endpoint `/signup`.
`Confirm_GET` , `Confirm_POST`	Um usuário envia um código de confirmação na interface do usuário hospedada.

Operation	Description
`ResendCode_POST`	Um usuário envia uma solicitação de reenvio de código de confirmação na interface do usuário hospedada.
`ForgotPassword_GET` , `ForgotPassword_POST`	Um usuário envia uma solicitação de redefinição de senha ao endpoint `/forgotPassword` .
`ConfirmForgotPassword_GET` , `ConfirmForgotPassword_POST`	Um usuário envia um código ao endpoint `/confirmForgotPassword` que confirma a solicitação de `ForgotPassword` .
`ResetPassword_GET` , `ResetPassword_POST`	Um usuário envia uma nova senha na interface do usuário hospedada.
`Mfa_GET`, `Mfa_POST`	Um usuário envia um código de autenticação multifator (MFA) na interface do usuário hospedada.
`MfaOption_GET` , `MfaOption_POST`	Um usuário escolhe seu método preferido de MFA na interface do usuário hospedada.
`MfaRegister_GET` , `MfaRegister_POST`	Um usuário envia um código de autenticação multifator (MFA) na interface do usuário hospedada ao registrar a MFA.
`Logout`	Um usuário faz logout no endpoint `/logout`.
`SAML2Logout_POST`	Um usuário faz logout no endpoint `/saml2/logout` .
`Error_GET`	Um usuário visualiza uma página de erro na interface do usuário hospedada.
`UserInfo_GET` , `UserInfo_POST`	Um usuário ou IdP troca informações com o endpoint userinfo.

Operation	Description
`Confirm_With_Link_GET`	Um usuário envia uma confirmação baseada em um link que o Amazon Cognito enviou em uma mensagem de e-mail.
`Event_Feedback_GET`	Um usuário envia feedback para o Amazon Cognito sobre um evento de proteção contra ameaças.

Managed login events

Eventos de login gerenciados em CloudTrail

Operation	Description
`login_POST`	Um usuário envia credenciais para o Endpoint de login.
`login_continue_POST`	Um usuário que já fez login uma vez opta por fazer login novamente.
`forgotPassword_POST`	Um usuário redefine sua senha.
`selectChallenge_POST`	Um usuário responde a um desafio de autenticação após enviar seu nome de usuário ou credenciais.
`confirmUser_GET`	Um usuário abre o link em uma mensagem de e-mail de confirmação ou verificação.
`mfa_back_POST`	Um usuário clica no botão Voltar após uma solicitação de MFA.
`mfa_options_POST`	Um usuário seleciona uma opção de MFA.
`mfa_phone_register_POST`	Um usuário envia um número de telefone para registrar como fator de MFA. Essa operação faz com que o Amazon Cognito

Operation	Description
	envie um código de MFA para seu número de telefone.
`mfa_phone_verify_POST`	Um usuário envia um código de MFA enviado para seu número de telefone.
`mfa_phone_resendCode_POST`	Um usuário envia uma solicitação de reenvio de código de MFA para seu número de telefone.
`mfa_totp_POST`	Um usuário envia um código MFA de TOTP.
`signup_POST`	Um usuário envia informações para sua página de login gerenciado `/signup`.
`signup_confirm_POST`	Um usuário envia um código de confirmação enviado por e-mail ou SMS.
`verifyCode_POST`	Um usuário envia uma senha de uso único (OTP) para autenticação sem senha.
`passkeys_add_POST`	Um usuário envia uma solicitação de registro de nova credencial de chave de acesso.
`passkeys_add_GET`	Um usuário navega até a página onde pode registrar uma chave de acesso.
`login_passkey_POST`	Um usuário faz login com uma chave de acesso.

Note

O Amazon Cognito registra`UserSub`, mas não `UserName` em CloudTrail registros, solicitações específicas de um usuário. Você pode encontrar um usuário para um determinado `UserSub` chamando a API `ListUsers` e usando um filtro para sub.

Eventos de bancos de identidades

Eventos de dados

O Amazon Cognito registra os seguintes eventos de identidade do Amazon Cognito como eventos CloudTrail de dados. Eventos de dados são operações de API de plano de dados de alto volume que CloudTrail não são registradas por padrão. Há cobranças adicionais para eventos de dados.

- GetCredentialsForIdentity
- GetId
- GetOpenIdToken
- GetOpenIdTokenForDeveloperIdentity
- UnlinkIdentity

Para gerar CloudTrail registros para essas operações de API, você deve ativar eventos de dados em sua trilha e escolher seletores de eventos para os grupos de identidade do Cognito. Para obter mais informações, consulte Registro eventos de dados em logs para trilhas no Guia do usuário do AWS CloudTrail .

Você também pode adicionar seletores de eventos de grupos de identidades à sua trilha com o comando da CLI a seguir.

```
aws cloudtrail put-event-selectors --trail-name <trail name> --advanced-event-selectors
 \
"{\
   \"Name\": \"Cognito Selector\",\
   \"FieldSelectors\": [\
      {\
         \"Field\": \"eventCategory\",\
         \"Equals\": [\
            \"Data\"\
         ]\
      },\
      {\
         \"Field\": \"resources.type\",\
         \"Equals\": [\
            \"AWS::Cognito::IdentityPool\"\
         ]\
      }\
   ]\
```

```
}"
```

Eventos de gerenciamento

O Amazon Cognito registra o restante das operações de API dos grupos de identidade do Amazon Cognito como eventos de gerenciamento. CloudTrail operações de API de eventos de gerenciamento de registros por padrão.

Para obter uma lista das operações de API dos grupos de identidades do Amazon Cognito nas quais o Amazon Cognito faz login CloudTrail, consulte a Referência da API dos grupos de identidades do Amazon Cognito.

Amazon Cognito Sync

O Amazon Cognito registra todas as operações da API do Amazon Cognito Sync como eventos de gerenciamento. Para obter uma lista das operações da API Amazon Cognito Sync nas quais o Amazon Cognito faz login, consulte a Referência CloudTrail da API Amazon Cognito Sync.

Análise de CloudTrail eventos do Amazon Cognito com o Amazon Logs Insights CloudWatch

Você pode pesquisar e analisar seus CloudTrail eventos do Amazon Cognito com o Amazon CloudWatch Logs Insights. Quando você configura sua trilha para enviar eventos para o CloudWatch Logs, CloudTrail envia somente os eventos que correspondem às suas configurações de trilha.

Para consultar ou pesquisar seus CloudTrail eventos do Amazon Cognito, no CloudTrail console, certifique-se de selecionar a opção Gerenciamento de eventos nas configurações da trilha para poder monitorar as operações de gerenciamento realizadas em seus AWS recursos. Você pode selecionar a opção Eventos do Insights nas configurações de trilha quando quiser identificar erros, atividades ou comportamento incomuns do usuário em sua conta.

Exemplos de consultas do Amazon Cognito

Você pode usar as seguintes consultas no CloudWatch console da Amazon.

Consultas gerais

Encontre os 25 eventos de log adicionados mais recentemente.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
| filter eventSource = "cognito-idp.amazonaws.com"
```

Obtenha uma lista dos 25 eventos de log adicionados mais recentemente que incluem exceções.

```
fields @timestamp, @message | sort @timestamp desc | limit 25
| filter eventSource = "cognito-idp.amazonaws.com" and @message like /Exception/
```

Consultas de exceção e erro

Encontre os 25 eventos de log adicionados mais recentemente com código de erro `NotAuthorizedException` junto com o grupo de usuários do Amazon Cognito `sub`.

```
fields @timestamp, additionalEventData.sub as user | sort @timestamp desc | limit 25
| filter eventSource = "cognito-idp.amazonaws.com" and errorCode=
 "NotAuthorizedException"
```

Encontre o número de registros com `sourceIPAddress` e o correspondente `eventName`.

```
filter eventSource = "cognito-idp.amazonaws.com"
| stats count(*) by sourceIPAddress, eventName
```

Encontre os 25 principais endereços IP que acionaram um erro de `NotAuthorizedException`.

```
filter eventSource = "cognito-idp.amazonaws.com" and errorCode=
 "NotAuthorizedException"
| stats count(*) as count by sourceIPAddress, eventName
| sort count desc | limit 25
```

Encontre os 25 principais endereços IP que chamaram a API `ForgotPassword`.

```
filter eventSource = "cognito-idp.amazonaws.com" and eventName = 'ForgotPassword'
| stats count(*) as count by sourceIPAddress
| sort count desc | limit 25
```

Exemplo de eventos do Amazon Cognito

O Amazon Cognito registra informações AWS CloudTrail sobre a atividade de autenticação do usuário e a atividade de gerenciamento administrativo. Isso se aplica tanto aos grupos de usuários quanto aos bancos de identidades. Por exemplo, você pode ver eventos `GetId` e `UpdateIdentityPool` na mesma trilha ou eventos `UpdateAuthEventFeedback` e `SetRiskConfiguration`. Você também verá logs de grupos de usuários para atividades de interface de usuário hospedadas que não correspondem às operações na API de grupos de

usuários. Esta seção mostra alguns exemplos de logs que você pode encontrar. Para entender o esquema de CloudTrail eventos de qualquer operação, gere uma solicitação para essa operação e analise os eventos que ela cria em sua trilha.

Uma trilha pode entregar eventos como arquivos de log para um bucket do Amazon S3 que você especificar. CloudTrail os arquivos de log 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 da solicitação e assim por diante. CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas de API, portanto, eles não aparecem em nenhuma ordem específica.

Tópicos

Exemplos de CloudTrail eventos para uma inscrição de interface de usuário hospedada

Os CloudTrail eventos de exemplo a seguir demonstram as informações que o Amazon Cognito registra quando um usuário se cadastra por meio da interface de usuário hospedada.

O Amazon Cognito registra o seguinte evento quando um novo usuário navega até a página de login da aplicação.


```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-04-06T05:38:12Z",
```

```
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Login_GET",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "errorCode": "",
    "errorMessage": "",
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200.0
        },
        "requestParameters":
        {
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "response_type":
            [
                "token"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        }
    },
    "eventID": "382ae09a-151d-4116-8f2b-6ac0a804a38c",
    "readOnly": true,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```


O Amazon Cognito registra o evento a seguir quando um novo usuário escolhe Sign up (Cadastrar-se) na página de login da aplicação.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-05T23:21:43Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Signup_GET",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
        },
        "requestParameters":
        {
            "response_type":
            [
                "code"
            ],
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        },
        "userPoolDomain": "mydomain.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "requestID": "7a63e7c2-b057-4f3d-a171-9d9113264fff",
    "eventID": "5e7b27a0-6870-4226-adb4-f86cd51ac5d8",
    "readOnly": true,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
```

```
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

O Amazon Cognito registra o evento a seguir quando um novo usuário seleciona um nome de usuário, insere um endereço de e-mail e escolhe uma senha na página de login da aplicação. O Amazon Cognito não registra informações de identificação sobre a identidade do usuário no. CloudTrail

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-05T23:22:05Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Signup_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 302
        },
        "requestParameters":
        {
            "password":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "requiredAttributes[email]":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
```

```
            "response_type":
            [
                "code"
            ],
            "_csrf":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "client_id":
            [
                "1example23456789"
            ],
            "username":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ]
        },
        "userPoolDomain": "mydomain.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "requestID": "9ad58dd8-3517-4aa8-96a5-d17a01df9eb4",
    "eventID": "c75eb7a5-eb8c-43d1-8331-f4412e756e69",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

O Amazon Cognito registra o evento a seguir quando um novo usuário acessa a página de confirmação do usuário na interface do usuário hospedada após se cadastrar.

```
{
    "eventVersion": "1.08",
    "userIdentity":
```

```
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-05T23:22:06Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Confirm_GET",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
        },
        "requestParameters":
        {
            "response_type":
            [
                "code"
            ],
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        },
        "userPoolDomain": "mydomain.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "requestID": "58a5b170-3127-45bb-88cc-3e652d779e0b",
    "eventID": "7f87291a-6d50-409a-822f-e3a5ec7e60da",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
```

```
    },
    "eventCategory": "Management"
}
```

O Amazon Cognito registra o evento a seguir quando, na página de confirmação do usuário na interface do usuário hospedada, um usuário insere um código enviado pelo Amazon Cognito em uma mensagem de e-mail.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-05T23:23:32Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Confirm_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 302
        },
        "requestParameters":
        {
            "confirm":
            [
                ""
            ],
            "deliveryMedium":
            [
                "EMAIL"
            ],
            "sub":
            [
                "704b1e47-34fe-40e9-8c41-504997494531"
            ],
            "code":
```

```
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "destination":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "response_type":
            [
                "code"
            ],
            "_csrf":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "cognitoAsfData":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "client_id":
            [
                "1example23456789"
            ],
            "username":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ]
        },
        "userPoolDomain": "mydomain.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "requestID": "9764300a-ed35-4f87-8a0f-b18b3fe2b11e",
    "eventID": "e24ac6e5-2f70-4c6e-ad4e-2f08a547bb36",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
```

```
    },
    "eventCategory": "Management"
}
```

Exemplo de CloudTrail evento para uma solicitação SAML

O Amazon Cognito registra o seguinte evento quando um usuário que foi autenticado com seu IdP SAML envia a afirmação SAML ao endpoint `/saml2/idpresponse`.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-06T00:50:57Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "SAML2Response_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 302
        },
        "requestParameters":
        {
            "RelayState":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "SAMLResponse":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ]
        },
        "userPoolDomain": "mydomain.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
```

```
    "requestID": "4f6f15d1-c370-4a57-87f0-aac4817803f7",
    "eventID": "9824b50f-d9d1-4fb8-a2c1-6aa78ca5902a",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "625647942648",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

Exemplos de CloudTrail eventos para solicitações ao endpoint do token

A seguir, exemplos de eventos de solicitações ao Endpoint de token.

O Amazon Cognito registra o evento a seguir quando um usuário que foi autenticado e recebeu um código de autorização envia o código ao endpoint `/oauth2/token`.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-12T22:12:30Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Token_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
        },
        "requestParameters":
        {
            "code":
```

```
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "grant_type":
            [
                "authorization_code"
            ],
            "redirect_uri":
            [
                "https://www.amazon.com"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        },
        "userPoolDomain": "mydomain.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "requestID": "f257f752-cc14-4c52-ad5b-152a46915238",
    "eventID": "0bd1586d-cd3e-4d7a-abaf-fd8bfc3912fd",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

O Amazon Cognito registra o evento a seguir quando o sistema de back-end envia uma solicitação de `client_credentials` para um token de acesso ao endpoint `/oauth2/token`.

```
{
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-12T21:07:05Z",
    "eventSource": "cognito-idp.amazonaws.com",
```

```
    "eventName": "Token_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
        },
        "requestParameters":
        {
            "grant_type":
            [
                "client_credentials"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        },
        "userPoolDomain": "mydomain.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "requestID": "4f871256-6825-488a-871b-c2d9f55caff2",
    "eventID": "473e5cbc-a5b3-4578-9ad6-3dfdcb8a6d34",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```


O Amazon Cognito registra o evento a seguir quando a aplicação troca um token de atualização por um novo ID e token de acesso com o endpoint `/oauth2/token`.


```
{
```

```
    "eventVersion": "1.08",
    "userIdentity":
    {
        "accountId": "123456789012"
    },
    "eventTime": "2022-05-12T22:16:40Z",
    "eventSource": "cognito-idp.amazonaws.com",
    "eventName": "Token_POST",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)...",
    "requestParameters": null,
    "responseElements": null,
    "additionalEventData":
    {
        "responseParameters":
        {
            "status": 200
        },
        "requestParameters":
        {
            "refresh_token":
            [
                "HIDDEN_DUE_TO_SECURITY_REASONS"
            ],
            "grant_type":
            [
                "refresh_token"
            ],
            "client_id":
            [
                "1example23456789"
            ]
        },
        "userPoolDomain": "mydomain.auth.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_EXAMPLE"
    },
    "requestID": "2829f0c6-a3a9-4584-b046-11756dfe8a81",
    "eventID": "12bd3464-59c7-44fa-b8ff-67e1cf092018",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "serviceEventDetails":
```

```
    {
        "serviceAccountId": "111122223333"
    },
    "eventCategory": "Management"
}
```

Exemplo de CloudTrail evento para CreateIdentityPool

O exemplo a seguir é uma entrada de log de uma solicitação da ação `CreateIdentityPool`. A solicitação foi feita por uma usuária do IAM chamada Alice.

```
{
    "eventVersion": "1.03",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/Alice",
        "accountId": "123456789012",
        "accessKeyId": "['EXAMPLE_KEY_ID']",
        "userName": "Alice"
    },
    "eventTime": "2016-01-07T02:04:30Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "CreateIdentityPool",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "USER_AGENT",
    "requestParameters": {
        "identityPoolName": "TestPool",
        "allowUnauthenticatedIdentities": true,
        "supportedLoginProviders": {
            "graph.facebook.com": "000000000000000"
        }
    },
    "responseElements": {
        "identityPoolName": "TestPool",
        "identityPoolId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE",
        "allowUnauthenticatedIdentities": true,
        "supportedLoginProviders": {
            "graph.facebook.com": "000000000000000"
        }
    },
    "requestID": "15cc73a1-0780-460c-91e8-e12ef034e116",
```

```
    "eventID": "f1d47f93-c708-495b-bff1-cb935a6064b2",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

Exemplo de CloudTrail evento para GetCredentialsForIdentity

O exemplo a seguir é uma entrada de log de uma solicitação da ação `GetCredentialsForIdentity`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Unknown"
    },
    "eventTime": "2023-01-19T16:55:08Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "GetCredentialsForIdentity",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.4",
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.get-credentials-for-identity",
    "requestParameters": {
        "logins": {
            "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
 "HIDDEN_DUE_TO_SECURITY_REASONS"
        },
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "aAaAaAaAaAaAab1111111111EXAMPLE",
            "expiration": "Jan 19, 2023 5:55:08 PM"
        },
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "requestID": "659dfc23-7c4e-4e7c-858a-1abce884d645",
    "eventID": "6ad1c766-5a41-4b28-b5ca-e223ccb00f0d",
    "readOnly": false,
    "resources": [{
        "accountId": "111122223333",
        "type": "AWS::Cognito::IdentityPool",
```

```
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
}
```

Exemplo de CloudTrail evento para GetId

O exemplo a seguir é uma entrada de log de uma solicitação da ação `GetId`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Unknown"
    },
    "eventTime": "2023-01-19T16:55:05Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "GetId",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.4",
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.get-id",
    "requestParameters": {
        "identityPoolId": "us-east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE",
        "logins": {
            "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
 "HIDDEN_DUE_TO_SECURITY_REASONS"
        }
    },
    "responseElements": {
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "requestID": "dc28def9-07c8-460a-a8f3-3816229e6664",
    "eventID": "c5c459d9-40ec-41fd-8f6b-57865d5a9975",
    "readOnly": false,
    "resources": [{
        "accountId": "111122223333",
        "type": "AWS::Cognito::IdentityPool",
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
```

```
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
}
```

Exemplo de CloudTrail evento para GetOpenIdToken

O exemplo a seguir é uma entrada de log de uma solicitação da ação `GetOpenIdToken`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Unknown"
    },
    "eventTime": "2023-01-19T16:55:08Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "GetOpenIdToken",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.4",
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.get-open-id-token",
    "requestParameters": {
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE",
        "logins": {
            "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
 "HIDDEN_DUE_TO_SECURITY_REASONS"
        }
    },
    "responseElements": {
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "requestID": "a506ba18-10d7-4fdb-9548-a8187b2e38bb",
    "eventID": "19ffc1a6-6ed8-4580-a4e1-3062c5ce6457",
    "readOnly": false,
    "resources": [{
        "accountId": "111122223333",
        "type": "AWS::Cognito::IdentityPool",
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
```

```
    "eventCategory": "Data"
}
```

Exemplo de CloudTrail evento para GetOpenIdTokenForDeveloperIdentity

O exemplo a seguir é uma entrada de log de uma solicitação da ação `GetOpenIdTokenForDeveloperIdentity`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA1EXAMPLE:johns-AssumedRoleSession",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/johns-AssumedRoleSession",
        "accountId": "111122223333",
        "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA1EXAMPLE",
                "arn": "arn:aws:iam::111122223333:role/Admin",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2023-01-19T16:53:14Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-01-19T16:55:08Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "GetOpenIdTokenForDeveloperIdentity",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "27.0.3.154",
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.get-open-id-token-for-developer-identity",
    "requestParameters": {
        "tokenDuration": 900,
        "identityPoolId": "us-east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE",
        "logins": {
            "JohnsDeveloperProvider": "HIDDEN_DUE_TO_SECURITY_REASONS"
        }
```

```
    },
    "responseElements": {
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "requestID": "b807df87-57e7-4dd6-b90c-b06f46a61c21",
    "eventID": "f26fed91-3340-4d70-91ae-cdf555547b76",
    "readOnly": false,
    "resources": [{
        "accountId": "111122223333",
        "type": "AWS::Cognito::IdentityPool",
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
}
```

Exemplo de CloudTrail evento para UnlinkIdentity

O exemplo a seguir é uma entrada de log de uma solicitação da ação `UnlinkIdentity`.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Unknown"
    },
    "eventTime": "2023-01-19T16:55:08Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "UnlinkIdentity",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.4",
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.unlink-identity",
    "requestParameters": {
        "logins": {
            "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
 "HIDDEN_DUE_TO_SECURITY_REASONS"
        },
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE",
        "loginsToRemove": ["cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa"]
    },
    "responseElements": null,
```

```
    "requestID": "99c2c8e2-9c29-416f-bb17-b650a5cbada9",
    "eventID": "d8e26126-202a-43c2-b458-3f225efaedc7",
    "readOnly": false,
    "resources": [{
        "accountId": "111122223333",
        "type": "AWS::Cognito::IdentityPool",
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
}
```

Acesse o Amazon Cognito usando um endpoint de interface ()AWS PrivateLink

Você pode usar AWS PrivateLink para criar uma conexão privada entre sua VPC e o Amazon Cognito. Você pode acessar o Amazon Cognito como se estivesse em sua VPC, sem o uso de um gateway de internet, dispositivo NAT, conexão VPN ou conexão. Direct Connect As instâncias na sua VPC não precisam de endereços IP públicos para acessar o Amazon Cognito.

Estabeleça essa conectividade privada criando um endpoint de interface, habilitado pelo AWS PrivateLink. Criaremos um endpoint de interface de rede em cada sub-rede que você habilitar para o endpoint de interface. Essas são interfaces de rede gerenciadas pelo solicitante que servem como ponto de entrada para o tráfego destinado ao Amazon Cognito.

Para saber mais, consulte Acessar os Serviços da AWS pelo AWS PrivateLink no Guia do AWS PrivateLink .

> **Important**
>
> Atualmente, os seguintes tipos de autenticação não são compatíveis com AWS PrivateLink:
>
> 1. Autorização máquina a máquina (M2M) com o fluxo de credenciais do cliente OAuth 2.0
> 2. Faça login com login gerenciado e a interface de usuário hospedada clássica.

Tópicos

- Fluxos de autenticação para AWS PrivateLink integração
- Modos operacionais para AWS PrivateLink
- Considerações sobre o Amazon Cognito
- Controle do acesso com políticas de controle de recursos
- Crie um endpoint de interface para o Amazon Cognito
- Criar uma política de endpoint para o endpoint de interface
- Crie uma política baseada em identidade para operações AWS PrivateLink

Fluxos de autenticação para AWS PrivateLink integração

A tabela a seguir descreve os fluxos de autenticação disponíveis para os clientes e as políticas do IAM que você pode aplicar para controlá-los. VPCs As políticas que você pode avaliar nas solicitações para grupos de usuários são políticas de controle de recursos (RCPs), políticas de VPC endpoint e políticas baseadas em identidade.

Recurso	Fluxo de autenticação	Políticas avaliadas quando o cliente transita por um VPC endpoint	Políticas avaliadas quando a origem do cliente é pública
Grupo de usuários	Login gerenciado e login clássico com interface de usuário hospedada	Nenhum (sem acesso) [1]	Nenhum [2]
Grupo de usuários	Machine-to-machine autorização	Nenhum (sem acesso) [1]	Nenhum [2]
Grupo de usuários	Solicitações não autenticadas do SDK e da API REST	RCPs, políticas de VPC endpoint [3]	RCPs
Grupo de usuários	Solicitações autenticadas SigV4 da API REST e do SDK	RCPs, políticas de endpoint de VPC, políticas baseadas em identidade [3]	RCPs, políticas baseadas em identidade

Recurso	Fluxo de autenticação	Políticas avaliadas quando o cliente transita por um VPC endpoint	Políticas avaliadas quando a origem do cliente é pública
Grupo de identidades	Solicitações não autenticadas do SDK e da API REST (fluxos básicos e aprimorados)	RCPs, políticas de VPC endpoint	RCPs
Grupo de identidades	Solicitações autenticadas SigV4 do SDK e da API REST (fluxo autenticado pelo desenvolvedor)	RCPs, políticas baseadas em identidade	RCPs, políticas baseadas em identidade

[1] Os VPC endpoints não aceitam solicitações de domínios de grupos de usuários. Se o cliente tiver uma rota para a Internet, o NAT será aplicado, tornando pública a origem.

[2] A existência de um domínio de grupo de usuários impede a conclusão de qualquer solicitação de grupo de usuários que transite por um VPC endpoint. Qualquer cliente pode usar caminhos de transporte público somente para o domínio do grupo de usuários e os endpoints do serviço de API, tornando o VPC endpoint inutilizável para o grupo de usuários. Grupos de usuários com domínios atribuídos são incompatíveis com o. AWS PrivateLink

[3] O grupo de usuários não deve ter um domínio atribuído.

Modos operacionais para AWS PrivateLink

Os seguintes exemplos de modelos de implementação são compatíveis com o AWS PrivateLink Amazon Cognito.

Recurso	Implementação	Ações
Grupo de usuários	Aplicativo SDK ou REST API totalmente privado	1. Excluir domínio 2. Criar endpoint da VPC 3. Configure o RCP para `Deny` todas as ações do cognito-idp, exceto do VPC

Recurso	Implementação	Ações
Grupo de usuários	Públicos e privados	1. Excluir domínio 2. Criar endpoint da VPC
Grupo de usuários	Servidor de autorização OAuth 2.0 privado ou público	1. Não disponível para VPC
Grupo de identidades	Totalmente privado	1. Criar endpoint da VPC 2. Configure o RCP para `Deny` todas as ações de identidade cognitiva, exceto da VPC
Grupo de identidades	Públicos e privados	1. Criar endpoint da VPC

Considerações sobre o Amazon Cognito

Antes de configurar um endpoint de interface para o Amazon Cognito, leia as considerações no Guia.AWS PrivateLink O Amazon Cognito oferece suporte para fazer chamadas para todas as ações da API do Amazon Cognito por meio do endpoint da interface. Para obter mais informações sobre essas operações, consulte a Referência da API de grupos de usuários do Amazon Cognito e a Referência da API de Identidades Federadas do Amazon Cognito.

AWS PrivateLink para o Amazon Cognito está disponível somente em regiões comerciais AWS .

Tópicos

- Grupos de usuários e AWS PrivateLink
- Grupos de identidades e AWS PrivateLink

Grupos de usuários e AWS PrivateLink

Você pode fazer solicitações para todas as operações de API de grupos de usuários por meio do endpoint da interface, mas não para operações que seu aplicativo solicita do servidor de autorização do grupo de usuários OAuth 2.0, por exemplo, concessões de credenciais de cliente e login gerenciado.

A API de grupos de `cognito-idp` usuários tem operações de API não autenticadas, autenticadas e autorizadas por token. Você pode conceder permissões para operações autenticadas nas políticas de controle de recursos e endpoints da VPC. Você também pode conceder permissões para operações não autenticadas e autorizadas por token, ao contrário das políticas baseadas em identidade. Os tipos de políticas de controle de recursos e endpoints de VPC são capazes de avaliar, negar ou permitir solicitações de operações que, de outra forma, seriam públicas.

As solicitações para endpoints de domínio também são públicas, mas você não pode avaliá-las nas políticas. O DNS privado da VPC não encaminha solicitações de domínios de grupos de usuários para o seu VPC endpoint. Você só pode fazer solicitações de serviços de domínio por meio de caminhos públicos da Internet. Para obter mais informações, consulte Efeitos das políticas nas operações do grupo de usuários.

Tópicos

Operações compatíveis

Os sistemas em uma VPC podem enviar solicitações para as ações da API do grupo de usuários, mas não para os endpoints do domínio do grupo de usuários. Os fluxos de trabalho do OpenID Connect (OIDC) e OAuth 2.0 que usam endpoints de domínio, por exemplo machine-to-machine(M2M), login federado e concessões de código de autorização, não podem ser acessados por meio de VPC endpoints. As políticas de VPC endpoint não afetam esses fluxos de trabalho HTTP e não podem processá-los. As solicitações para endpoints de domínio de dentro de uma VPC sempre falham no endpoint da interface, mas continuam disponíveis por meio de DNS público e roteamento quando você configura endpoints de VPC para seus grupos de usuários.

Para evitar a atribuição de domínios de sistemas em uma VPC, o Amazon Cognito bloqueia `CreateUserPoolDomain` solicitações no endpoint da interface. Isso evita a adição de domínios aos seus grupos de usuários de sistemas que estão em uma VPC. Para evitar a adição de um domínio de todos os sistemas, aplique uma política de controle de recursos (RCP) como o exemplo a seguir ao seu Conta da AWS. Essa política bloqueia a `CreateUserPoolDomain` ação contra o grupo de usuários especificado.

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

```
        {
            "Principal": "*",
            "Effect": "Deny",
            "Action": [
                "cognito-idp:CreateUserPoolDomain"
            ],
            "Resource": "arn:aws:cognito-idp:us-east-1:123456789012:userpool/us-
east-1_EXAMPLE"
        }
    ]
}
```

Seu grupo de usuários pode ter um domínio e, em todos os casos, esse domínio não está disponível por meio do AWS PrivateLink. Todas as solicitações de API de grupos de usuários baseadas em SDK para terminais `cognito-idp` de serviço aceitam solicitações por meio de AWS PrivateLink, com exceção de. `CreateUserPoolDomain` Os endpoints do serviço de API do grupo de usuários e os endpoints do domínio permanecem sempre acessíveis por meio de caminhos públicos da Internet. Para abordar o acesso de fontes públicas, implemente a AWS WAF web ACLs.

Efeitos das políticas nas operações do grupo de usuários

Todas as operações de API do grupo de usuários, mesmo aquelas que normalmente são públicas e não autenticadas, podem ser controladas nas políticas de endpoint da VPC e nas políticas de controle de recursos (). RCPs Você também pode aplicar restrições ao acesso ao grupo de usuários em políticas baseadas em identidade com chaves de condição da VPC. Somente solicitações que incluam informações de autenticação no formato SigV4 podem ser controladas em políticas baseadas em identidade. O login gerenciado e as operações clássicas de interface de usuário hospedada são uma categoria separada e não estão qualificadas para o trânsito de VPC ou para a aplicação de qualquer tipo de política em suas ações.

Operações não autenticadas

As operações do Amazon Cognito para aplicativos do lado do cliente não são autenticadas com o SigV4. As operações de exemplo estão na política de exemplo emCriar uma política de endpoint para o endpoint de interface. Exemplos adicionais de operações não autenticadas são e. `GetUser` `AssociateSoftwareToken` Quando você adiciona essas operações às políticas baseadas em identidade, elas não têm efeito. No entanto, você pode permitir ou restringir o acesso a essas ações nas políticas de endpoint da VPC e. RCPs

As operações não autenticadas não estão associadas a um diretor do IAM. Sua política de VPC endpoint ou RCP deve permitir que todos os diretores realizem essas ações.

Operações autenticadas

As operações de API para administração de grupos de usuários e autenticação do lado do servidor são autenticadas com SigV4. Para operações autenticadas, você pode restringir os principais com políticas de endpoint que você aplica ao endpoint da VPC, políticas de controle de recursos em sua organização e em políticas baseadas em identidade que você aplica aos diretores. As políticas baseadas em identidade e controle de recursos reconhecem a VPC com chaves de condição baseadas em rede, como e. `aws:SourceVpc` `aws:SourceVpce`

Para obter mais informações sobre classes administrativas, do lado do servidor e do lado do cliente, de operações de API para grupos de usuários, consulte. Modelos de autorização para autenticação de API e SDK

Grupos de identidades e AWS PrivateLink

Os grupos de identidade do Amazon Cognito oferecem suporte a todas as operações de API por meio de. AWS PrivateLink

Tópicos

- Operações compatíveis
- Limitações do contexto de rede com AWS STS integração
- Chaves de contexto específicas do serviço

Operações compatíveis

Todas as operações de API de grupos de identidades são suportadas por meio do endpoint da interface. Os grupos de identidades não têm endpoints de domínio e não estão sujeitos às mesmas limitações. No entanto, os grupos de identidades têm considerações específicas para controles de acesso baseados em rede devido à sua integração com o. AWS STS

Limitações do contexto de rede com AWS STS integração

Os grupos de identidades usam AWS STS `AssumeRoleWithWebIdentity` operações para fornecer AWS credenciais temporárias. Quando os grupos de identidades AWS STS entram AWS PrivateLink no fluxo de autenticação aprimorado, as chaves de contexto de rede`aws:SourceIp`, como`aws:SourceVpc`, e, `aws:SourceVpce` contêm valores da infraestrutura de serviços dos grupos de identidades, não do contexto de rede do seu aplicativo.

Se suas políticas de confiança ou políticas de controle de recursos (RCPs) da função do IAM usarem chaves de condição baseadas em rede para restringir o acesso, as operações de grupos de identidades poderão ser negadas inesperadamente. Para resolver essa limitação, você pode usar uma das seguintes abordagens:

Etiquetas principais para identificação de serviços

Marque as funções do IAM usadas com grupos de identidades e modifique suas políticas para permitir operações quando o diretor tiver a tag apropriada. Primeiro, adicione uma tag à sua função no grupo de identidades:

```
aws iam tag-role \
    -\-role-name MyIdentityPoolRole \
    -\-tags Key=CognitoServiceCall,Value=true
```

Em seguida, modifique suas políticas baseadas em rede para permitir diretores marcados. Por exemplo, em um RCP:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Principal": "*",
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Resource": "*",
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": ["allowed-ip-ranges"]
                },
                "StringNotEqualsIfExists": {
                    "aws:ResourceTag/CognitoServiceCall": "true"
                }
            }
        }
    ]
}
```

Chaves de contexto específicas do serviço

Os grupos de identidades fornecem chaves de contexto específicas do serviço para autorização em nível de recurso nas políticas de endpoint de VPC e. RCPs Com essas chaves de contexto,

você pode habilitar um controle de acesso refinado e distinguir entre usuários autenticados e não autenticados nas políticas.

Chaves de contexto específicas do serviço disponíveis para operações não SIGv4, como,, GetIdGetCredentialsForIdentityGetOpenIdTokenUnlinkIdentity

- `cognito-identity-unauth:IdentityPoolArn`- Filtra o acesso pelo ARN do pool de identidades para usuários não autenticados
- `cognito-identity-unauth:AccountId`- Filtra o acesso pelo Conta da AWS ID para usuários não autenticados
- `cognito-identity-auth:IdentityPoolArn`- Filtra o acesso pelo ARN do pool de identidades para usuários autenticados
- `cognito-identity-auth:AccountId`- Filtra o acesso pelo Conta da AWS ID para usuários autenticados

Chaves de contexto específicas de serviço disponíveis para operações SigV4, como e DeleteIdentitiesDescribeIdentity

- `cognito-identity:IdentityPoolArn`- Filtra o acesso pelo ARN do pool de identidades

Você pode usar essas chaves de contexto nas políticas de endpoint da VPC para restringir o acesso com base no status da autenticação, conforme demonstrado no exemplo a seguir:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "cognito-identity:GetId",
                "cognito-identity:GetCredentialsForIdentity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cognito-identity-unauth:IdentityPoolArn": "arn:aws:cognito-
identity:us-east-1:123456789012:identitypool/us-east-1:12345678-ffff-ffff-ffff-123456"
                }
```

```
            }
        }
    ]
}
```

Controle do acesso com políticas de controle de recursos

O Amazon Cognito oferece suporte ao controle do acesso aos seus recursos com políticas de controle de recursos ()RCPs. Com chaves de condição baseadas em rede, RCPs pode definir as redes e ações que são permitidas para AWS PrivateLink acessar seus grupos de usuários e grupos de identidades. As `Action` instruções contidas RCPs podem controlar o acesso às operações de API do grupo de usuários autenticadas e não autenticadas.

Por exemplo, o exemplo de política a seguir impede o acesso a todos os grupos de usuários de uma VPC específica.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyCognitoAccessOutsideVPC",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "cognito-idp:*",
      "Resource": "*",
      "Condition": {
        "StringNotEqualsIfExists": {
          "aws:SourceVpc": "vpc-02d6770f46ef1653b"
        }
      }
    }
  ]
}
```

Crie um endpoint de interface para o Amazon Cognito

Você pode criar um endpoint de interface para o Amazon Cognito usando o console da Amazon VPC ou o (). AWS Command Line Interface AWS CLI Para obter mais informações, consulte Criar um endpoint de interface no Guia do usuário do AWS PrivateLink .

Crie um endpoint de interface para grupos de usuários do Amazon Cognito usando o seguinte nome de serviço:

```
com.amazonaws.region.cognito-idp
```

Crie um endpoint de interface para grupos de identidade do Amazon Cognito usando o seguinte nome de serviço:

```
com.amazonaws.region.cognito-identity
```

Se você habilitar o DNS privado para o endpoint da interface, poderá fazer solicitações de API para o Amazon Cognito usando seu nome de DNS regional padrão. Por exemplo, `cognito-idp.us-east-1.amazonaws.com` para grupos de usuários e grupos `cognito-identity.us-east-1.amazonaws.com` de identidades.

Criar uma política de endpoint para o endpoint de interface

Uma política de endpoint é um recurso do IAM que pode ser anexado ao endpoint de interface. A política de endpoint padrão permite acesso total ao Amazon Cognito por meio do endpoint da interface. Para controlar o acesso permitido ao Amazon Cognito a partir da sua VPC, anexe uma política de endpoint personalizada ao endpoint da interface.

Uma política de endpoint especifica as seguintes informações:

- As entidades principais que podem realizar ações (Contas da AWS, usuários do IAM e perfis do IAM).
- As ações que podem ser realizadas.
- Os recursos nos quais as ações podem ser executadas.
- As condições que devem ser satisfeitas antes que a solicitação seja permitida ou negada.

Para obter mais informações, consulte Controlar o acesso aos serviços usando políticas de endpoint no Guia do AWS PrivateLink .

Exemplo: política de VPC endpoint para ações de grupos de usuários

Veja a seguir um exemplo de uma política de endpoint personalizada para grupos de usuários. Quando você anexa essa política ao seu endpoint de interface, ela concede acesso às ações do grupo de usuários listadas para todos os diretores em todos os recursos.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Principal": {
            "AWS": "arn:aws:iam::123456789012:assumed-role/MyWebAppRole/
MyWebAppSession"
         },
         "Effect": "Allow",
         "Action": [
            "cognito-idp:AdminInitiateAuth",
            "cognito-idp:AdminRespondToAuthChallenge",
            "cognito-idp:AdminSetUserPassword"
         ],
         "Resource":"arn:aws:cognito-idp:us-east-1:123456789012:userpool/us-
east-1_EXAMPLE"
      },
      {
         "Effect": "Allow",
         "Action": [
            "cognito-idp:InitiateAuth",
            "cognito-idp:RespondToAuthChallenge",
            "cognito-idp:ForgotPassword",
            "cognito-idp:ConfirmForgotPassword"
         ],
         "Resource":"arn:aws:cognito-idp:us-east-1:123456789012:userpool/us-
east-1_EXAMPLE"
      }
   ]
}
```

Exemplo: política de VPC endpoint para ações de grupos de identidades

Veja a seguir um exemplo de uma política de endpoint personalizada para grupos de identidades. Essa política usa chaves de contexto específicas do serviço para restringir o acesso a usuários autenticados de um grupo de identidades específico.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": "*",
```

```
         "Action": [
            "cognito-identity:GetId",
            "cognito-identity:GetCredentialsForIdentity",
            "cognito-identity:GetOpenIdToken"
         ],
         "Resource": "*",
         "Condition": {
            "StringEquals": {
               "cognito-identity-auth:IdentityPoolArn": "arn:aws:cognito-identity:us-
east-1:123456789012:identitypool/us-east-1:12345678-ffff-ffff-ffff-123456"
            }
         }
      }
   ]
}
```

Crie uma política baseada em identidade para operações AWS PrivateLink

Políticas baseadas em identidade são recursos do IAM que você pode anexar aos AWS diretores. Você pode controlar o acesso ao Amazon Cognito por meio de VPC endpoints com políticas baseadas em identidade para operações autenticadas pelo IAM. Ao contrário das políticas de endpoint, você não pode configurar permissões para operações não autenticadas em políticas baseadas em identidade. As operações autenticadas ou administrativas exigem a autorização do Signature Version 4. Para grupos de usuários, as operações autenticadas incluem solicitações de autenticação do lado do servidor, como solicitações AdminInitiateAuthadministrativas, como. UpdateUserPool Para grupos de identidades, as operações autenticadas incluem solicitações administrativas como DeleteIdentitiese. DescribeIdentity

Uma política baseada em identidade especifica as seguintes informações:

- As ações que podem ser realizadas.
- Os recursos nos quais as ações podem ser executadas.
- As condições que devem ser satisfeitas antes que a solicitação seja permitida ou negada.

Exemplo: política baseada em identidade para autenticação do lado do servidor do grupo de usuários

O exemplo de política a seguir concede acesso às ações do grupo de usuários listadas no grupo de usuários especificado, a partir do endpoint especificado. Aplique essa política à função assumida do IAM para seu aplicativo web.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cognito-idp:AdminInitiateAuth",
                "cognito-idp:AdminRespondToAuthChallenge",
                "cognito-idp:AdminSetUserPassword"
            ],
            "Resource": "arn:aws:cognito-idp:us-east-1:123456789012:userpool/us-
east-1_EXAMPLE",
            "Condition": {
                "StringEquals": {
                    "aws:SourceVpce": "vpce-1a2b3c4d"
                }
            }
        }
    ]
}
```

Exemplo: política baseada em identidade para operações administrativas de grupos de identidades

O exemplo de política a seguir concede acesso às ações administrativas do grupo de identidades do VPC endpoint especificado. Aplique essa política ao diretor do IAM que precisa realizar a administração do grupo de identidades.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cognito-identity:DeleteIdentities",
                "cognito-identity:DescribeIdentity"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceVpce": "vpce-1a2b3c4d"
                },
                "StringEquals": {
```

```
                    "cognito-identity:IdentityPoolArn": "arn:aws:cognito-identity:us-
east-1:123456789012:identitypool/us-east-1:12345678-ffff-ffff-ffff-123456"
                }
            }
        }
    ]
}
```

Validação de conformidade para o Amazon Cognito

Auditores terceirizados avaliam a segurança e a conformidade do Amazon Cognito como parte de AWS vários programas de conformidade. Isso inclui SOC, PCI, FedRAMP, HIPAA e outros.

Para obter uma lista de AWS serviços no escopo de programas de conformidade específicos, consulte AWS serviços no escopo por programa de conformidade AWS . Para obter informações gerais, consulte Programas de conformidade da AWS.

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Baixar relatórios em AWS Artifact .

Sua responsabilidade relativa à compatibilidade quando for usado o Amazon Cognito é determinada pela confidencialidade dos seus dados, pelos objetivos de compatibilidade da sua empresa e pelas leis e regulamentos aplicáveis. A AWS fornece os seguintes recursos para ajudar com a compatibilidade:

- Guias de início rápido de segurança e conformidade : esses guias de implantação apresentam considerações de arquitetura e etapas para a implantação de ambientes básicos focados na segurança e na conformidade na AWS.
- Documento técnico sobre arquitetura para segurança e conformidade com a HIPAA — Este whitepaper descreve como as empresas podem usar para criar aplicativos compatíveis com a HIPAA. AWS
- AWS recursos de https://aws.amazon.com/compliance/resources/ de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- Avaliação de recursos com as regras do Guia do AWS Config Desenvolvedor — AWS Config; avalia o quão bem suas configurações de recursos estão em conformidade com as práticas internas, as diretrizes do setor e os regulamentos.

- AWS Security Hub CSPM— Esse AWS serviço fornece uma visão abrangente do seu estado de segurança interno, AWS que ajuda você a verificar sua conformidade com os padrões e as melhores práticas do setor de segurança.

Resiliência no Amazon Cognito

A infraestrutura AWS global é construída em torno de AWS regiões e zonas de disponibilidade. As regiões fornecem várias zonas de disponibilidade separadas e isoladas fisicamente, que são conectadas com baixa latência, alta throughput e redes altamente redundantes. Com as zonas de disponibilidade, é possível projetar e operar aplicações e bancos de dados que automaticamente executam o failover entre as zonas sem interrupção. As zonas de disponibilidade são altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de data center tradicionais.

Para obter mais informações sobre AWS regiões e zonas de disponibilidade, consulte infraestrutura AWS global.

Tópicos

Fatores em relação a dados regionais

Cada grupo de usuários do Amazon Cognito é criado em uma AWS região e armazenam os dados do perfil do usuário somente nessa região. Os grupos de usuários podem enviar dados do usuário para uma AWS região diferente, dependendo de como os recursos opcionais são configurados.

- Se a configuração de endereço de e-mail padrão do `no-reply@verificationemail.com` é usada para a verificação de rota dos endereços de e-mail com grupos de usuários do Amazon Cognito, os e-mails são roteados pela mesma região do grupo de usuários associado.
- Se um endereço de e-mail diferente for usado para configurar o Amazon Simple Email Service (Amazon SES) com grupos de usuários do Amazon Cognito, esse endereço de e-mail será roteado AWS pela região associada ao endereço de e-mail no Amazon SES.
- As mensagens de SMS dos grupos de usuários do Amazon Cognito são roteadas pela mesma região do Amazon SNS, salvo indicação contrária em Configuring Email or Phone Verification (Configurar verificação por e-mail ou telefone).

- Se os dados de análise do Amazon Pinpoint forem usados com grupos de usuários do Amazon Cognito, os dados do evento serão encaminhados para a região Leste dos EUA (Norte da Virgínia).

> **Note**
>
> O Amazon Pinpoint está disponível em várias AWS regiões da América do Norte, Europa, Ásia e Oceania. As regiões do Amazon Pinpoint incluem a API do Amazon Pinpoint. Se uma região do Amazon Pinpoint for suportada pelo Amazon Cognito, o Amazon Cognito enviará eventos para projetos do Amazon Pinpoint dentro da mesma região do Amazon Pinpoint. Se uma região não for comportada pelo Amazon Pinpoint, o Amazon Cognito somente poderá enviar eventos na região us-east-1. Para informações detalhadas sobre regiões do Amazon Pinpoint, consulte Endpoints e cotas do Amazon Pinpoint e Usar análise do Amazon Pinpoint com grupos de usuários do Amazon Cognito.

Segurança da infraestrutura no Amazon Cognito

Como um serviço gerenciado, o Amazon Cognito é protegido pela segurança de rede AWS global. Para obter informações sobre serviços AWS de segurança e como AWS proteger a infraestrutura, consulte AWS Cloud Security. Para projetar seu AWS ambiente usando as melhores práticas de segurança de infraestrutura, consulte Proteção de infraestrutura no Security Pillar AWS Well-Architected Framework.

Você usa chamadas de API AWS publicadas para acessar o Amazon Cognito pela rede. Os clientes devem oferecer compatibilidade com:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com perfect forward secrecy (PFS) como DHE (Ephemeral Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Análise de configuração e vulnerabilidade em grupos de usuários do Amazon Cognito

AWS lida com tarefas básicas de segurança, como sistema operacional (SO) convidado e aplicação de patches em bancos de dados, configuração de firewall e recuperação de desastres. Esses procedimentos foram revisados e certificados por terceiros certificados. Para obter mais detalhes, consulte os seguintes recursos da :

- Validação de conformidade para o Amazon Cognito
- Modelo de responsabilidade compartilhada

AWS políticas gerenciadas para o Amazon Cognito

Para adicionar permissões a usuários, grupos e funções, é mais fácil usar políticas AWS gerenciadas do que escrever políticas você mesmo. 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 rapidamente, você pode usar nossas políticas AWS gerenciadas. Essas políticas abrangem casos de uso comuns e estão disponíveis em sua AWS conta. Para obter mais informações sobre políticas AWS gerenciadas, consulte políticas AWS gerenciadas no Guia do usuário do IAM.

AWS os serviços mantêm e atualizam as políticas AWS gerenciadas. Você não pode alterar as permissões nas políticas AWS gerenciadas. Os serviços ocasionalmente acrescentam permissões adicionais a uma política gerenciada pela AWS para oferecer suporte a novos recursos. Esse tipo de atualização afeta todas as identidades (usuários, grupos e funções) em que a política está anexada. É mais provável que os serviços atualizem uma política gerenciada pela AWS quando um novo recurso for iniciado ou novas operações se tornarem disponíveis. Os serviços não removem as permissões de uma política AWS gerenciada, portanto, as atualizações de políticas não violarão suas permissões existentes.

Além disso, AWS oferece suporte a políticas gerenciadas para funções de trabalho que abrangem vários serviços. Por exemplo, a política ReadOnlyAccess AWS gerenciada fornece acesso somente de leitura a todos os AWS serviços e recursos. Quando um serviço lança um novo recurso, AWS adiciona permissões somente de leitura para novas operações e recursos. Para obter uma lista e

descrições das políticas de perfis de trabalho, consulte Políticas gerenciadas pela AWS para perfis de trabalho no Guia do usuário do IAM.

AWS políticas gerenciadas do IAM que concedem acesso ao Amazon Cognito

- `AmazonCognitoPowerUser`: permissões para acessar e gerenciar todos os aspectos dos grupos de identidades e de usuários. Para ver as permissões dessa política, consulte AmazonCognitoPowerUser.
- `AmazonCognitoReadOnly`: permissões para acesso somente leitura aos grupos de identidades e de usuários. Para ver as permissões dessa política, consulte AmazonCognitoReadOnly.
- `AmazonCognitoDeveloperAuthenticatedIdentities`: permissões para o sistema de autenticação se integrar ao Amazon Cognito. Para ver as permissões dessa política, consulte AmazonCognitoDeveloperAuthenticatedIdentities.

Essas políticas são mantidas pela equipe do Amazon Cognito, portanto, mesmo quando novas APIs são adicionadas, seus usuários continuam tendo o mesmo nível de acesso.

> **Note**
>
> Ao criar um banco de identidades, você pode criar automaticamente perfis para acesso de usuários autenticados e convidados. O administrador que cria o banco de identidades com novos perfis do IAM também deve ter permissões do IAM para criar perfis.

Grupos de identidades com acesso de convidado não autenticado aplicam uma política AWS gerenciada adicional como política de sessão para usuários não autenticados. Essa política AWS gerenciada não tem uso administrativo pretendido. Em vez disso, limita o escopo das permissões que você pode aplicar aos usuários convidados no fluxo de autenticação avançado dos bancos de identidades. Para obter mais informações, consulte Perfis do IAM.

AWS políticas gerenciadas do IAM que o Amazon Cognito concede aos usuários convidados

- `AmazonCognitoUnAuthedIdentitiesSessionPolicy` - Em combinação com uma política de sessão em linha, limita as permissões que os administradores do IAM podem conceder aos usuários convidados do banco de identidades. O Amazon Cognito aplica automaticamente essa política às sessões de convidados. Para obter mais informações, consulte A política de sessões AWS gerenciadas para convidados.

Atualizações do Amazon Cognito para políticas gerenciadas AWS

Veja detalhes sobre as atualizações das políticas AWS gerenciadas do Amazon Cognito desde que esse serviço começou a monitorar essas alterações. Para obter alertas automáticos sobre alterações feitas nesta página, inscreva-se no feed RSS na página Document history (Histórico de documentos) do Amazon Cognito.

Alteração	Descrição	Data
`AmazonCognitoPowerUser` : alteração	O Amazon Cognito adicionou novas ações para permitir o uso da operação de AWS End User Messaging SMS API DescribeAccountAttributespara usuários administrativos avançados de grupos de usuários do Amazon Cognito.	27 de fevereiro de 2025
`AmazonCognitoUnAuthedIdentitiesSessionPolicy` : alteração	O Amazon Cognito adicionou novas ações para permitir o uso de AWS Key Management Service usuários não autenticados (convidados) em grupos de identidade.	30 de outubro de 2024
`AmazonCognitoUnAuthedIdentitiesSessionPolicy` : alteração	O Amazon Cognito adicionou novas ações para permitir o uso do Amazon Location Service para usuários não autenticados (convidados) em bancos de identidades.	9 de agosto de 2024

Alteração	Descrição	Data
`AmazonCognitoUnAuthedIdentitiesSessionPolicy` : nova política	Foi adicionada uma política AWS gerenciada para redução do escopo de privilégios de usuários convidados em grupos de identidades.	14 de julho de 2023
`AmazonCognitoPowerUser` e `AmazonCognitoReadOnly` : alteração	Foram adicionadas novas permissões para permitir que usuários avançados visualizem e gerenciem associações da AWS WAF web ACLs com grupos de usuários do Amazon Cognito. Foram adicionadas novas permissões para permitir que usuários somente para leitura visualizem associações da AWS WAF web ACLs com grupos de usuários do Amazon Cognito.	19 de julho de 2022

Alteração	Descrição	Data
`AmazonCognitoPowerUser` : alteração	Adição de uma nova permissão para permitir que o Amazon Cognito chame as operações `PutIdentityPolicy` e `ListConfigurationSets` do Amazon Simple Email Service. Essa alteração permite que grupos de usuários do Amazon Cognito atualizem as políticas de autorização de envio do Amazon SES e apliquem conjuntos de configurações do Amazon SES quando você configura o envio de e-mails em seu grupo de usuários.	17 de novembro de 2021

Alteração	Descrição	Data
`AmazonCognitoPowerUser` : alteração	Adicionada uma nova permissão para permitir que o Amazon Cognito chame a operação `GetSMSSandboxAccountStatus` do Amazon Simple Notification Service. Essa alteração permite que os grupos de usuários do Amazon Cognito decidam se você precisa sair da área restrita para testes do Amazon Simple Notification Service para conseguir enviar mensagens a todos os usuários finais por meio de grupos de usuários.	1º de junho de 2021
O Amazon Cognito começou a monitorar alterações.	O Amazon Cognito começou a monitorar as mudanças em suas políticas AWS gerenciadas.	1.º de março de 2021

Solucionar problemas do Amazon Cognito

Este capítulo fornece soluções para problemas comuns que você pode encontrar ao trabalhar com o Amazon Cognito. As implementações do Amazon Cognito podem enfrentar vários desafios em fluxos de autenticação, configurações de grupos de usuários e configurações de federação de identidades. Se estiver desenvolvendo uma nova aplicação ou mantendo uma existente, este guia de solução de problemas ajudará você a identificar e resolver problemas comuns rapidamente.

Erros de configuração de domínios personalizados

Ao configurar nomes de domínios personalizados no Amazon Cognito, você poderá receber mensagens de erro. Os erros comuns incluem problemas de validação, problemas de certificado ou conflitos de domínio.

`Custom domain is not a valid subdomain`

Esse erro indica um problema com a resolução de DNS para o domínio principal. O Amazon Cognito não é compatível com domínios de primeiro nível e exige que o domínio principal tenha um registro DNS A para validação.

Problema

Esse erro indica um problema com a resolução de DNS para o domínio principal. O Amazon Cognito não é compatível com domínios de primeiro nível e exige que o domínio principal tenha um registro DNS A para validação.

Solução

- Criar um registro A para o domínio principal: você deve criar um registro A na configuração de DNS para o domínio principal do seu domínio personalizado.
  - Exemplo: se seu domínio personalizado for `auth.xyz.yourdomain.com`, o domínio principal será `xyz.yourdomain.com`. Se você quiser configurar `xyz.yourdomain.com` como um domínio personalizado, o principal será `yourdomain.com`.
  - O domínio principal deverá apontar para um endereço IP válido. Se não apontar para um endereço IP real, você poderá usar um endereço IP fictício, como `8.8.8.8`.
- Verificar a propagação de DNS (opcional, mas recomendado): para garantir que seu provedor de DNS tenha propagado a alteração, você pode executar um comando `dig`.

  - Se estiver usando `auth.xyz.yourdomain.com` como domínio personalizado: `dig A xyz.yourdomain.com +short`
  - Se estiver usando `xyz.yourdomain.com` como domínio personalizado: `dig A yourdomain.com +short`
  - O comando deverá retornar o endereço IP que você configurou. Se não retornar, espere até que a alteração tenha se propagado totalmente.
- Remover o registro A do domínio principal: após criar com sucesso o domínio personalizado no Amazon Cognito, você poderá remover o registro A criado para o domínio principal, caso seja fictício.

Para obter mais informações, consulte Usar o próprio domínio para a IU hospedada.

Domain already associated with another user pool

Os nomes de domínio personalizados devem ser exclusivos em todas Contas da AWS as regiões.

Problema

Os nomes de domínio personalizados devem ser exclusivos em todas Contas da AWS as regiões.

Solução

- Para usar o nome de domínio para um novo grupo de usuários, é necessário excluir o domínio personalizado do grupo de usuários ao qual ele está associado atualmente.
- Aguardar após a exclusão: leva algum tempo para o domínio personalizado ser totalmente excluído do primeiro grupo de usuários. Criar um novo domínio personalizado com o mesmo nome imediatamente após a exclusão ainda pode resultar nesse erro. Aguarde alguns minutos antes de tentar novamente.

One or more of the CNAMEs that you provided are already associated with a different resource

Quando você cria um domínio personalizado, o Amazon Cognito cria uma distribuição AWS gerenciada pela Amazon CloudFront . Um nome de domínio só pode ser usado com uma CloudFront distribuição da Amazon. Esse erro ocorre se o nome de domínio já estiver em uso como um domínio alternativo para outra CloudFront distribuição da Amazon.

Problema

Quando você cria um domínio personalizado, o Amazon Cognito cria uma distribuição AWS gerenciada pela Amazon CloudFront. Um nome de domínio só pode ser usado com uma CloudFront distribuição da Amazon. Esse erro ocorre se o nome de domínio já estiver em uso como um domínio alternativo para outra CloudFront distribuição da Amazon.

Solução

- Opção 1: use um nome de domínio diferente para seu domínio personalizado do Amazon Cognito.
- Opção 2: Se você usar o nome de domínio do Amazon Cognito, não o use com outra distribuição da Amazon CloudFront .

The specified SSL certificate doesn't exist

O Amazon Cognito usa a Amazon CloudFront, que exige que o certificado AWS Certificate Manager (ACM) esteja no `us-east-1` (Norte da Virgínia) Região da AWS, independentemente da região do grupo de usuários.

Problema

O Amazon Cognito usa a Amazon CloudFront, que exige que o certificado AWS Certificate Manager (ACM) esteja no `us-east-1` (Norte da Virgínia) Região da AWS, independentemente da região do grupo de usuários.

Solução

- Verificar a região do certificado: confirme se o certificado do ACM está na região `us-east-1`.
- Verificar a validade do certificado: verifique se o certificado selecionado não está expirado.
- Certificados importados: se você importou o certificado para o ACM, verifique se:
  - Ele foi emitido por uma autoridade de certificação pública.
  - Ele inclui a cadeia de certificados correta.
- AWS KMS Verificação de política: uma `deny` declaração explícita em uma política AWS Key Management Service (AWS KMS) para o usuário ou função do IAM que está criando o domínio pode causar esse erro. Especificamente, verifique se há negações explícitas nas ações `kms:DescribeKey`, `kms:CreateGrant` e `kms:*`.

Erro **`Invalid refresh token`**

Problema

Você recebe um erro `Invalid refresh token` ao tentar usar um token de atualização para obter novos tokens de acesso e ID do seu grupo de usuários do Amazon Cognito usando a operação de API `AdminInitiateAuth` ou `InitiateAuth`.

Solução

Implemente as seguintes etapas de solução de problemas com base na configuração do grupo de usuários e no uso da API:

- Garantir um ID do cliente da aplicação consistente: ao chamar a API `AdminInitiateAuth` ou `InitiateAuth` para atualização de token, você deve usar o mesmo ID do cliente da aplicação usado durante a autenticação inicial que gerou o token de atualização.
- Confirme o dispositivo: se você tiver o rastreamento de dispositivos ativado em seu grupo de usuários, mas o dispositivo do usuário não tiver sido confirmado, você deve primeiro chamar a ConfirmDeviceAPI. Depois que o usuário confirmar o dispositivo, você poderá trocar o token de atualização.
- Incluir a chave do dispositivo na solicitação de atualização: se o monitoramento de dispositivos estiver habilitado, inclua a chave exclusiva do dispositivo como um `AuthParameter` ao usar o fluxo `REFRESH_TOKEN_AUTH`:

```
{
  "AuthFlow": "REFRESH_TOKEN_AUTH",
  "AuthParameters": {
    "REFRESH_TOKEN": "example_refresh_token",
    "SECRET_HASH": "example_secret_hash", // Required if your app client uses a
 client secret
    "DEVICE_KEY": "example_device_key"
  }
}
```

- Usar **`USER_SRP_AUTH`** para monitoramento de dispositivos: se você estiver usando o monitoramento de dispositivos, o fluxo de autenticação inicial deverá ser `USER_SRP_AUTH`.

Para obter mais informações, consulte Trabalhar com dispositivos de usuários no grupo de usuários.

Erros de resposta SAML inválida na federação

Os usuários recebem várias `Invalid SAML response` e erros semelhantes ao tentarem se federar no Amazon Cognito usando o SAML 2.0. Esses erros podem ocorrer devido a problemas de mapeamento de atributos, problemas de certificado ou incompatibilidades de configuração.

`Invalid user attributes: Required attribute`

Problema

Um usuário não tem um valor para um atributo obrigatório no grupo de usuários, ou o IdP está tentando remover ou atualizar um atributo imutável.

Solução

- Verifique os Atributos obrigatórios definidos na configuração do grupo de usuários.
- Usando ferramentas de captura de rede em seu navegador, recupere a resposta SAML. Talvez seja necessário realizar a decodificação de URL e base64. Verifique se o atributo está presente na declaração SAML.
- Faça login no provedor de identidades e analise os atributos que ele está enviando para o Amazon Cognito. Verifique se o IdP está configurado para enviar o atributo obrigatório usando o nome correto.
- Para atributos imutáveis, execute o seguinte AWS CLI comando para identificá-los:. `aws cognito-idp describe-user-pool --user-pool-id USER-POOL-ID --query 'UserPool.SchemaAttributes[?Mutable==`false`].Name'`
- Nos mapeamentos de atributos SAML do IdP, exclua qualquer mapeamento que tenha como alvo um atributo imutável do Amazon Cognito. Como alternativa, atualize o atributo de destino para um atributo diferente e mutável.

`Invalid SAML response received: SAML Response signature is invalid`

Problema

O IdP atualizou seu certificado de assinatura SAML, causando uma incompatibilidade entre o certificado na resposta SAML e o arquivo de metadados armazenado no Amazon Cognito.

Solução

1. Baixe o arquivo de metadados mais recente do seu IdP.
2. No console do Amazon Cognito, navegue até Provedores sociais e externos do grupo de usuários, edite o provedor SAML e substitua o arquivo de metadados existente pelo arquivo recém-baixado.

`Audience restriction` ou `Application with identifier not found`

Problema

Um ID de entidade incorreto está configurado no IdP ou a declaração usa o nome de recurso uniforme (URN) de outro grupo de usuários.

Solução

1. Obtenha o ID do grupo de usuários do Amazon Cognito na seção Visão geral no console.
2. No console de gerenciamento do IdP, atualize o ID da entidade na aplicação SAML para o grupo de usuários. Configure o ID da entidade para corresponder ao formato `urn:amazon:cognito:sp:`*`USER_POOL_ID`*. Substitua `USER_POOL_ID` pelo ID do grupo de usuários da etapa anterior.

`An error was encountered with the requested page`

Problema

O URL do Assertion Consumer Service (ACS) registrado com o IdP está configurado incorretamente ou o IdP não está enviando a resposta SAML usando a associação POST obrigatória.

Solução

- No console de gerenciamento do IdP, atualize a aplicação com o formato de URL do ACS correto, garantindo que ele use a vinculação `HTTP POST`.
- Formato de domínio padrão: `https://cognito-idp.`*`Region`*`.amazonaws.com/`*`your user pool ID`*`/saml2/idpresponse`
- Formato de domínio personalizado: `https://`*`auth.example.com`*`/saml2/idpresponse`

`Invalid relayState from identity provider`

Problema

O parâmetro `RelayState` está ausente ou é inválido, ou o URL é incompatível entre o IdP e o Amazon Cognito.

Solução

- Para fluxos iniciados pelo provedor de serviços (iniciados pelo SP): sempre inicie a autenticação no endpoint `/oauth2/authorize` do grupo de usuários. As declarações SAML que não retornam parâmetros `RelayState` da visita inicial do usuário ao Amazon Cognito não são solicitações válidas iniciadas pelo SP.
- Para fluxos iniciados pelo provedor de identidades (iniciados pelo IdP): o IdP deve incluir o parâmetro `RelayState` com a declaração SAML no endpoint `/saml2/idpresponse`, usando o formato obrigatório: `redirect_uri=REDIRECT_URI&state=STATE`.

Para obter mais informações, consulte Como usar provedores de identidade SAML com um grupo de usuários.

Usuários de login gerenciado não podem selecionar um fator de MFA

Problema

Os usuários não conseguem escolher seu método preferencial de MFA ao fazer login por meio do login gerenciado, ou não estão sendo solicitados a usar o método de MFA esperado.

Solução

O Amazon Cognito segue uma lógica específica para determinar quais fatores de MFA estão disponíveis para os usuários com base nas configurações do grupo de usuários, nos atributos do usuário e na configuração de recuperação da conta. Por exemplo, os usuários não podem usar a MFA por e-mail se o e-mail estiver configurado como o método principal de recuperação da conta.

Para substituir a seleção padrão do fator de MFA, você pode implementar uma dessas abordagens:

- Para aplicativos públicos: use SetUserMFAPreferencecom um token de acesso válido para permitir que os usuários definam suas próprias preferências de MFA

- Para aplicativos gerenciados pelo administrador: use AdminSetUserMFAPreferencecom AWS credenciais para configurar as preferências de MFA do usuário

Ambas as operações permitem que você habilite ou desabilite os métodos de MFA por SMS, e-mail e TOTP para usuários individuais e defina um método como preferencial.

Para obter mais informações, consulte Adicionar MFA a um grupo de usuários.

Usuários sem senha e com chave de acesso não conseguem usar a MFA

Problema

Os usuários que fazem login com métodos de autenticação de senha de uso único (OTP) não podem adicionar ou usar a autenticação multifator.

Solução

Os fluxos de login OTP não são compatíveis com MFA. No entanto, a autenticação por chave de acesso com verificação do usuário pode atender aos requisitos de MFA. `FactorConfiguration`Defina como `MULTI_FACTOR_WITH_USER_VERIFICATION` em seu grupo de usuários `WebAuthnConfiguration` para permitir que as chaves de acesso sejam contabilizadas como autenticação multifatorial.

Para mais informações, consulte Autenticação com grupo de usuários.

Não é possível receber o código de redefinição de senha por e-mail/SMS

Problema

Os usuários não recebem códigos de verificação durante o fluxo de trabalho de esquecimento de senha por e-mail ou SMS.

Solução

Há vários motivos pelos quais os códigos de verificação podem não ser recebidos. Para solucionar esse problema, siga esta lista de verificação:

- Verifique as pastas de spam e lixo eletrônico do usuário.
- Confirme se o usuário existe no grupo de usuários.

- Verifique se o status do usuário não é `FORCE_CHANGE_PASSWORD`. Se esse for o caso, o usuário foi criado por um administrador e o Amazon Cognito solicitará que ele defina uma senha quando fizer login com a senha temporária.
- Verifique se o usuário tem um atributo de e-mail ou telefone verificado.
- Verifique o limite de gastos da sua conta para mensagens SMS.
- Verifique a cota de envio de mensagens do Amazon Simple Email Service (Amazon SES).
- Verifique se o SMS e o Amazon SES (se o grupo de usuários estiver configurado para Enviar e-mail com o Amazon SES) foram retirados do sandbox. Se não tiverem sido retirados do sandbox, endereços de e-mail ou números de telefone que não foram verificados pelo Amazon SES ou não AWS End User Messaging SMS poderão receber códigos de redefinição de senha.
- Se o usuário tiver um fator de MFA ativo, verifique se ele não está tentando gerar uma mensagem para o mesmo fator. Por exemplo, usuários com MFA por e-mail ativa não podem enviar códigos de redefinição de senha por e-mail.

Para obter mais informações, consulte Configurações de e-mail para grupos de usuários do Amazon Cognito e Configurações de mensagens SMS para grupos de usuários do Amazon Cognito.

A redefinição de senha falha com atributos de recuperação não verificados: **`Could not reset password for the account, please contact support or try again`**

Problema

Os usuários recebem esse erro ao tentar redefinir a senha devido a métodos de recuperação não verificados ou conflitos de configuração de MFA. O erro ocorre quando o atributo de e-mail ou número de telefone do usuário não é verificado ou quando suas configurações de MFA impedem o uso do método de recuperação configurado.

Solução

Analise a configuração de recuperação da conta do seu grupo de usuários e o status de verificação do usuário afetado:

- Verifique as configurações de recuperação: em seu grupo de usuários, navegue até Entrar > Recuperação de conta de usuário. Verifique se a recuperação da conta de autoatendimento está ativada e revise a configuração do método de entrega da mensagem de recuperação.

- Verifique a verificação do atributo do usuário: verifique se o usuário tem um endereço de e-mail ou número de telefone verificado que corresponda à configuração do seu método de recuperação. No console, acesse o perfil do usuário, selecione Atributos do usuário > Editar e marque o atributo apropriado como verificado.
- Resolver conflitos de MFA: usuários cujo método preferido de MFA é e-mail não podem receber códigos de redefinição de senha por e-mail, e usuários com MFA por SMS não podem receber códigos por SMS. Atualize seu método de entrega de mensagens de recuperação para fornecer opções alternativas, como SMS, se disponível, caso contrário, e-mail ou e-mail, se disponível, caso contrário, SMS.
- Verificação administrativa: use a operação da API AdminUpdateUserAttributespara verificar programaticamente os atributos do usuário quando o acesso ao console não estiver disponível.

Para obter mais informações, consulte Senhas, recuperação de contas e políticas de senha.

Erros do **SECRET_HASH**

Problema

Solicitações de API de autenticação para clientes da aplicação com segredos do cliente retornam erros como `An error occurred (NotAuthorizedException) when calling the ForgotPassword operation: Client 1example23456789 is configured with secret but SECRET_HASH was not received.`

Solução

Sua aplicação deve calcular o `SECRET_HASH` para o usuário atual, o cliente da aplicação e o segredo do cliente. O método de cálculo é:

```
Base64 ( HMAC_SHA256 ( "client secret", "Username" + "Client Id" ) )
```

Para obter o segredo do cliente:

1. Abra o console do Amazon Cognito e navegue até o cliente da aplicação pelo menu Clientes da aplicação. Na seção Informações do cliente de aplicação, localize Segredo do cliente. Selecione Mostrar segredo do cliente e o segredo do cliente da aplicação será exibido.
2. Gere uma DescribeUserPoolClientsolicitação. O segredo do cliente estará incluído na resposta.

Para obter mais informações, consulte Computar valores de hash de segredo.

O console do Amazon Cognito escolhe uma configuração padrão para um novo grupo de usuários

Problema

Quando você configura um novo grupo de usuários no console, o Amazon Cognito escolhe várias configurações padrão para você. Algumas configurações não podem ser alteradas depois que o grupo de usuários é criado. Como você pode fazer escolhas informadas e entender o que o Amazon Cognito selecionou automaticamente?

Solução

A nova configuração do grupo de usuários no console do Amazon Cognito foi projetada para testes e prototipagem rápidos. O console apresenta somente as opções de configuração mais críticas, aquelas que não podem ser alteradas após a criação do grupo de usuários. Todas as outras configurações que o Amazon Cognito configura automaticamente podem ser modificadas posteriormente.

Recomendamos a seguinte abordagem:

1. Use o console para criar grupos de usuários de teste enquanto refina sua implementação.
2. Após determinar a configuração de produção, aplique essas configurações aos grupos de usuários de teste.
3. Use operações DescribeUserPoolde DescribeUserPoolClientAPI para gerar modelos JSON de sua configuração testada.
4. Use esses modelos com ferramentas de implantação como CDK, API REST ou CloudFormation para criar seus recursos de produção. AWS SDKs

Para obter mais informações, consulte Conceitos básicos dos grupos de usuários.

Recursos adicionais para solução de problemas

Para obter orientações adicionais sobre soluções de problemas e soluções fornecidas pela comunidade, você também pode explorar os seguintes recursos externos:

- AWS re:post Comunidade do Amazon Cognito — Procure perguntas e soluções da comunidade

- AWS Centro de conhecimento > Artigos do Amazon Cognito - Artigos de solução de problemas selecionados

Como marcar recursos do Amazon Cognito

Uma tag é um rótulo de metadados que você atribui ou AWS atribui a um AWS recurso. Cada tag consiste em uma chave e um valor. Em tags atribuídas por você, você mesmo define a chave e o valor. Por exemplo, você pode definir a chave como `stage` e o valor de um atributo como `test`.

As tags ajudam a:

- Identifique e organize seus AWS recursos. Muitos AWS serviços oferecem suporte à marcação, para que você possa atribuir a mesma tag a recursos de serviços diferentes. Isso ajuda você a indicar quais recursos estão relacionados. Por exemplo, é possível atribuir a mesma tag a um grupo de usuários do Amazon Cognito atribuída a uma tabela do Amazon DynamoDB.
- Acompanhe seus AWS custos. Você pode ativar essas tags no Gerenciamento de Faturamento e Custos da AWS painel. AWS usa etiquetas de alocação de custos para categorizar seus custos e entregar um relatório mensal de alocação de custos para você. Para mais informações, consulte Usar etiquetas de alocação de custos no Guia do usuário do AWS Billing .
- Controlar o acesso aos recursos de acordo com as tags atribuídas a eles. É possível controlar o acesso especificando chaves e valores de etiquetas nas condições para uma política do AWS Identity and Access Management (IAM). Por exemplo, você poderia permitir que um usuário atualizasse um grupo de usuários somente se esse grupo tiver uma tag `owner` com o valor do nome desse usuário. Para mais informações, consulte Controlar o acesso usando etiquetas no Guia do usuário do IAM.

Você pode usar a API do Amazon Cognito AWS Command Line Interface ou a API do Amazon Cognito para adicionar, editar ou excluir tags para grupos de usuários e identidades. Também é possível gerenciar etiquetas para grupos de usuários usando o console do Amazon Cognito.

Para obter dicas sobre como usar tags, consulte a postagem AWS Tagging Strategies no blog AWS Answers.

As seções a seguir fornecem mais informações sobre tags para o Amazon Cognito.

Recursos compatíveis no Amazon Cognito

Os seguintes recursos do Amazon Cognito são compatíveis com a marcação:

- Grupos de usuários

- Bancos de identidades

Restrições de tags

As restrições a seguir se aplicam às etiquetas nos recursos do Amazon Cognito:

- Número máximo de tags que você pode atribuir a um recurso: 50
- Comprimento máximo da chave: 128 caracteres Unicode
- Comprimento máximo do valor: 256 caracteres Unicode
- Caracteres válidos para chaves e valores: a-z, A-Z, 0-9, espaço, e os seguintes caracteres: _ . : / = + - @
- Chaves e valores diferenciam maiúsculas de minúsculas
- Não use `aws:` como um prefixo para chaves, pois ele é reservado para uso da AWS

Como gerenciar etiquetas usando o console do Amazon Cognito

Você pode usar o console do Amazon Cognito para gerenciar as tags atribuídas aos seus grupos de usuários.

Para adicionar etiquetas a um grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente na lista ou crie um grupo de usuários.
4. Clique no menu Configurações e localize a guia Etiquetas.
5. Selecione Add tags (Adicionar etiqueta) para adicionar sua primeira etiqueta. Se tiver atribuído etiquetas anteriormente a esse grupo de usuários, em Manage tags (Gerenciar etiquetas), selecione Add another (Adicionar outra).
6. Especifique valores para Tag Key (Chave de tags) e Tag Value (Valor da tag).
7. Para cada etiqueta adicional que quiser inserir, escolha Add another (Adicionar outra).
8. Quando terminar de adicionar etiquetas, escolha Save changes (Salvar alterações).

Para marcar um banco de identidades, navegue até o menu Grupos de identidades e selecione ou crie um banco de identidades. Na guia Propriedades do grupo de identidades, localize Etiquetas. Escolha Adicionar Tag.

AWS CLI exemplos

AWS CLI Ele fornece comandos que ajudam você a gerenciar as tags que você atribui aos grupos de usuários e grupos de identidades do Amazon Cognito.

Atribuir tags

Use os comandos a seguir para atribuir tags aos seus grupos de usuários e de identidades já existentes.

Example Comando **`tag-resource`** para grupos de usuários

Atribua tags a um grupo de usuários utilizando `tag-resource` no conjunto de comandos `cognito-idp`:

```
$ aws cognito-idp tag-resource \
> --resource-arn user-pool-arn \
> --tags Stage=Test
```

Esse comando inclui os seguintes parâmetros:

- `resource-arn`: o nome de recurso da Amazon (ARN) do grupo de usuários ao qual você está aplicando tags. Para examinar o ARN, escolha o grupo de usuários no console do Amazon Cognito e visualize o valor de Pool ARN (ARN do grupo) na guia General settings (Configurações gerais).
- `tags`: os pares chave-valor das etiquetas, no formato *`key`*=*`value`*.

Para atribuir várias tags ao mesmo tempo, você deve especificá-las em uma lista separada por vírgulas:

```
$ aws cognito-idp tag-resource \
> --resource-arn user-pool-arn \
> --tags Stage=Test,CostCenter=80432,Owner=SysEng
```

Example Comando **`tag-resource`** para grupos de identidades

Atribua tags a um grupo de identidades utilizando `tag-resource` no conjunto de comandos `cognito-identity`:

```
$ aws cognito-identity tag-resource \
> --resource-arn identity-pool-arn \
> --tags Stage=Test
```

Esse comando inclui os seguintes parâmetros:

- `resource-arn`: o nome do recurso da Amazon (ARN) do grupo de identidades ao qual você está aplicando tags. Para pesquisar o ARN, escolha o grupo de identidades no console do Amazon Cognito e selecione Edit identity pool (Editar grupo de identidades). Depois, em Identity pool ID (ID do grupo de identidades), selecione Show ARN (Mostrar ARN).
- `tags`: os pares chave-valor das etiquetas, no formato *`key`*=*`value`*.

Para atribuir várias tags ao mesmo tempo, você deve especificá-las em uma lista separada por vírgulas:

```
$ aws cognito-identity tag-resource \
> --resource-arn identity-pool-arn \
> --tags Stage=Test,CostCenter=80432,Owner=SysEng
```

Visualizar tags

Use os comandos a seguir para visualizar as tags que você atribuiu aos seus grupos de usuários e de identidades.

Example Comando **`list-tags-for-resource`** para grupos de usuários

Visualize as tags atribuídas a um grupo de usuários utilizando `list-tags-for-resource` no conjunto de comandos `cognito-idp`:

```
$ aws cognito-idp list-tags-for-resource --resource-arn user-pool-arn
```

Example Comando **`list-tags-for-resource`** para grupos de identidades

Visualize as tags atribuídas a um grupo de identidades utilizando `list-tags-for-resource` no conjunto de comandos `cognito-identity`:

```
$ aws cognito-identity list-tags-for-resource --resource-arn identity-pool-arn
```

Remover tags

Use os comandos a seguir para remover tags de seus grupos de usuários e de identidades.

Example Comando **`untag-resource`** para grupos de usuários

Remova tags de um grupo de usuários utilizando `untag-resource` no conjunto de comandos `cognito-idp`:

```
$ aws cognito-idp untag-resource \
> --resource-arn user-pool-arn \
> --tag-keys Stage CostCenter Owner
```

Para o parâmetro `--tag-keys`, especifique uma ou mais chaves de etiqueta. Não inclua os valores das etiquetas. Separe as chaves com espaços.

Example Comando **`untag-resource`** para grupos de identidades

Remova tags de um grupo de identidades utilizando `untag-resource` no conjunto de comandos `cognito-identity`:

```
$ aws cognito-identity untag-resource \
> --resource-arn identity-pool-arn \
> --tag-keys Stage CostCenter Owner
```

Para o parâmetro `--tag-keys`, especifique uma ou mais chaves de etiqueta. Não inclua os valores das etiquetas.

> **Important**
>
> Após excluir um usuário ou grupo de identidades, as etiquetas relacionadas ao grupo excluído ainda poderão aparecer no console ou em chamadas de API por até 30 dias após a exclusão.

Aplicar tags durante a criação de recursos

Use os comandos a seguir para atribuir tags no momento em que você cria um grupo de usuários ou um grupo de identidades.

Example Comando **`create-user-pool`** com etiquetas

Quando você cria um grupo de usuários utilizando o comando `create-user-pool`, pode especificar tags com o parâmetro `--user-pool-tags`:

```
$ aws cognito-idp create-user-pool \
> --pool-name user-pool-name \
> --user-pool-tags Stage=Test,CostCenter=80432,Owner=SysEng
```

Os pares de chave-valor para etiquetas devem estar no formato *`key`*=*`value`*. Se estiver adicionando várias etiquetas, você deve especificá-las em uma lista separada por vírgulas.

Example Comando **`create-identity-pool`** com etiquetas

Quando você cria um grupo de identidades utilizando o comando `create-identity-pool`, pode especificar tags com o parâmetro `--identity-pool-tags`:

```
$ aws cognito-identity create-identity-pool \
> --identity-pool-name identity-pool-name \
> --allow-unauthenticated-identities \
> --identity-pool-tags Stage=Test,CostCenter=80432,Owner=SysEng
```

Os pares de chave-valor para etiquetas devem estar no formato *`key`*=*`value`*. Se estiver adicionando várias etiquetas, você deve especificá-las em uma lista separada por vírgulas.

Como gerenciar etiquetas usando a API do Amazon Cognito

Você pode usar as ações a seguir na APIos do Amazon Cognito para gerenciar as tags dos seus grupos de usuários e de identidades.

Ações de API para etiquetas de grupo de usuários

Use as ações de API a seguir para atribuir, visualizar e remover tags de grupos de usuários.

- TagResource

- ListTagsForResource
- UntagResource
- CreateUserPool

Ações de API para etiquetas de grupo de identidades

Use as ações de API a seguir para atribuir, visualizar e remover tags de grupos de identidades.

- TagResource
- ListTagsForResource
- UntagResource
- CreateIdentityPool

Cotas no Amazon Cognito

O Amazon Cognito tem cotas padrão, anteriormente chamadas de limites, para o número máximo de operações que você pode executar em sua conta. O Amazon Cognito também tem cotas para o número máximo e o tamanho dos recursos do Amazon Cognito.

Cada cota do Amazon Cognito representa um volume máximo de solicitações em uma em uma Região da AWS . Conta da AWS Por exemplo, as aplicações podem fazer solicitações de API até a taxa de cota padrão (RPS) para operações `UserAuthentication` em todos os grupos de usuários na região Leste dos EUA (Norte da Virgínia). Seus aplicativos na Ásia-Pacífico (Tóquio) podem produzir o mesmo volume de solicitações em todos os grupos de usuários em sua própria região. AWS só pode conceder uma solicitação de aumento de cota em uma região por vez. Um aumento bem-sucedido da cota na região Leste dos EUA (Norte da Virgínia) não afeta a taxa máxima de solicitação na região Ásia-Pacífico (Tóquio).

Tópicos

Noções básicas das cotas de taxas de solicitação de API

Categorização de cotas

O Amazon Cognito aplica uma taxa máxima de solicitação para operações de API. Para mais informações sobre operações de API que o Amazon Cognito disponibiliza, consulte os guias de referência de API para grupos de usuários e bancos de identidades. Nos grupos de usuários, essas operações são agrupadas em categorias de casos de uso comuns, como `UserAuthentication` ou `UserCreation`. Para obter uma lista das operações de API por grupo de usuários, consulte Categorias de operação da API de grupos de usuários do Amazon Cognito e cotas de taxa de solicitação.

No Console do Service Quotas, é possível monitorar a utilização de cotas por categoria, grupos de usuários e bancos de identidades. Se a taxa de solicitação de seus grupos de usuários do Amazon Cognito ou exceder uma cota, será possível adquirir capacidade adicional. É possível monitorar o uso da cota do grupo de usuários por categoria e adquirir aumentos no console do Service Quotas.

As cotas de operação são definidas como o número de solicitações permitidas por segundo (RPS) para todas as operações em uma categoria. O serviço de grupos de usuários do Amazon Cognito aplica cotas a todas as operações em cada categoria. Por exemplo, a categoria `UserCreation` inclui quatro operações, `SignUp`, `ConfirmSignUp`, `AdminCreateUser` e `AdminConfirmSignUp`. Ela é alocada com uma cota combinada de 50 RPS. Se várias operações ocorrerem ao mesmo tempo, cada uma delas dentro dessa categoria pode chamar até 50 RPS separadamente ou de maneira combinada.

> **Note**
>
> As cotas de categoria só se aplicam aos grupos de usuários. O Amazon Cognito aplica cada cota do grupo de identidades a uma única operação. Para cotas de taxa de solicitação por categoria e por operação, AWS mede a taxa agregada de todas as solicitações de todos os grupos de usuários ou grupos de identidades em sua Conta da AWS região.

Operações de API de grupos de usuários do Amazon Cognito com processamento especial de taxa de solicitação

As cotas de operação são medidas e aplicadas para o total combinado de solicitações no nível da categoria, com exceção das operações `AdminRespondToAuthChallenge` e `RespondToAuthChallenge`, em que regras especiais de processamento são aplicadas.

A categoria `UserAuthentication` inclui quatro operações na API de grupos de usuários do Amazon Cognito: `AdminInitiateAuth`, `InitiateAuth`, `AdminRespondToAuthChallenge` e `RespondToAuthChallenge`. Além disso, a autenticação do usuário na interface hospedada contribui para essa cota. As operações `InitiateAuth` e `AdminInitiateAuth` são medidas e aplicadas por cota de categoria. As operações correspondentes `RespondToAuthChallenge` e `AdminRespondToAuthChallenge` estão sujeitas a uma cota separada que é três vezes o limite da categoria `UserAuthentication`. Essa cota elevada acomoda vários desafios de autenticação configurados nas aplicações. A cota é suficiente para abranger a grande maioria dos casos de uso. Depois que sua aplicação responde até três desafios de autenticação, outras solicitações contam

para a cota da categoria `UserAuthentication`. A autenticação multifator (MFA), a autenticação de dispositivos e a autenticação personalizada são exemplos de solicitações de desafio que você pode criar em seu grupo de usuários.

Por exemplo, se sua cota para a categoria `UserAuthentication` for 80 RPS, você poderá chamar `RespondToAuthChallenge` ou `AdminRespondToAuthChallenge` a uma taxa de até 240 RPS (80 RPS x 3). Se seu grupo de usuários solicitar quatro rodadas de desafio por autenticação e 70 usuários fizerem login por segundo, o total de `RespondToAuthChallenge` será 280 RPS (70 x 4), que representa 40 RPS acima da cota. Os 40 RPS extras são adicionados a 70 chamadas `InitiateAuth`, totalizando o uso da categoria `UserAuthentication` em 110 RPS (40 + 70). Como esse valor excede em 30 RPS a cota de categoria definida em 80 RPS, o Amazon Cognito limita as solicitações da aplicação.

Usuários ativos mensalmente

Quando o Amazon Cognito calcula a cobrança do grupo de usuários, ele cobra uma taxa para cada usuário ativo mensal (MAU). Considere sua contagem de MAU atual e projetada em seu planejamento para solicitações de aumento de cota. Um usuário é considerado como MAU se, em determinado mês, houver uma operação de identidade relacionada a esse usuário. Ao vincular usuários federados a usuários locais com a federação SAML ou OIDC, o usuário local será contabilizado como um MAU de diretório corporativo ou `EnterpriseMAU`, independentemente de o usuário fazer login de forma direta ou por meio de federação. Para obter mais informações, consulte Preços do Amazon Cognito.

- Cadastro ou criação administrativa de um usuário. A importação de CSV do usuário não contribui para a contagem de MAU.
- Confirmação da conta do usuário ou verificação de atributos.
- Login e resposta do desafio. As operações que você autoriza com o token de acesso do usuário atualmente conectado não contribuem para sua contagem de MAU. No entanto, como o login produz tokens de acesso, essas operações indicam que o usuário associado é um MAU.
- Logout e revogação do token.
- Redefinição de senhas por autoatendimento e configuração de senhas de usuário como administrador. A redefinição de senhas de usuário como administrador (AdminResetUserPassword) não contribui para sua contagem de MAU.
- Alteração de atributos de usuário ou associação do grupo.
- Consulta de atributos detalhados de um usuário como administrador.

> **Note**
>
> A categoria Consultar atributos detalhados de um usuário como administrador inclui a operação da API AdminGetUser, mas não ListUsers. Uma user-by-user consulta detalhada em um grande grupo de usuários pode ter um impacto significativo na sua AWS fatura. Para evitar custos adicionais, colete dados do usuário com `ListUsers` ou armazene informações do usuário em um banco de dados externo.

Você não receberá cobranças por sessões adicionais de nenhum usuário ativo, nem por usuários que não estiveram ativos em um mês do calendário. Em um mês em que você alterou seu plano de recursos do grupo de usuários entre as opções disponíveis de Lite, Essentials e Plus, sua fatura desse mês é calculada a partir da soma dos usuários ativos mensais (MAUs) em cada nível, com cada MAU atribuído ao nível atribuído com o preço mais alto quando o usuário estava ativo. Por exemplo:

1. No início do mês, seu grupo de usuários está no plano de recursos Plus.
2. O usuário A faz login no primeiro dia do mês.
3. O usuário B faz login no primeiro e no último dia do mês.
4. No décimo dia do mês, você muda seu plano de recursos para o Essentials.
5. O usuário C faz login no último dia do mês.

Nesse cenário, o usuário A e o usuário B são Plus MAUs e o usuário C é um MAU do Essentials.

MAU Lite

Um usuário que esteve ativo pelo menos uma vez em um mês quando o grupo de usuários constava no plano de recursos Lite e nunca esteve ativo quando o grupo de usuários constava nos planos Essentials ou Plus.

MAU Essentials

Um usuário que esteve ativo pelo menos uma vez em um mês quando o grupo de usuários constava no plano de recursos Essentials e nunca esteve ativo quando o grupo de usuários constava no plano Plus.

MAU Plus

Um usuário que esteve ativo pelo menos uma vez em um mês quando o grupo de usuários constava no plano Plus.

Para obter mais informações, consulte Planos de recursos de grupos de usuários.

Gerenciar cotas de taxas de solicitação de API

Identificar os requisitos de cota

Important

Se você aumentar as cotas do Amazon Cognito para categorias como `UserAuthentication`, `UserCreation` ou `AccountRecovery`, talvez seja necessário aumentar as cotas para outros Serviços da AWS. Por exemplo, as mensagens enviadas pelo Amazon Cognito com o Amazon Simple Notification Service (Amazon SNS) ou o Amazon Simple Email Service (Amazon SES) podem falhar se as cotas de taxa de solicitação forem insuficientes nesses serviços.

Para calcular os requisitos de cota, determine quantos usuários ativos interagirão com sua aplicação em um período específico. Por exemplo, se você espera que a aplicação faça login em uma média de um milhão de usuários ativos em um período de oito horas, você precisará autenticar uma média de 35 usuários por segundo.

Além disso, presumindo que a sessão média do usuário seja de duas horas e os tokens sejam configurados para expirar após uma hora, cada usuário deve atualizar seus tokens uma vez durante essa sessão. A cota média necessária para a categoria `UserAuthentication` suportar essa carga é de 70 RPS.

Se você assumir uma peak-to-average proporção de 3:1 considerando a variação da frequência de login do usuário durante o período de oito horas, precisará da cota desejada de 200 RPS. `UserAuthentication`

> **Note**
>
> Se você chamar várias operações para cada ação do usuário, precisará resumir as taxas de chamada de operação individuais no nível da categoria.

Otimizar as taxas de solicitação para limites de cota

Como o aumento dos limites de taxa da API adiciona custos à sua AWS fatura, considere ajustes em seu modelo de uso antes de solicitar um aumento de cota. Veja a seguir alguns exemplos de arquitetura de aplicação que otimiza as taxas de solicitação.

Repetir a tentativa após um período de espera de recuo

Você pode detectar erros em cada chamada de API e, em seguida, tentar novamente após um período de recuo. Você pode ajustar o algoritmo de recuo de acordo com as necessidades da empresa e com a carga. A Amazon SDKs tem uma lógica de repetição integrada. Para obter mais informações, consulte Ferramentas para desenvolver AWS.

Usar um banco de dados externo para atributos atualizados com frequência

Se a aplicação exigir várias chamadas para um grupo de usuários para a leitura ou gravação de atributos personalizados, use armazenamento externo. Você pode usar seu banco de dados preferido para armazenar atributos personalizados ou usar uma camada de cache para carregar um perfil de usuário durante o login. Você pode referenciar esse perfil do cache quando necessário, em vez de recarregar o perfil do usuário de um grupo de usuários.

Validar tokens web JSON (JWTs) no lado do cliente

As aplicações têm que validar tokens JWT antes de confiar neles. Você pode verificar a assinatura e a validade dos tokens no lado do cliente sem enviar solicitações de API para um grupo de usuários. Depois que o token é validado, você pode confiar em solicitações no token e usar as solicitações em vez de fazer mais chamadas de API de `getUser`. Para obter mais informações, consulte Como verificar um token Web JSON.

Acelerar o tráfego para sua aplicação Web com uma sala de espera

Se você espera tráfego de um grande número de usuários fazendo login durante um evento de tempo limitado, como fazer um exame ou participar de um evento ao vivo, você pode otimizar o tráfego de solicitações com mecanismos de autocontrole de utilização. Você pode, por exemplo, configurar uma sala de espera onde os usuários podem ficar até que uma sessão

esteja disponível, permitindo que você processe solicitações quando tiver capacidade disponível. Consulte Solução AWS Virtual Waiting Room para uma arquitetura de referência de uma sala de espera.

Cache JWTs

Reutilize os tokens de acesso até que eles expirem. Para ver um exemplo de estrutura com armazenamento em cache de tokens em um API Gateway, consulte Gerenciar a expiração e o armazenamento em cache do token do grupo de usuários. Em vez de gerar solicitações de API para consultar informações do usuário, armazene os tokens de ID em cache até que eles expirem e leia os atributos do usuário no cache.

Para obter mais informações sobre como trabalhar com taxas de solicitação de API em AWS, consulte Gerenciamento e monitoramento da limitação de API em suas cargas de trabalho. Para obter informações sobre como otimizar as operações do Amazon Cognito que adicionam custos à AWS sua fatura, consulte. Gerenciar custos

Rastrear o uso da cota

O Amazon Cognito gera `CallCount` `ThrottleCount` métricas na Amazon CloudWatch para cada categoria de operação de API no nível da conta. Você pode usar o `CallCount` para rastrear o número total de chamadas feitas pelos clientes relacionadas a uma categoria. Você pode usar o `ThrottleCount` para rastrear o número total de chamadas com controle de utilização feitas pelos clientes relacionadas a uma categoria. Você pode usar as métricas `CallCount` e `ThrottleCount` usando a estatística `Sum` para contar o número total de chamadas em uma categoria. Para obter mais informações, consulte métricas CloudWatch de uso.

Ao monitorar cotas de serviço, a utilização é a porcentagem de uma cota de serviço em uso. Por exemplo, se o valor da cota for de 200 recursos e 150 recursos estiverem em uso, a utilização será de 75%. Uso é o número de recursos ou operações em uso para uma cota de serviço.

Acompanhamento do uso por meio de CloudWatch métricas

Você pode rastrear e coletar métricas de utilização de grupos de usuários do Amazon Cognito com. CloudWatch O CloudWatch painel exibe métricas sobre tudo AWS service (Serviço da AWS) o que você usa. Com CloudWatch, você pode criar alarmes métricos para notificá-lo ou alterar um recurso específico que você está monitorando. Para obter mais informações sobre CloudWatch métricas, consulte Rastrear suas métricas CloudWatch de uso.

Monitorar a utilização por meio de métricas do Service Quotas

Os grupos de usuários do Amazon Cognito estão integrados ao Service Quotas, que é uma interface de console que permite visualizar e gerenciar o uso da cota de serviço. No console Service Quotas, você pode pesquisar o valor de uma cota específica, visualizar informações de monitoramento, solicitar um aumento de cota ou configurar alarmes. CloudWatch Depois que sua conta estiver ativa por um tempo, você poderá visualizar um gráfico da utilização dos seus recursos.

A coluna Valor da cota aplicada em nível de conta no console do Service Quotas para grupos de usuários e bancos de identidades do Amazon Cognito exibe sua cota atual. A coluna Utilização exibe sua taxa atual de uso da cota. As cotas ajustáveis de grupos de usuários requests-per-second (RPS) do Amazon Cognito exibem seu uso atual. O console Service Quotas também pode direcionar você até as CloudWatch métricas para uma análise mais detalhada de uma métrica de cota selecionada. Para obter mais informações sobre como visualizar cotas no console do Service Quotas, consulte Visualizar Service Quotas.

Rastreie usuários ativos mensais (MAUs)

O número de usuários ativos mensais (MAUs) em seu grupo de usuários contribui com dados importantes para seu planejamento de aumentos nas cotas de taxa de solicitação. Você pode comparar suas taxas de solicitação de API com o número de usuários ativos em determinado período. Com esse conhecimento, você pode calcular como um aumento nos usuários ativos de suas aplicações afetará suas cotas no modelo de uso. Por exemplo, imagine que as aplicações combinadas no Oeste dos EUA (Oregon) resultaram em 2 milhões de usuários ativos em um mês e sua categoria `UserAuthentication` recebeu erros ocasionais de controle de utilização na cota padrão de 120 solicitações por segundo (RPS). No mês anterior, antes de sua campanha publicitária bem-sucedida, você tinha 1 milhão MAUs e seus aplicativos nunca ultrapassaram 80 RPS. Se você prevê um aumento semelhante como resultado de um novo comercial de TV, poderá comprar 40 RPS adicionais para acomodar o próximo milhão de usuários com uma cota ajustada de 160 RPS.

Para revisar seu MAUs

Acesse o console AWS Billing e analise uma fatura recente. Em cobranças por serviço, você pode filtrar no Cognito para ver um detalhamento do seu MAUs período de cobrança.

Solicitar um aumento de cota

O Amazon Cognito tem uma cota para o número máximo de operações por segundo que você pode realizar em seus grupos de usuários e grupos de identidades em cada um. Região da AWS Você pode solicitar um aumento nas cotas ajustáveis de taxas de solicitação de APIs nos grupos de usuários do Amazon Cognito. Verifique sua cota atual e compre um aumento no console Service

Quotas ou com as operações da API Service Quotas `ListAWSDefaultServiceQuotas` e `RequestServiceQuotaIncrease`.

- Para comprar um aumento de cota usando o console do Service Quotas, consulte Solicitar aumento de cota de API no Guia do usuário do Service Quotas.
- AWS visa a conclusão das solicitações de aumento de cota em 10 dias. No entanto, várias considerações podem fazer com que o tempo de processamento da solicitação exceda 10 dias. Algumas solicitações, por exemplo, podem exigir que o Amazon Cognito provisione capacidade adicional de hardware, e aumentos sazonais nos volumes de solicitações podem causar atrasos.
- Se a cota ainda não estiver disponível no Service Quotas, use o formulário de aumento de limites do serviço.

> **Important**
>
> Somente cotas ajustáveis podem ser aumentadas. Você deve comprar mais capacidade de cota. Para saber preços de aumento de cota, consulte Preço do Amazon Cognito.

Categorias de operação da API de grupos de usuários do Amazon Cognito e cotas de taxa de solicitação

Como o Amazon Cognito tem classes sobrepostas de operações de API com diferentes modelos de autorização, cada operação pertence a uma categoria. Cada categoria tem sua própria cota combinada para todas as operações da API de membros, em todos os grupos de usuários em uma Região da AWS em sua conta. Você só pode solicitar um aumento das cotas das categorias ajustáveis. Para obter mais informações, consulte Solicitar um aumento de cota. Os ajustes de cota se aplicam aos grupos de usuários em sua conta em uma única região. O Amazon Cognito restringe as operações em algumas categorias[3] a cinco solicitações por segundo (RPS), por grupo de usuários. A Cota padrão (RPS) também se aplica a todos os grupos de usuários em uma Conta da AWS.

> **Note**
>
> A cota para cada categoria é medida em Usuários ativos mensais (MAUs). Contas da AWS com menos de dois milhões MAUs podem operar dentro da cota padrão. Se você tiver menos de um milhão MAUs e o Amazon Cognito estiver limitando as solicitações,

considere otimizar seu aplicativo. Para obter mais informações, consulte Otimizar as taxas de solicitação para limites de cota.

Cotas de operação de categoria são aplicadas a todos os usuários em todos os grupos de usuários em uma Região da AWS. O Amazon Cognito também mantém uma cota para o número de solicitações que a aplicação pode gerar com relação a um usuário. Você deve limitar as solicitações de API por usuário conforme mostrado na tabela a seguir.

Cotas de taxa de solicitação por usuário de grupos de usuários do Amazon Cognito

Operation	Operações por usuário por segundo
Ler perfil de usuário Exemplos: `GetUser`, `GetDevice` , `InitiateAuth` , `RespondToAuthChallenge`	10
Gravar perfil de usuário Exemplos: `UpdateUserAttributes` , `SetUserSettings`	10

Você deve limitar as solicitações de API por categoria conforme mostrado na tabela a seguir.

Cotas de taxa de solicitação por categoria de grupos de usuários do Amazon Cognito

Categoria	Description	Cota padrão (RPS)	Ajustável
`UserAuthentication` • InitiateAuth • Atualização de token com `InitiateAuth` ou com o endpoint do token	Operações que autenticam (fazem login) um usuário. Essas operações estão sujeitas a Operações de API de grupos de usuários do Amazon Cognito	120	Sim

Categoria	Description	Cota padrão (RPS)	Ajustável
• RespondToAuthChallenge[1] • AdminInitiateAuth • AdminRespondToAuthChallenge[1] • GetTokensFromRefreshToken • Login gerenciado ou login de IU hospedada clássica e MFA para usuários locais em código de autorização ou concessões implícitas [2]	com processamento especial de taxa de solicitação.		
`UserCreation` • SignUp • ConfirmSignUp • AdminCreateUser • AdminConfirmSignUp	Operações que criam ou confirmam um usuário local do Amazon Cognito. Este é um usuário criado e verificado diretamente por seus grupos de usuários do Amazon Cognito.	50	Sim

Categoria	Description	Cota padrão (RPS)	Ajustável
`UserFederation` Operações que federam (autentic am) usuários com um provedor de identidad e de terceiros em seus grupos de usuários do Amazon Cognito.	Operações que enviam uma resposta de IdP a um endpoint de federação de grupos de usuários. As operações do OIDC ou do provedor social que resultam em um token de IdP e todas as solicitações de SAML contribuem para essa cota.	25	Sim
`UserAccou ntRecovery` • ChangePassword • ConfirmFo rgotPassword • ForgotPassword • AdminRese tUserPassword • AdminSetU serPassword • RespondTo AuthChallenge[1] • AdminResp ondToAuth Challenge[1] • Redefinição de senha de login gerenciado	Operações que recuperam a conta do usuário ou alteram ou atualizam a senha do usuário.	30	Não

Categoria	Description	Cota padrão (RPS)	Ajustável
`UserRead` • AdminGetUser • GetUser	Operações que recuperam um usuário dos grupos de usuários.	120	Sim
`UserUpdate` • AdminAddUserToGroup • AdminDeleteUserAttributes • AdminUpdateUserAttributes • AdminDeleteUser • AdminDisableUser • AdminEnableUser • AdminLinkProviderForUser • AdminDisableProviderForUser • VerifyUserAttribute • DeleteUser • DeleteUserAttributes • UpdateUserAttributes • AdminUserGlobalSignOut • GlobalSignOut • AdminRemoveUserFromGroup	As operações que os clientes usam para gerenciar usuários e atributos de usuários.	25	Não

Categoria	Description	Cota padrão (RPS)	Ajustável
`UserToken` • RevokeToken	Operações para gerenciamento de tokens	120	Sim
`UserResou rceRead` • AdminGetDevice • AdminList GroupsForUser • AdminListDevices • GetDevice • ListDevices • GetUserAttributeVe rificationCode • ResendCon firmationCode • AdminList UserAuthEvents	As operações que recuperam informaçõ es de recursos de usuários do Amazon Cognito, como um dispositivo lembrado ou uma associação de grupo.	50	Sim

Categoria	Description	Cota padrão (RPS)	Ajustável
`UserResourceUpdate` • AdminForgetDevice • AdminUpdateAuthEventFeedback • AdminSetUserMFAPreference • AdminSetUserSettings • AdminUpdateDeviceStatus • UpdateDeviceStatus • UpdateAuthEventFeedback • ConfirmDevice • SetUserMFAPreference • SetUserSettings • VerifySoftwareToken • AssociateSoftwareToken • ForgetDevice	As operações que atualizam informações de recursos de um usuário, como um dispositivo lembrado ou uma associação de grupo.	25	Não
`UserList` • ListUsers • ListUsersInGroup	Operações que retornam uma lista de usuários.	30	Não

Categoria	Description	Cota padrão (RPS)	Ajustável
`UserPoolRead` • DescribeUserPool • ListUserPools	Operações que leem seus grupos de usuários.	15	Não
`UserPoolUpdate` • CreateUserPool • UpdateUserPool • DeleteUserPool	Operações que criam, atualizam ou excluem seus clientes do grupo de usuários.	15	Não

Categoria	Description	Cota padrão (RPS)	Ajustável
`UserPoolResourceRead` • DescribeIdentityProvider • DescribeResourceServer • DescribeUserImportJob • DescribeUserPoolDomain • AdquiraCSVHeader • GetGroup • GetSigningCertificate • GetIdentityProviderByIdentifier • GetUserPoolMfaConfig • ListGroups • ListIdentityProviders • ListResourceServers • ListTagsForResource • ListUserImportJobs • DescribeRiskConfiguration • Adquira UICustomization	Operações que recuperam informações sobre recursos, como grupos ou servidores de recursos, de um grupo de usuários.[3]	20	Não

Categoria	Description	Cota padrão (RPS)	Ajustável
• DescribeManagedLoginBranding • DescribeManagedLoginBrandingByClient			

Categoria	Description	Cota padrão (RPS)	Ajustável
`UserPoolResourceUpdate` • AddCustomAttributes • CreateGroup • CreateIdentityProvider • CreateResourceServer • CreateUserImportJob • CreateUserPoolDomain • DeleteGroup • DeleteIdentityProvider • DeleteResourceServer • DeleteUserPoolDomain • SetUserPoolMfaConfig • StartUserImportJob • StopUserImportJob • UpdateGroup • UpdateIdentityProvider • UpdateResourceServer • UpdateUserPoolDomain	Operações que modificam recursos, como grupos ou servidores de recursos, em um grupo de usuários.[3]	15	Não

Categoria	Description	Cota padrão (RPS)	Ajustável
• SetRiskConfiguration • Conjunto UICustomization • TagResource • UntagResource • CreateManagedLoginBranding • UpdateManagedLoginBranding • DeleteManagedLoginBranding			
`UserPoolClientRead` • DescribeUserPoolClient • ListUserPoolClients	Operações que recuperam informações sobre os clientes do grupo de usuários. [3]	15	Não
`UserPoolClientUpdate` • CreateUserPoolClient • DeleteUserPoolClient • UpdateUserPoolClient	Operações que criam, atualizam e excluem os clientes do grupo de usuários. [3]	15	Não

Categoria	Description	Cota padrão (RPS)	Ajustável
`ClientAuthentication` `client_credentials` concede solicitações de tipo para o endpoint do token.	Operações que geram credenciais para serem usadas na autorização machine-to-machine de solicitações	150	Não

[1] Uma resposta `RespondToAuthChallenge` ou `AdminRespondToAuthChallenge` com `ChallengeName` de `NEW_PASSWORD_REQUIRED` é considerada em relação à categoria `UserAccountRecovery`. Todas as outras respostas do desafio são consideradas para a categoria `UserAuthentication`.

[2] Cada login gerenciado ou operação de IU hospedada clássica durante o login contribui com uma solicitação para a cota. Por exemplo, um usuário que faz login e fornece um código de MFA contribui com duas solicitações. O resgate de tokens em concessões de códigos de autorização está sujeito a uma alocação de cota adicional na mesma taxa da sua cota na categoria `UserAuthentication`.

[3] Qualquer operação individual nesta categoria tem uma restrição que impede que a operação seja chamada a uma taxa superior a 5 RPS para um único grupo de usuários.

Limites de taxa de solicitações em massa para domínios de grupos de usuários

As cotas a seguir se aplicam ao volume geral de solicitações para um domínio do grupo de usuários.

Operation	Description	Cota padrão (RPS)	Ajustável
Solicitações do IP de origem	Volume de solicitações de um endereço IP para um domínio	300	Não
Solicitações para cliente de aplicação	Volume de solicitações para um ID de	300	Não

Operation	Description	Cota padrão (RPS)	Ajustável
	cliente de aplicação em um domínio		
Solicitações para domínio	Volume geral de solicitações para os serviços de um domínio do grupo de usuários	500	Não
Solicitações para documentos de chave da web JSON	Volume de solicitações `jwks.json` em um Conta da AWS em um Região da AWS	50.000	Não

Cotas de taxa de solicitação de operação da API de grupos de identidades (identidades federadas) do Amazon Cognito

Operation	Description	Cota padrão (RPS)[1]	Ajustável	Elegibilidade para aumento de cota
`GetId`	Recupere um ID de identidade de um grupo de identidades.	25	Sim	Entre em contato com a equipe da conta.
`GetOpenIdToken`	Recupere um token OpenID de um grupo de identidades no fluxo de trabalho clássico.	200	Sim	Entre em contato com a equipe da conta.

Operation	Description	Cota padrão (RPS)[1]	Ajustável	Elegibilidade para aumento de cota
`GetCredentialsForIdentity`	Recupere AWS credenciais de um grupo de identidades no fluxo de trabalho aprimorado.	200	Sim	Entre em contato com a equipe da conta.
`GetOpenIdTokenForDeveloperIdentity`	Recupere um token OpenID de um grupo de identidades no fluxo de trabalho de desenvolvedor.	50	Sim	Entre em contato com a equipe da conta.
`ListIdentities`	Recupere uma lista de identidades IDs em um pool de identidades.	5	Sim	Entre em contato com a equipe da conta.
`DeleteIdentities`	Exclua uma ou mais identidades registradas de um banco de identidades.	10	Sim	Entre em contato com a equipe da conta.
`TagResource`	Aplique uma tag a um banco de identidades.	5	Sim	Entre em contato com a equipe da conta.
`UntagResource`	Remova uma tag de um banco de identidades.	5	Sim	Entre em contato com a equipe da conta.

Operation	Description	Cota padrão (RPS)[1]	Ajustável	Elegibilidade para aumento de cota
`ListTagsForResource`	Exiba uma lista das tags aplicadas a um banco de identidades.	10	Sim	Entre em contato com a equipe da conta.

[1] A cota padrão é a cota mínima de taxa de solicitação para os grupos de identidades Região da AWS em qualquer um dos seus. Conta da AWS Sua cota de RPS pode ser maior em algumas regiões.

Cotas de número e tamanho do recurso

As cotas de recursos são o número ou tamanho máximo de recursos, campos de entrada, duração do tempo e outros recursos diversos no Amazon Cognito.

É possível solicitar um ajuste em algumas cotas de recursos no console do Service Quotas ou em um formulário de aumento de limite de serviço. Para solicitar um aumento de cota usando o console do Service Quotas, consulte Solicitar um aumento de cota no Guia do usuário do Service Quotas. Se a cota ainda não estiver disponível no Service Quotas, use o formulário de aumento de limites do serviço.

> **Note**
>
> As cotas de recursos no Conta da AWS nível, como grupos de usuários por região, se aplicam aos recursos do Amazon Cognito em cada um. Região da AWS Por exemplo, você pode ter 1.000 grupos de usuários na região Leste dos EUA (Norte da Virgínia) e outros 1.000 na região Europa (Estocolmo).

As tabelas a seguir indicam as cotas de recursos padrão e se elas são ajustáveis.

Cotas de recursos de grupos de usuários do Amazon Cognito

As cotas a seguir descrevem o número ou o tamanho máximo de itens que você pode criar em grupos de usuários.

Recurso	Quota	Ajustável	Cota máxima
Clientes da aplicação por grupo de usuários	1.000	Sim	10.000
Grupos de usuários por região	1.000	Sim	10.000
Provedores de identidade por grupo de usuários	300	Sim	1.000
Servidores de recursos por grupo de usuários	25	Sim	300
Usuários por grupo de usuários	40.000.000	Sim	Entre em contato com a equipe da conta.
Total de alterações combinadas no gatilho do Lambda de pré-geração de tokens[1]	5.000	Sim	Entre em contato com a equipe da conta.
Estilos de identidade visual de login gerenciado por grupo de usuários	20	Não	N/D
Documentos de termos de login	40	Não	N/D

Recurso	Quota	Ajustável	Cota máxima
gerenciados por grupo de usuários			
Atributos personalizados por grupo de usuários	50	Não	N/D
Caracteres por atributo	2.048 bytes	Não	N/D
Caracteres no nome de um atributo personalizado	20	Não	N/D
Caracteres de senha mínimos necessários na política de senha	6 a 99	Não	N/D
Mensagens de e-mail enviadas diariamente por Conta da AWS[2]	50	Não	N/D
Mensagens de MFA por e-mail enviadas para um endereço de e-mail a cada hora por endereço IP do solicitante	5-20	Não	N/D
Caracteres no assunto do e-mail	140	Não	N/D
Caracteres na mensagem de e-mail	20.000	Não	N/D

Recurso	Quota	Ajustável	Cota máxima
Caracteres na mensagem de verificação por SMS	140	Não	N/D
Caracteres na senha	256	Não	N/D
Caracteres no nome do provedor de identidade	32	Não	N/D
Caracteres em uma resposta SAML	100.000	Não	N/D
Identificadores por provedor de identidade	50	Não	N/D
Identidades vinculadas a um usuário	5	Não	N/D
Chave de WebAuthn senha/autenticadores por usuário	20	Não	N/D
Retorno de chamada URLs por cliente de aplicativo	100	Não	N/D
Sair URLs por cliente de aplicativo	100	Não	N/D
Escopos por servidor de recursos	100	Não	N/D
Escopos por cliente da aplicação	50	Não	N/D

Recurso	Quota	Ajustável	Cota máxima
Domínios personalizados por região	4	Não	N/D
Grupos aos quais cada usuário pode pertencer	100	Não	N/D
Grupos por grupo de usuários	10.000	Não	N/D

[1] Essa cota pode ser encontrada em tokens de uma Acionador do Lambda antes da geração do token. O número de reivindicações existentes e adicionadas, além dos escopos nos tokens de acesso e identidade em uma transação, deve somar um número menor ou igual a essa cota. Declarações e escopos suprimidos não contribuem com essa cota.

[2] Essa cota se aplicará somente se você estiver usando o atributo de e-mail padrão para um grupo de usuários do Amazon Cognito. Para um volume de entrega de e-mails maior, configure o grupo de usuários para usar a configuração de e-mail do Amazon SES. Essa restrição é redefinida diariamente às 09:00 UTC. Para obter mais informações, consulte Configurações de e-mail para grupos de usuários do Amazon Cognito.

Parâmetros de validade de sessão de grupos de usuários do Amazon Cognito

As cotas a seguir descrevem as configurações disponíveis para a duração dos artefatos de autenticação e das sessões de usuário nos grupos de usuários.

Token	Quota
Token de ID	5 minutos – 1 dia
Token de atualização	1 hora – 3.650 dias
Token de acesso	5 minutos – 1 dia

Token	Quota
Cookie de sessão da interface de usuário hospedada	1 hora
Token de sessão de autenticação	Três a quinze minutos

Cotas de recursos de segurança de código de grupos de usuários do Amazon Cognito

As cotas a seguir descrevem os períodos disponíveis relacionados aos códigos para login, cadastro e redefinição de senha.

Recurso	Quota
Período de validade do código de confirmação de cadastro	24 horas
Período de validade do código de verificação do atributo do usuário	24 horas
Período de validade do código de autenticação multifator (MFA)	De 3 a 15 minutos
Período de validade do código de esquecimento de senha	1 hora
Número máximo de solicitações de `ConfirmForgotPassword` e de `ForgotPassword` por usuário, por hora [1]	5 a 20
Número máximo de solicitações de `ResendConfirmationCode` por usuário, por hora	5
Número máximo de solicitações de `ConfirmSignUp` por usuário, por hora	15

Recurso	Quota
Número máximo de solicitações de `ChangePassword` por usuário, por hora	5
Número máximo de solicitações de `GetUserAttributeVerificationCode` por usuário, por hora	5
Número máximo de solicitações de `VerifyUserAttribute` por usuário, por hora	15

[1] O Amazon Cognito avalia os fatores de risco na solicitação de atualização de senhas e atribui uma cota vinculada ao nível de risco avaliado. Para obter mais informações, consulte Comportamento para esquecimento da senha.

Cotas de recursos de trabalho de importação de usuários do grupo de usuários do Amazon Cognito

As cotas a seguir descrevem os recursos e os limites disponíveis para trabalhos de importação do usuário.

Recurso	Quota	Ajustável	Cota máxima
Trabalhos de importação de usuário por grupo de usuários	1.000	Sim	Entre em contato com a equipe da conta.
Máximo de caracteres por linha do CSV de importação de usuários	16.000	Não	N/D
Tamanho máximo do arquivo CSV	100 MB	Não	N/D

Recurso	Quota	Ajustável	Cota máxima
Número máximo de usuários por arquivo CSV	500.000	Não	N/D

Cotas de recursos de bancos de identidades do Amazon Cognito (identidades federadas)

As cotas a seguir descrevem o número ou o tamanho máximo de itens que você pode criar em bancos de identidades.

Recurso	Quota	Ajustável	Cota máxima
Bancos de identidades por conta	1.000	Sim	N/D
Provedores de grupos de usuários do Amazon Cognito por grupo de identidades	50	Sim	1000
Caracteres de um nome do grupo de identidades	128 bytes	Não	N/D
Caracteres de um nome do provedor de login	2.048 bytes	Não	N/D
Identidades por grupo de identidades	Ilimitado	Não	N/D
Provedores de identidade para os quais podem ser especificados	10	Não	N/D

Recurso	Quota	Ajustável	Cota máxima
mapeamentos de função			
Resultados de uma única chamada de lista ou pesquisa	60	Não	N/D
Regras de Role-based access control (RBAC – Controle de acesso com base em função)	25	Não	N/D

Cotas de recursos do Amazon Cognito Sync

As cotas a seguir descrevem o número ou o tamanho máximo dos itens que você pode criar no Amazon Cognito Sync.

Recurso	Quota	Ajustável	Cota máxima
Conjuntos de dados por identidade	20	Sim	Entre em contato com a equipe da conta.
Registros por conjunto de dados	1,024	Sim	Entre em contato com a equipe da conta.
Tamanho de um único conjunto de dados	1 MB	Sim	Entre em contato com a equipe da conta.
Caracteres no nome do conjunto de dados	128 bytes	Não	N/D
Tempo de espera para uma publicação em massa após uma	24 horas	Não	N/D

Recurso	Quota	Ajustável	Cota máxima
solicitação bem-sucedida			

Histórico do documento do Amazon Cognito

A tabela a seguir descreve adições importantes à documentação do Amazon Cognito. Também fazemos atualizações secundárias frequentes na documentação em resposta ao feedback enviado. Para enviar feedback, localize o link Feedback na parte inferior de qualquer página na documentação do Amazon Cognito.

Alteração	Descrição	Data
Documentação atualizada do comportamento de repetição do acionador Lambda.	Documentação atualizada para esclarecer que o Amazon Cognito pode repetir as invocações da função Lambda que atinjam o tempo limite, sem especificar um número fixo de novas tentativas.	23 de março de 2026
Rotação secreta de clientes para clientes de aplicativos.	Agora você pode associar até dois segredos por cliente de aplicativo para rotação de segredos. Você também pode fornecer seu próprio valor secreto personalizado ao adicionar um segredo.	30 de janeiro de 2026
O Inbound Federation Lambda Trigger está disponível nos grupos de usuários do Amazon Cognito.	Agora você pode usar o gatilho lambda de federação de entrada para transform ar atributos de usuários federados durante o processo de autenticação com provedores de identidade externos.	28 de janeiro de 2026
AWS PrivateLink está disponível nos grupos de	Agora você pode enviar solicitações para as operações da API do pool de identidade	10 de dezembro de 2025

identidade do Amazon Cognito.	do Amazon Cognito por meio de VPC endpoints.	
AWS PrivateLink está disponível nos grupos de usuários do Amazon Cognito.	Agora você pode enviar solicitações para operações de API de grupos de usuários do Amazon Cognito por meio de VPC endpoints.	4 de novembro de 2025
Os grupos de usuários do Amazon Cognito agora são compatíveis com a vinculação de recursos.	Os servidores de recursos agora podem realizar a verificação de público-alvo dos tokens de acesso com a vinculação de recursos RFC 8707.	24 de outubro de 2025
Foi adicionado um capítulo sobre solução de problemas.	Novo capítulo com soluções para problemas comuns com grupos de usuários.	24 de outubro de 2025
Foi adicionada uma nova cota para solicitações de documentos de chaves web JSON do grupo de usuários.	Nova entrada de cota para busca nos servidores de autorização `.well-known/jwks.json` do grupo de usuários OAuth 2.0.	13 de outubro de 2025
Documentos de termos e condições no login gerenciado.	Agora você pode adicionar links para os termos e condições e política de privacidade nas páginas de cadastro de login gerenciado.	2 de outubro de 2025
Nova aplicação de exemplo de introdução para bancos de identidades.	Agora você pode configurar uma aplicação de exemplo que demonstra os recursos dos bancos de identidades.	30 de setembro de 2025

Atualizações dos requisitos para políticas de confiança de perfis do IAM para bancos de identidades.	Agora você pode alterar as políticas de confiança de perfil para a entidade principal de serviço de banco de identidades do Amazon Cognito somente quando uma chave de condição define o escopo do público-alvo da federação OIDC (identida de da web) para um ou mais bancos de identidades.	1.º de agosto de 2025
O Amazon Cognito agora está disponível no México (Central) . Região da AWS	Agora é possível criar recursos do Amazon Cognito na região do México (Centro).	24 de julho de 2025
O Amazon Cognito agora está disponível na Ásia-Pacífico (Tailândia). Região da AWS	Agora é possível criar recursos do Amazon Cognito na região da Ásia-Pacífico (Tailândia).	24 de julho de 2025
AWS WAF web ACLs no login gerenciado.	Agora você pode aplicar regras de ACL AWS WAF da web a clientes de aplicativos de grupos de usuários que tenham a versão gerenciada da marca de login.	24 de junho de 2025
Foram atualizados exemplos de acionadores do Lambda.	Foi atualizada a função de exemplo para acionadores do Lambda de remetente personalizado de e-mail e SMS para ser compatível com o Node.js 22.x. O exemplo agora também está mais acessível para testes.	19 de maio de 2025

Novo parâmetro `prompt`.	Agora você tem maior controle sobre a reautenticação das sessões existentes de login gerenciado com o parâmetro `prompt`. Você também pode transmitir valores desse parâmetro para provedores de terceiros.	15 de maio de 2025
Metadados de clientes para solicitações de M2M.	Agora você pode passar metadados do cliente nas credenciais do cliente ou solicitações machine-to-machine (M2M). O Amazon Cognito transmite metadados de clientes M2M para o acionador do Lambda de pré-geração de tokens.	29 de abril de 2025
Alternância de tokens de atualização.	Agora você pode obter novos tokens de atualização e invalidar os originais nas solicitações de token de atualização.	22 de abril de 2025
O Amazon Cognito agora está disponível na Ásia-Pacífico (Malásia). Região da AWS	Agora você pode criar recursos do Amazon Cognito na região da Ásia-Pacífico (Malásia).	07 de março de 2025

Personalização do token de acesso para identidades de máquinas.	O gatilho Lambda de pré-geração de token agora tem um evento de versão três que modifica as reivindicações e os escopos do token de acesso nas concessões de credenciais do cliente para autorização (M2M). machine-to-machine	3 de março de 2025
Foram atualizadas informações sobre a política gerenciada pela AWS`AmazonCognitoPowerUser` .	Foi adicionada uma AWS End User Messaging SMS operação na política AWS gerenciada para usuários avançados de grupos de usuários do Amazon Cognito.	27 de fevereiro de 2025
Foi atualizada a visão geral da integração do OpenID Connect (OIDC).	Foi adicionado um diagrama que ilustra como o Amazon Cognito se autentica com provedores de identidades OIDC.	25 de fevereiro de 2025
Foram adicionadas informações sobre a lógica de MFA.	Foi adicionado um diagrama que ilustra como o Amazon Cognito aplica as configurações de autenticação multifator (MFA) do grupo de usuários aos usuários em runtime.	25 de fevereiro de 2025
Foram adicionadas praticas recomendadas de segurança para grupos de usuários do Amazon Cognito.	Foi adicionada uma página sobre como proteger segredos e seguir as práticas recomendadas de segurança na configuração do grupo de usuários.	25 de fevereiro de 2025

Atualizações nos recursos de introdução para grupos de usuários.	A experiência de introdução com grupos de usuários do Amazon Cognito tem um novo design de console e opções de aplicações.	21 de novembro de 2024
Novo modelo de preços com planos de recursos.	Foi atualizado o modelo de cobrança para grupos de usuários. Os recursos avançados de segurança agora incluem proteção contra ameaças. Os componentes da licença de recursos de segurança avançados agora estão disponíveis nos planos de recursos Essentials e Plus.	21 de novembro de 2024
Novo recurso de login gerenciado.	O login gerenciado foi lançado, uma atualização da IU hospedada.	21 de novembro de 2024
Um novo método de autenticação e novos fluxos de autenticação.	Agora você pode fazer login nos grupos de usuários do Amazon Cognito com chaves de acesso e senhas de uso único.	21 de novembro de 2024
Informações atualizadas sobre `AmazonCognitoUnAuthedIdentitiesSessionPolicy`.	Transferiu AWS Key Management Service as operações na política AWS gerenciada para reduzir o escopo de identidades não autenticadas da política em linha para a política gerenciada. AWS	1.º de novembro de 2024

Parâmetro `login_hint` adicionado.	Agora você pode adicionar uma dica de nome de usuário às solicitações de autorização para a interface do usuário hospedada, o OIDC IdPs e o Google. IdPs	3 de outubro de 2024
Novos recursos avançados de segurança para MFA do e-mail.	Agora você pode enviar códigos de autenticação multifator (MFA) por mensagem de e-mail com recursos avançados de segurança.	12 de setembro de 2024
Novo conteúdo e alterações na página.	Títulos modificados, conteúdo desnecessário removido, introduções baseadas em cenários adicionadas, grupos de usuários movidos e referência de endpoints de interface do usuário hospedada aos grupos de usuários.	9 de setembro de 2024
Informações atualizadas sobre `AmazonCognitoUnAuthedIdentitiesSessionPolicy`.	A política AWS gerenciada para redução do escopo de identidades não autenticadas em grupos de identidades agora permite o Amazon Location Service.	9 de agosto de 2024

Nova prevenção de ameaças para autenticação personalizada com acionadores do Lambda e detecção aprimorada de ameaças.	Agora você pode analisar o login de autenticação personalizada com proteção contra ameaças e aplicar respostas de autenticação adaptáveis. A proteção contra ameaças também analisa o tráfego de login em busca de uma distância geográfica impossível entre tentativas.	8 de agosto de 2024
Novos recursos avançados de segurança para prevenção de reutilização de senhas e exportação de logs de atividades do usuário.	Agora você pode exportar logs de atividades do usuário e definir uma política de histórico de senhas com recursos avançados de segurança nos grupos de usuários do Amazon Cognito.	6 de agosto de 2024
O Amazon Cognito agora está disponível no Oeste do Canadá (Calgary) e na Ásia-Pacífico (Hong Kong). Regiões da AWS	Agora você pode criar recursos do Amazon Cognito nas regiões do Oeste do Canadá (Calgary) e Ásia-Pacífico (Hong Kong).	9 de julho de 2024
Descrição aprimorada do comportamento da aplicação para segurança avançada	Informações atualizadas sobre dados de contexto do dispositivo para autenticação adaptável de segurança avançada.	10 de junho de 2024
Suporte adicionado para objetos complexos no acionador do Lambda pré-token	Agora você pode adicionar matrizes e objetos JSON às reivindicações de ID e token de acesso.	30 de maio de 2024

Informações atualizadas sobre o Verified Permissions e o Amazon Cognito.	O Amazon Verified Permissions agora tem uma integração mais direta com o Amazon Cognito.	15 de maio de 2024
Identidades verificadas pelo Amazon SES em várias regiões.	Em algumas regiões Regiões da AWS sem o Amazon SES, os grupos de usuários do Amazon Cognito fazem balanceamento de carga entre duas regiões remotas.	10 de maio de 2024
Informações adicionadas sobre autorização M2M e o gerenciamento de custos.	Saiba como usar concessões de credenciais de clientes para casos de uso machine-to-machine (M2M) com grupos de usuários do Amazon Cognito.	9 de maio de 2024
O Amazon Cognito agora está disponível na Europa (Espanha) e Ásia-Pacífico (Hyderabad). Regiões da AWS	Agora você pode criar recursos do Amazon Cognito nas regiões Europa (Espanha) e Ásia-Pacífico (Hyderabad).	15 de abril de 2024
O Amazon Cognito agora está disponível na Ásia-Pacífico (Melbourne). Região da AWS	Agora você pode criar recursos do Amazon Cognito na região Ásia-Pacífico (Melbourne).	4 de abril de 2024
Foi adicionado um exemplo de aplicativo para Android no Flutter para grupos de usuários do Amazon Cognito.	Você pode criar um aplicativo móvel inicial para o Amazon Cognito a partir de um exemplo de aplicativo Flutter ativado. GitHub	4 de abril de 2024

Novo conteúdo de introdução	Conteúdo expandido para começar, cenários comuns, práticas recomendadas para vários locatários e acesso a recursos após o login.	1º de abril de 2024
O Amazon Cognito agora está disponível na Europa (Zurique) . Região da AWS	Agora é possível criar grupos de usuários do Amazon Cognito na região Europa (Zurique).	14 de março de 2024
O Amazon Cognito agora está disponível no Oriente Médio (EAU). Região da AWS	Agora é possível criar grupos de usuários do Amazon Cognito na região Oriente Médio (EAU).	8 de março de 2024
Novos recursos do SAML e conteúdo aprimorado.	Agora você pode assinar solicitações SAML, criptogra far respostas SAML e configurar o SSO SAML iniciado pelo IdP.	1.º de fevereiro de 2024
Aumentos de cota disponíveis.	Agora você pode comprar capacidade adicional para as cotas de taxas de solicitação do Amazon Cognito.	25 de janeiro de 2024
Os bancos de identidades do Amazon Cognito são compatíveis com as taxas de solicitação no Service Quotas.	Agora você pode monitorar cotas requests-per-secon d (RPS) para grupos de identidade do Amazon Cognito e solicitar aumento no console Service Quotas.	19 de dezembro de 2023
Foi adicionado um novo atributo para personalização do conteúdo dos tokens de acesso.	Agora é possível adicionar, modificar e remover declaraçõ es e escopos nos tokens de acesso do grupo de usuários.	12 de dezembro de 2023

Conteúdo aprimorado sobre clientes e OAuth escopos de aplicativos.	Edições e correções de Configurações específicas da aplicação com clientes de aplicação e Escopos, M2M e servidores de recursos para aumentar a clareza. Instruções do console herdado removidas .	14 de novembro de 2023
Foi aprimorado o conteúdo sobre dispositivos e autenticação de dispositivos.	Novo conteúdo sobre o uso de chaves de dispositivo e autenticação SRP do dispositivo.	18 de outubro de 2023
Console de gerenciamento da AWS Orientação atualizada.	Foi removida a referência ao console de grupos de usuários e os tópicos dentro dos assuntos relacionados foram redistribuídos. Foram adicionadas orientações à organização baseada em guias no console do Amazon Cognito.	30 de agosto de 2023
Foi removida a ênfase sobre o acesso direto ao endpoint LOGIN.	Foi adicionada uma visão geral visual do grupo de usuários Endpoint de login e foi enfatizado o início da autenticação com Autorizar endpoint.	30 de agosto de 2023
O Amazon Cognito agora está disponível na Ásia-Pacífico (Osaka) e em Israel (Tel Aviv). Regiões da AWS	Agora você pode criar recursos do Amazon Cognito nas regiões Asia Pacific (Osaka) e Israel (Tel Aviv).	30 de agosto de 2023

Foram introduzidas informações sobre autorização para o Amazon Cognito com o Amazon Verified Permissions.	Em sua aplicação, você pode invocar a API do Verified Permissions para gerar decisões de acesso de uma autoridade central.	1º de agosto de 2023
Foi adicionado um novo recurso para registrar a atividade detalhada do usuário no Amazon CloudWatch Logs.	Agora você pode registrar erros de entrega de e-mails e mensagens SMS em grupos de CloudWatch registros.	1º de agosto de 2023
Informações atualizadas sobre a política AWS gerenciada para usuários convidados do pool de identidades.	O escopo de permissões para usuários convidados do pool de identidades agora inclui uma política de sessão embutida e uma AWS política de sessão gerenciada.	16 de maio de 2023
Melhoria de conteúdo e novas instruções de console para bancos de identidades do Amazon Cognito.	Adição de novas orientações do console para refletir a nova experiência do console, detalhes aprimorados de integração de código para bancos de identidades.	16 de maio de 2023
Adições e melhorias na página inicial do serviço e na página inicial dos grupos de usuários.	Páginas de visão geral atualizadas para o Amazon Cognito e grupos de usuários.	16 de maio de 2023
Melhorias gerais na documentação do token do grupo de usuários.	Tokens de exemplo foram atualizados, novas informações sobre verificação de tokens foram adicionadas.	16 de fevereiro de 2023

Agora é possível registrar eventos de dados de bancos de identidades do Amazon Cognito no AWS CloudTrail.	CloudTrail suporta a seleção de grupos de identidade do Amazon Cognito, operações de API de alto volume em trilhas que registram eventos de dados.	15 de fevereiro de 2023
Exemplos e descrições de acionadores do Lambda atualizados.	Os exemplos de gatilhos do Lambda foram atualizados para a JavaScript versão 3. Agora você pode correlacionar diretamente os gatilhos do Lambda às ações da API.	31 de janeiro de 2023
Os grupos de identidade do Amazon Cognito aplicam uma política AWS gerenciada a sessões não autenticadas.	Os usuários do pool de identidades que se autentica m usando o fluxo aprimorado agora têm uma política AWS gerenciada adicional aplicada à sessão.	31 de janeiro de 2023
Foram adicionados exemplos de código.	Este guia agora inclui código de exemplo para sua aplicação Amazon Cognito em várias linguagens de programação.	23 de janeiro de 2023
Foram adicionadas informaçõ es sobre modelos de API e autenticação com grupos de usuários do Amazon Cognito.	Os grupos de usuários do Amazon Cognito têm várias interfaces e formatos de API para solicitar autorização.	15 de dezembro de 2022
O Amazon Cognito agora está disponível na Europa (Milão). Região da AWS	Agora é possível criar grupos de usuários do Amazon Cognito na região Europa (Milão).	6 de dezembro de 2022

Foram adicionadas informações sobre a proteção contra exclusão do grupo de usuários.	Quando você cria um novo grupo de usuários com o Console de gerenciamento da AWS, ele agora está protegido contra exclusão por padrão.	20 de outubro de 2022
Adição de um guia do usuário para a interface de usuário hospedada e informações sobre a MFA com TOTP na interface hospedada.	Os usuários agora podem registrar um dispositivo MFA com TOTP na UI hospedada do Amazon Cognito. Agora você pode visualizar a interface do usuário hospedada padrão.	8 de setembro de 2022
Foram adicionadas informações sobre o AWS WAF Amazon Cognito.	Agora você pode associar uma AWS WAF Web ACL a um grupo de usuários do Amazon Cognito.	3 de agosto de 2022
Foram adicionados mais exemplos de AWS CloudTrail eventos.	Agora, o Amazon Cognito registra na trilha as solicitações da federação e da interface do usuário hospedada.	15 de junho de 2022
Foram adicionadas informações sobre a verificação de atributos em duas etapas.	Agora você pode escolher se o usuário deve verificar um novo endereço de e-mail ou número de telefone antes de fazer login com ele.	9 de junho de 2022

Documentação de federação atualizada. Novo recurso de propagação de endereço IP.	Instruções atualizadas para configurar o grupo de usuários nas redes sociais. IdPs Adição de informações sobre perfis de usuários federados e mapeamento de atributos . Foram adicionadas novas informações sobre impressões digitais do dispositivo para segurança avançada.	31 de maio de 2022
Fazer login de usuários federados sem interação com a interface do usuário hospedada	Foi adicionada uma nova página sobre como marcar aplicações para que o Amazon Cognito direcione silenciosamente os usuários para login federado.	29 de maio de 2022
Sistema de mensagens de e-mail e SMS na região para grupos de usuários do Amazon Cognito	Agora você pode usar o Amazon Simple Notification Service para mensagens SMS e o Amazon Simple Email Service para mensagens de e-mail no Região da AWS mesmo grupo de usuários.	14 de março de 2022
Atualizações na página de cotas	Foram adicionadas e esclarecidas cotas de recurso e taxas de solicitação.	10 de janeiro de 2022
Nova experiência do console de grupos de usuários do Amazon Cognito	Instruções atualizadas para criar e gerenciar grupos de usuários no console atualizado do Amazon Cognito.	18 de novembro de 2021

RevokeToken API e endpoint de revogação	Você pode usar a RevokeToken operação para revogar um token de atualização para um usuário.	10 de junho de 2021
Práticas recomendadas de vários locatários	Práticas recomendadas para aplicações de vários locatários foram adicionadas.	4 de março de 2021
Atributos para controle de acesso	Os grupos de identidade do Amazon Cognito fornecem atributos para controle de acesso (AFAC) como uma forma de os clientes concederem aos usuários acesso aos recursos. AWS A autorização pode ser feita com base nos atributos dos usuários do provedor de identidade que eles usaram para federar com o Amazon Cognito.	15 de janeiro de 2021
Acionador do Lambda remetente personalizado de SMS e o Acionador do Lambda remetente personalizado de e-mail	O acionador do Lambda remetente personalizado de SMS e o Acionador do Lambda remetente personalizado de e-mail permitem que um provedor de terceiros envie notificações por e-mail e SMS para seus usuários com o código de função do Lambda.	30 de novembro de 2020

Atualizações de token do Amazon Cognito	Informações de validade atualizadas foram adicionadas aos tokens de acesso, ID e atualização.	29 de outubro de 2020
Service Quotas do Amazon Cognito	O Service Quotas está disponível para cotas de categoria do Amazon Cognito. Você pode usar o console Service Quotas para visualizar o uso da cota, solicitar um aumento da cota e criar CloudWatch alarmes para monitorar o uso da cota. Como parte dessa alteração, a seção CloudWatch Métricas disponíveis para grupos de usuários do Amazon Cognito foi atualizada para refletir as novas informações. O novo nome da seção é: Rastreamento de cotas e uso em CloudWatch e Service Quotas	29 de outubro de 2020
Categorização de cotas do Amazon Cognito	As categorias de cotas estão disponíveis para ajudar você a monitorar o uso da cota e solicitar um aumento. As cotas são agrupadas em categorias com base em casos de uso comuns.	17 de agosto de 2020
O Amazon Cognito é compatível com o governo dos EUA AWS CLoud	O Amazon Cognito agora é suportado na região AWS GovCloud (EUA).	13 de maio de 2020

Atualizações de documentos do Amazon Cognito Pinpoint	Foi adicionada nova função vinculada ao serviço. Foram atualizadas as instruções em “Usar análise do Amazon Pinpoint com grupos de usuários do Amazon Cognito”.	13 de maio de 2020
Novo capítulo de segurança dedicado ao Amazon Cognito	O capítulo Segurança pode ajudar sua organização a obter informações detalhadas sobre a segurança integrada e configurável dos AWS serviços. Nossos novos capítulos oferecem informaçõ es sobre a segurança da nuvem e na nuvem.	30 de abril de 2020
O Amazon Cognito Identity Pools agora oferece suporte para o recurso Fazer login com a Apple	O recurso Fazer login com a Apple está disponível em todas as regiões em que o Amazon Cognito opera, exceto na região cn-north-1.	7 de abril de 2020
Novo versionamento da API do Facebook	Seleção de versão adicionada à API do Facebook	3 de abril de 2020
Atualização da indistinção de maiúsculas e minúsculas em nome de usuário	Recomendação adicionada sobre como habilitar a indistinç ão de maiúsculas e minúscula s em nome de usuário antes de criar um grupo de usuários.	11 de fevereiro de 2020

Novas informações sobre AWS Amplify	Foram adicionadas informaçõ es sobre a integração do Amazon Cognito com seu aplicativo web ou móvel AWS Amplify SDKs usando bibliotec as. Foram removidas as informações anteriores sobre o uso do Amazon Cognito SDKs. AWS Amplify	22 de novembro de 2019
Novo atributo para acionador es do grupo de usuários	O Amazon Cognito agora inclui um `clientMetadata` parâmetro nas informações do evento que ele passa para AWS Lambda as funções da maioria dos acionadores do grupo de usuários. É possível usar esse parâmetro para aprimorar o fluxo de trabalho de autenticação personalizado com dados adicionais.	4 de outubro de 2019
Limite atualizado	O limite de limitação para a ação da ListUsers API foi atualizado.	25 de junho de 2019
Novo limite	Agora os limites flexíveis dos grupos de usuários incluem um limite para o número de usuários.	17 de junho de 2019

Configurações de e-mail do Amazon SES para grupos de usuários do Amazon Cognito	Você pode configurar um grupo de usuários para que o Amazon Cognito envie e-mails aos seus usuários usando a configuração do Amazon SES. Essa configuração permite que o Amazon Cognito envie um e-mail com um volume de entrega mais alto do que é possível.	8 de abril de 2019
Suporte a marcação	Adicionadas informações sobre marcação de recursos do Amazon Cognito.	26 de março de 2019
Alterar o certificado para um domínio personalizado	Se você usar um domínio personalizado para hospedar a interface do usuário hospedada do Amazon Cognito, poderá alterar o certificado SSL para esse domínio, conforme necessário.	19 de dezembro de 2018
Novo limite	Um novo limite é adicionado para o número máximo de grupos a que cada usuário pode pertencer.	14 de dezembro de 2018
Limites atualizados	Os limites flexíveis para grupos de usuários estão atualizados.	11 de dezembro de 2018

Atualização da documentação para verificar endereços de e-mail e números de telefone	Adição de informações sobre como configurar o grupo de usuários para exigir verificação de e-mail ou telefone quando um usuário se cadastra em seu aplicativo.	20 de novembro de 2018
Atualização da documentação para testar e-mails	Adicionada a orientação para iniciar os e-mails no Amazon Cognito enquanto você testa sua aplicação.	13 de novembro de 2018
Segurança avançada do Amazon Cognito	Novos recursos de segurança foram adicionados para permitir que os desenvolv edores protejam seus aplicativ os e usuários de bots mal-intencionados, protejam contas de usuário em relação a credenciais comprometidas e ajustem automaticamente os desafios necessários para fazer login com base no risco calculado da tentativa de login.	14 de junho de 2018
Domínios personalizados para interface do usuário hospedada do Amazon Cognito	Permite que os desenvolv edores usem seu próprio domínio totalmente personali zado para a interface do usuário hospedada em grupos de usuários do Amazon Cognito.	4 de junho de 2018

Provedor de identidades OIDC de grupos de usuários do Amazon Cognito	Adição de login do grupo de usuários por meio de um provedor de identidade OpenID Connect (OIDC) como Salesforce ou Ping Identity.	17 de maio de 2018
Acionador do Lambda de migração do Amazon Cognito	Adicionadas páginas que abrangem o recurso Acionador do Lambda de migração	8 de abril de 2018
Atualização do Guia do desenvolvedor do Amazon Cognito	Adição dos tópicos de nível superior "O que é o Amazon Cognito" e "Conceitos básicos do Amazon Cognito". Alguns cenários comuns foram adicionados e o índice dos grupos de usuários foi reorganizado. Adicionada um nova seção “Conceito s básicos dos grupos de usuários do Amazon Cognito”.	6 de abril de 2018
Segurança avançada beta do Amazon Cognito	Novos atributos de segurança foram adicionados para permitir que os desenvolv edores defendam aplicações e usuários contra bots mal-inten cionados, protejam contas de usuário em relação a credenciais comprometidas na internet e ajustem automatic amente os desafios necessári os para fazer login com base no risco calculado da tentativa de login.	28 de novembro de 2017

Integração do Amazon Pinpoint	Adicionada a capacidade de usar o Amazon Pinpoint para fornecer análises para suas aplicações de grupos de usuários do Amazon Cognito para enriquecer os dados do usuário para campanhas do Amazon Pinpoint.	26 de setembro de 2017
Recursos de federação e da UI das aplicações integradas dos grupos de usuários do Amazon Cognito	Adição de capacidade para permitir que os usuários façam login no grupo de usuários via Facebook, Google, Login with Amazon ou um provedor de identidade SAML. Foi adicionada uma interface de usuário de aplicativo integrada personalizável e suporte OAuth 2.0 com declarações personalizadas.	10 de agosto de 2017
Alterações de recursos relacionadas à conformidade com a PCI e a HIPAA	Adição de capacidade para permitir que os usuários usem um número de telefone ou endereço de e-mail como nome de usuário.	6 de julho de 2017
Grupos de usuários e recursos de controle de acesso por função	Adição de capacidade administrativa para criar e gerenciar grupos de usuários. Os administradores podem atribuir funções do IAM a usuários com base na associação ao grupo e nas regras criadas pelo administrador.	15 de dezembro de 2016

Atualização da documentação	Exemplos atualizados que mostram como usar AWS Lambda gatilhos com grupos de usuários.	27 de novembro de 2016
Atualização da documentação	Exemplos de código do iOS atualizados.	18 de novembro de 2016
Atualização da documentação	Adição de informações sobre fluxo de confirmação para contas de usuário.	9 de novembro de 2016
Criar recurso de contas de usuário	Adição de capacidade administrativa para criar contas de usuário por meio do console do Amazon Cognito e da API.	6 de outubro de 2016
Recurso de importação de usuários	Adição da função de importação em massa para grupos de usuários do Cognito. Use esse recurso para migrar os usuários do provedor de identidade existente para um grupo de usuários do Amazon Cognito.	1 de setembro de 2016
Disponibilidade geral dos grupos de usuários do Cognito	Adição do recurso de grupos de usuários do Cognito. Use esse recurso para criar e manter um diretório de usuário e adicionar cadastro e login ao aplicativo móvel ou web usando grupos de usuários.	28 de julho de 2016

Compatibilidade com o SAML	Adição de suporte para autenticação com provedores de identidade por meio do Security Assertion Markup Language 2.0 (SAML 2.0).	23 de junho de 2016
CloudTrail integração	Integração adicionada com AWS CloudTrail.	18 de fevereiro de 2016
Integração de eventos com o Lambda	Permite que você execute uma AWS Lambda função em resposta a eventos importantes no Amazon Cognito.	9 de abril de 2015
Fluxo de dados para o Amazon Kinesis	Fornece controle e insight sobre os fluxos de dados.	4 de março de 2015
Suporte ao OpenID Connect	Ativa o suporte para os provedores OpenID Connect.	23 de novembro de 2014
Sincronização por Push	Habilita o suporte para a sincronização por push silenciosa.	6 de novembro de 2014
Adição de suporte a identidades autenticadas por desenvolvedor	Permite que os desenvolvedores que têm seus próprios sistemas de gerenciamento de autenticação e identidade sejam tratados como um provedor de identidade no Amazon Cognito.	29 de setembro de 2014
Disponibilidade geral do Amazon Cognito		10 de julho de 2014

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.