aws

Guia do Desenvolvedor

Amazon Cognito

Amazon Cognito: Guia do Desenvolvedor

Table of Contents

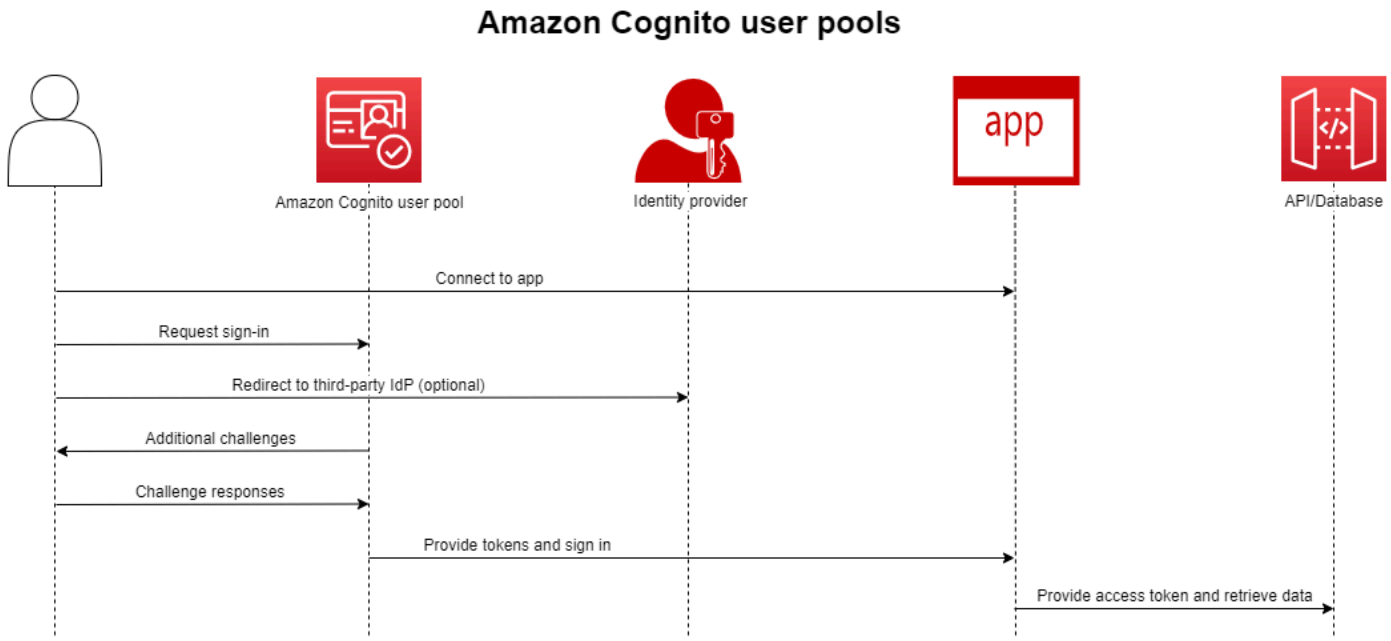
O que é o Amazon Cognito?

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 credenciais da AWS e tokens de acesso do 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.

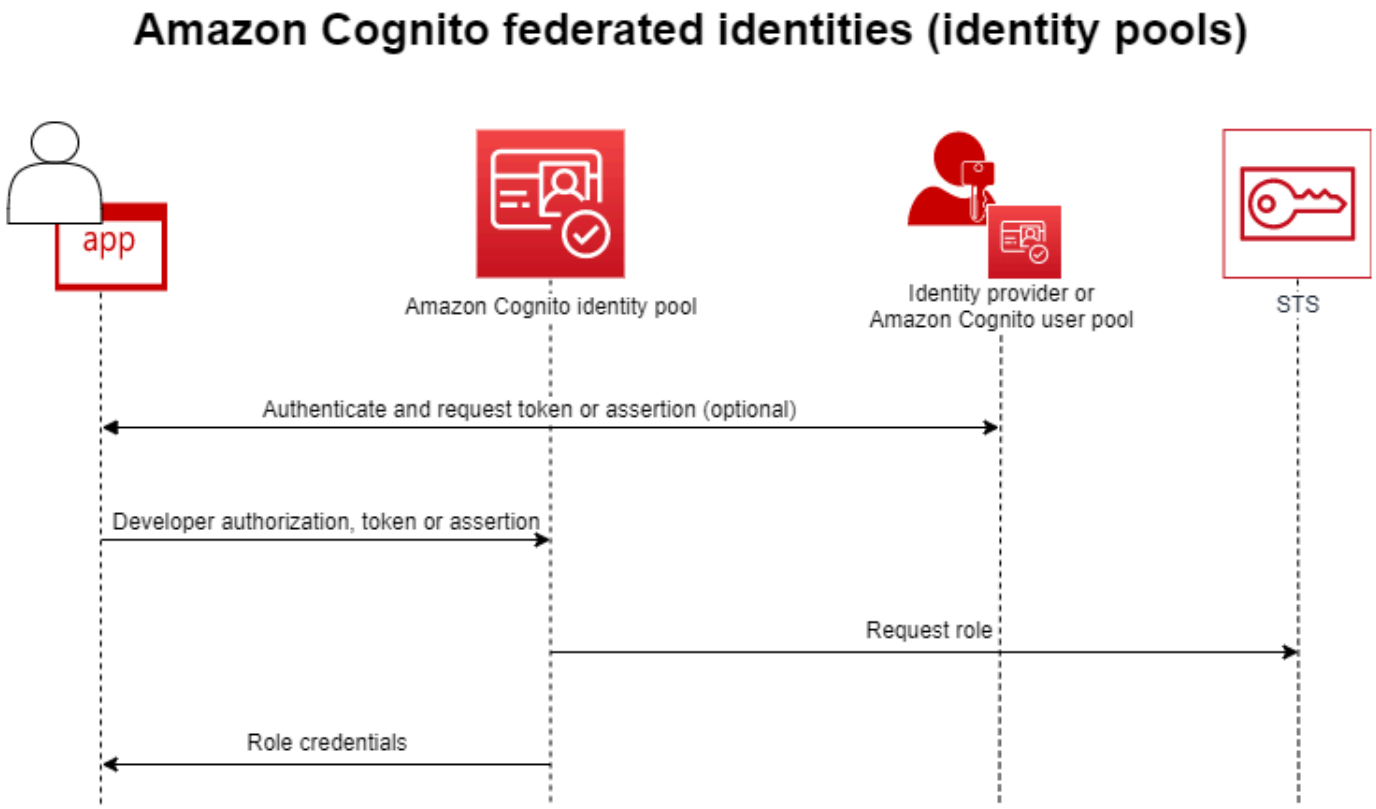
Grupos de usuários



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 autenticação única (SSO) em sua aplicação para identidades de clientes da sua organização nas lojas públicas de identidades do OAuth 2.0 da Amazon, do Google, da Apple e do 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 (JWTs) autenticados diretamente para uma aplicação, um servidor web ou uma API.

Grupos 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 da aplicação faz login por meio de um grupo de usuários e recebe tokens OAuth 2.0.
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 a segurança na nuvem no Amazon Cognito para estar em conformidade com SOC 1 a 3, ISO 27001 e HIPAA-BAA, mas não com 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 AWS Management Console, 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 gerencia as interações com IdPs, mapeando suas reivindicações em um formato de token central.

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

Atributo	Descrição
IdP OIDC	Emita tokens de ID para autenticar usuários
Servidor de autorização	Emita tokens de acesso para autorizar o acesso do usuário às APIs
SAML 2,0 COLHER DE CHÁ	Transforme declarações SAML em tokens de ID e acesso

OIDC SP	Transforme tokens OIDC em tokens de ID e acesso
OAuth 2.0 SP	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 a interface hospedada
Suporte de API para sua própria interface	Crie, gerencie e autentique usuários por meio de solicitações de API em SDKs suportados[1] AWS
MFA	Use mensagens SMS, TOTPs ou o dispositivo do 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]
Grupos	Crie agrupamentos lógicos de usuários e uma hierarquia de declarações de função do IAM ao passar tokens para grupos de identidades
Personalize tokens de ID	Personalize seus tokens de ID 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 está disponível somente para usuários locais.

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.

Grupos 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 Ao apresentar uma prova de autenticação em um banco de identidades em forma de declarações confiáveis de um provedor de identidades (IdP) social SAML 2.0, OpenID Connect (OIDC) ou OAuth 2.0, você associa o usuário a uma identidade no banco 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 sua própria prova de autenticação personalizada ou nenhuma autenticação. Você pode conceder AWS credenciais temporárias a qualquer usuário do aplicativo que as solicite, com identidades não autenticadas. Os bancos de identidades também aceitam declarações e emitem credenciais com base em seu próprio esquema personalizado, com identidades autenticadas pelo desenvolvedor.

Com os bancos de identidades do Amazon Cognito, 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

Atributo	Descrição
Grupo de usuários do Amazon Cognito SP	Troque um token de ID do seu grupo de usuários por credenciais de identidade da web de AWS STS
SAML 2,0 COLHER DE CHÁ	Declarações SAML do Exchange para credenciais de identidade na web de AWS STS
OIDC SP	Troque tokens OIDC por credenciais de identidade na web de AWS STS
OAuth 2.0 SP	Troque tokens OAuth da Amazon, Facebook, Google, Apple e Twitter por credenciais de identidade na web de AWS STS
SP personalizado	Com AWS credenciais, troque reivindicações em qualquer formato por credenciais de identidade na web de AWS STS
Acesso não autenticado	Emita credenciais de identidade 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 baseado em atributos	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 Introdução aos grupos de identidade 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

Atributo	Descrição	Grupos de usuários	Grupos de identidades
IdP OIDC	Emita tokens de ID OIDC para autenticar usuários do aplicativo	✓	
Servidor de autorização de API	Emita tokens de acesso para autorizar o acesso do usuário a APIs, bancos de dados e outros recursos que aceitam escopos de autorização do OAuth 2.0	✓	
Servidor de autorização de identidade web IAM	Gere tokens que você pode trocar AWS STS por AWS credenciais temporárias		✓
SAML 2.0 SP e IdP OIDC	Emita tokens OIDC personalizados com base em declarações de um IdP SAML 2.0	✓	
OIDC PS e OIDC IdP	Emita tokens OIDC personalizados com base em declarações de um IdP do OIDC	✓	

OAuth 2.0 PS e OIDC IdP	Emita tokens OIDC personalizados com base nos escopos dos provedores sociais do OAuth 2.0, como Apple e Google	✓	
SAML 2.0 SP e corretor de credencia is	Emitir AWS credencia is temporárias com base em declarações de um IdP do SAML 2.0		✓
OIDC SP e corretor de credenciais	Emitir AWS credencia is temporárias com base em declarações de um IdP do OIDC		✓
OAuth 2.0 SP e corretor de credencia is	Emita AWS credencia is temporárias com base nos escopos dos provedores sociais do OAuth 2.0, como Apple e Google		✓
Grupo de usuários e agente de credenciais do Amazon Cognito	Emita AWS credencia is temporárias com base em declaraçõ es do OIDC de um grupo de usuários do Amazon Cognito		✓
SP personalizado e corretor de credencia is	Emita AWS credencia is temporárias com base na autorização do IAM do desenvolv edor		✓

Serviço de front-end de autenticação	Cadastre, gerencie e autentique usuários com a interface hospedada	✓
Suporte de API para sua própria interface de autenticação	Crie, gerencie e autentique usuários por meio de solicitações de API em SDKs suportados[1] AWS	✓
MFA	Use mensagens SMS, TOTPs ou o dispositivo do 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]	✓
Grupos	Crie agrupamentos lógicos de usuários e uma hierarquia de declarações de função do IAM ao passar tokens para grupos de identidades	✓

Personalize tokens de ID	Personalize seus tokens de ID com reivindicações novas, modificadas e suprimidas	✓		
AWS WAF ACLs da web	Monitore e controle as solicitações para seu ambiente 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 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 suas funções para serem assumidas somente no contexto do seu grupo de identidades.			✓

Controle de acesso baseado em atributos	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 está disponível somente para usuários locais.

Conceitos básicos do Amazon Cognito

Por exemplo, aplicativos de grupos de usuários, consulteConceitos básicos dos grupos de usuários.

Para obter uma introdução aos grupos de identidades, consulteIntrodução aos grupos de identidade do Amazon Cognito.

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

Para vídeos, artigos, documentação e mais exemplos de aplicativos, consulte os recursos para desenvolvedores do Amazon Cognito.

Para usar o Amazon Cognito, você precisa de uma Conta da AWS. Para ter 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 obter uma lista de todas as regiões onde o Amazon Cognito está disponível no momento, consulte Regiões e endpoints da AWS na 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.

Como a autenticação funciona com grupos de usuários e grupos de identidades do 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 AWS SDK. Se você não quiser criar seus próprios componentes de interface de usuário (UI), você pode invocar uma interface web pré-criada (a interface hospedada) ou a página de login do seu provedor de identidade (IdP) terceirizado.

Este tópico é uma visão geral de algumas 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 e autorização da API do grupo de usuários com um AWS SDK
- Autenticação do grupo de usuários com a interface hospedada
- Autenticação do grupo de usuários com um provedor de identidade de terceiros
- Autenticação do pool de

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

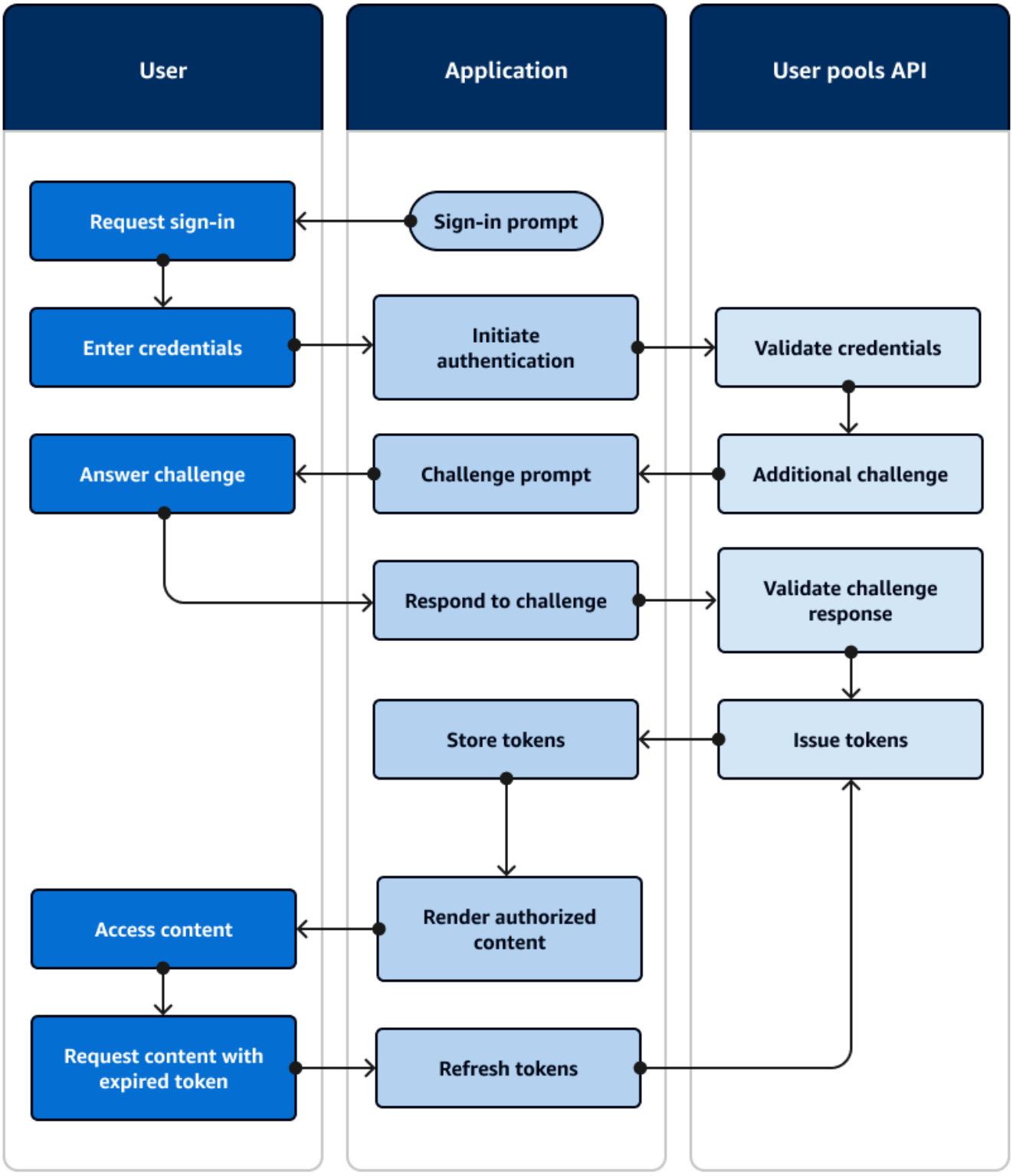
incorporados nesses SDKs chamam a API de grupos de usuários do Amazon Cognito. O mesmo namespace da API de grupos de usuários tem operações para configuração de grupos de usuários e para autenticação de usuários. Para uma visão geral mais completa, consulteUsar a API de grupos de usuários e endpoints de grupo de usuários do Amazon Cognito.

A autenticação de API se ajusta ao modelo em que seus aplicativos 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 um aplicativo maior. Ela exige lógica programática para lidar com cadeias complexas de desafios e respostas.

Esse aplicativo não precisa implementar uma implementação completa de terceiros confiáveis do 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 sua autenticação com o SDK do Amazon Cognito em seu ambiente de desenvolvimento.

A autenticação de API com escopos OAuth 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 tempo de execução com umAcionador do Lambda antes da geração do token.

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


User
Application
User pools API
Request sign-in
Sign-in prompt
Enter credentials
Initiate authentication
Validate credentials
Answer challenge
Challenge prompt
Additional challenge
Respond to challenge
Validate challenge response
Store tokens
Issue tokens
Access content
Render authorized content
Request content with expired token
Refresh tokens

Fluxo de autenticação da API

1. Um usuário acessa seu aplicativo.
2. Eles selecionam um link “Fazer login”.
3. Eles inserem seu 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 se o usuário ativou a autenticação multifator (MFA).
6. O grupo de usuários responde com um desafio que solicita um código de MFA.
7. O aplicativo 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 os JWTs do usuário.
11. O aplicativo decodifica, valida e armazena ou armazena em cache os JWTs do usuário.
12. O aplicativo exibe o componente de controle de acesso solicitado.
13. O usuário visualiza seu conteúdo.
14. Posteriormente, o token de acesso do usuário expirou e ele solicitou a visualização de um componente de acesso controlado.
15. O aplicativo 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 aumentar 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 possam indicar uma tentativa de login mal-intencionada. Esse fluxo é praticamente o mesmo para operações de inscrição, 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
- Usar a API de grupos de usuários e endpoints de grupo de usuários do Amazon Cognito

Autenticação do grupo de usuários com a interface hospedada

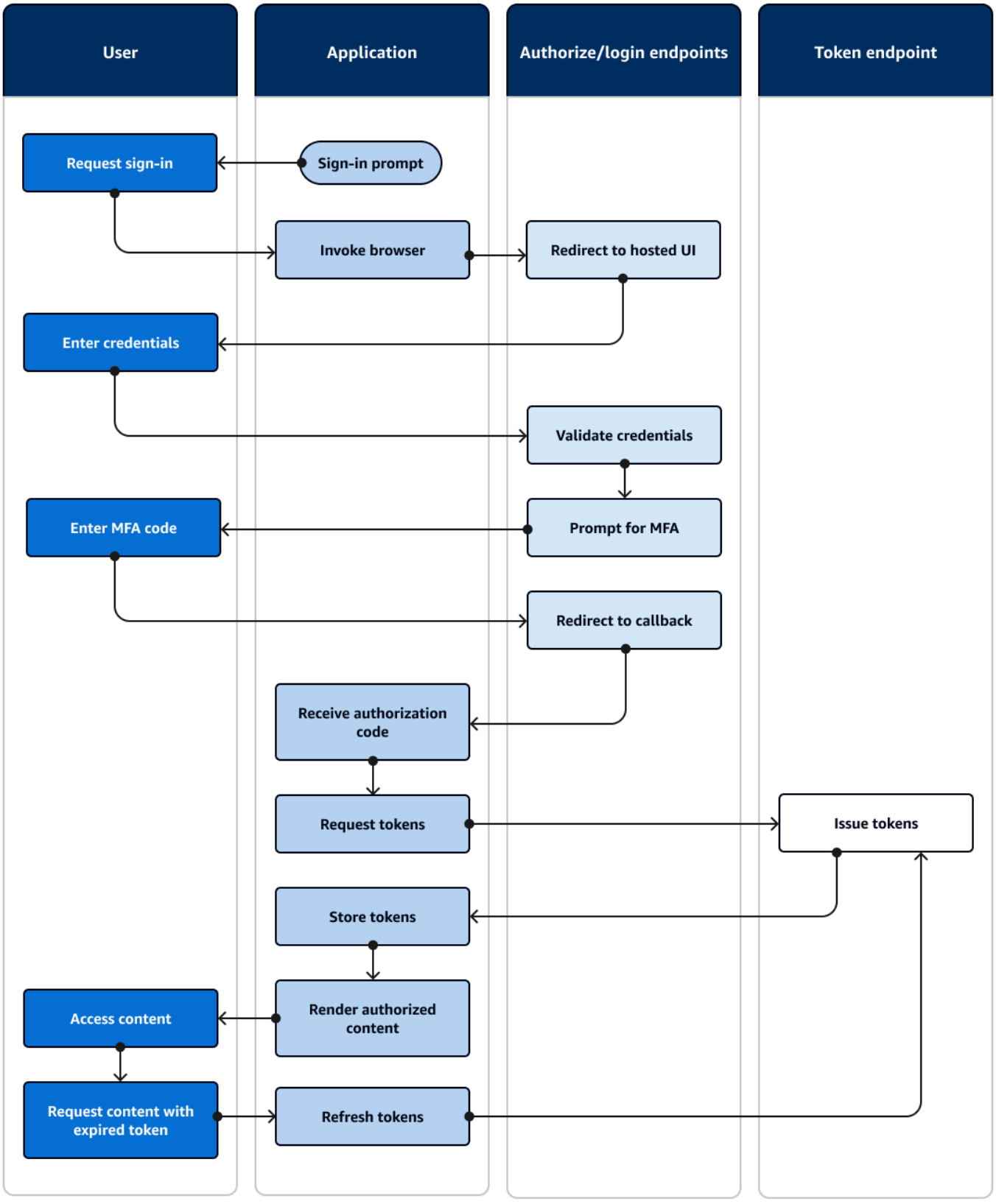
A interface de usuário hospedada é um site vinculado ao seu grupo de usuários e ao cliente do aplicativo. Ele pode realizar operações de login, inscrição e redefinição de senha para seus usuários. Um aplicativo com um componente de interface de usuário hospedado para autenticação pode exigir menos esforço do desenvolvedor para ser implementado. Um aplicativo pode ignorar os componentes da interface do usuário para autenticação e invocar a interface hospedada no navegador do usuário.

Os aplicativos coletam os JWTs dos usuários com um local de redirecionamento da web ou do aplicativo. Os aplicativos que implementam a interface de usuário hospedada podem se conectar a grupos de usuários para autenticação como se fossem um IdP do OpenID Connect (OIDC).

A autenticação de interface de usuário hospedada se encaixa no modelo em que os aplicativos precisam de um servidor de autorização, mas não precisam de recursos como autenticação personalizada, integração de grupos de identidades ou autoatendimento de atributos do usuário. Quando quiser usar algumas dessas opções avançadas, você pode implementá-las com um componente de grupos de usuários para um SDK.

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

O diagrama a seguir ilustra uma sessão de login típica para autenticação de interface de usuário hospedada.


User
Application
Authorize/login endpoints
Token endpoint
Request sign-in
Sign-in prompt
Invoke browser
Redirect to hosted UI
Enter credentials
Validate credentials
Enter MFA code
Prompt for MFA
Redirect to callback
Receive authorization code
Request tokens
Issue tokens
Store tokens
Access content
Render authorized content
Request content with expired token
Refresh tokens

Fluxo de autenticação de UI hospedada

1. Um usuário acessa seu aplicativo.
2. Eles selecionam um link “Fazer login”.
3. O aplicativo direciona o usuário para um prompt de login de UI hospedado.
4. Eles inserem seu nome de usuário e senha.
5. O grupo de usuários valida as credenciais do usuário e determina se o usuário ativou a autenticação multifator (MFA).
6. A interface do usuário hospedada solicita que o usuário insira um código de MFA.
7. O usuário insere seu código de MFA.
8. A interface do usuário hospedada redireciona o usuário para o aplicativo.
9. O aplicativo coleta o código de autorização do parâmetro de solicitação de URL que a interface hospedada anexou ao URL de retorno de chamada.
10. O aplicativo solicita tokens com o código de autorização.
11. O endpoint do token retorna JWTs para o aplicativo.
12. O aplicativo decodifica, valida e armazena ou armazena em cache os JWTs do usuário.
13. O aplicativo exibe o componente de controle de acesso solicitado.
14. O usuário visualiza seu conteúdo.
15. Posteriormente, o token de acesso do usuário expirou e ele solicitou a visualização de um componente de acesso controlado.
16. O aplicativo determina que a sessão do usuário deve persistir. Ele solicita novos tokens do endpoint do token com o token de atualização.

Variantes e personalização

Você pode personalizar a aparência da interface do usuário hospedada com CSS em qualquer cliente de aplicativo. Você também pode configurar clientes de aplicativos com seus próprios provedores de identidade, escopos, acesso aos atributos do usuário e configuração de segurança avançada.

Recursos relacionados

- Configurar e usar a interface de usuário hospedada e endpoints de federação do Amazon Cognito
- Cadastrar-se e fazer login com a UI hospedada

- Escopos, M2M e autorização de API com servidores de recursos
- Referência da interface do usuário hospedada e endpoints de federação do grupo de usuários

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

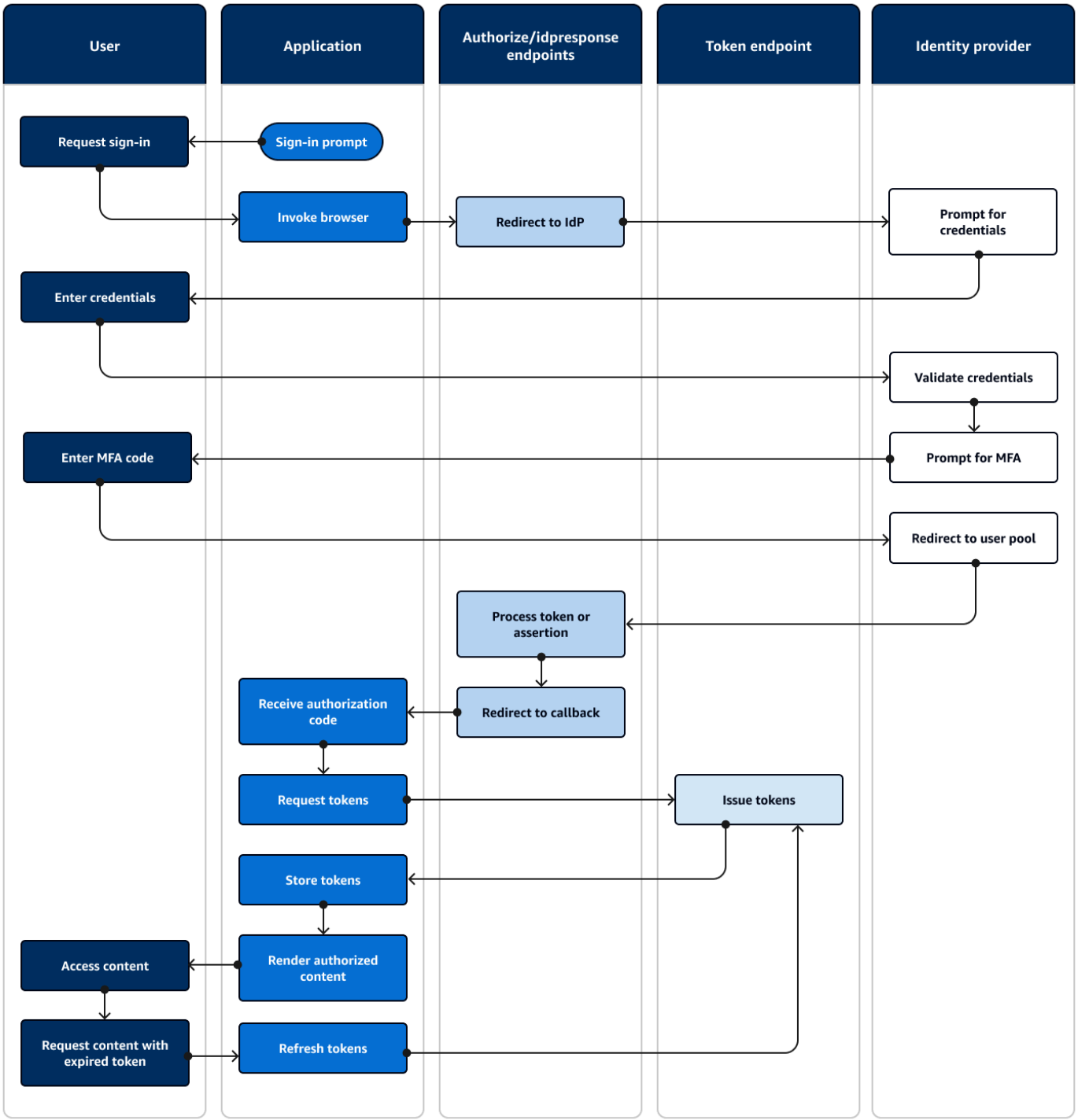
O login com um provedor de identidade externo (IdP), ou autenticação federada, é um modelo semelhante à interface de usuário hospedada. Seu aplicativo é uma parte dependente do OIDC em seu grupo de usuários, enquanto seu grupo de usuários serve como uma passagem para um IdP. O IdP pode ser um diretório de usuários consumidores, como Facebook ou Google, ou pode ser um diretório corporativo SAML 2.0 ou OIDC, como o Azure.

Em vez da interface de usuário hospedada no navegador do usuário, seu aplicativo 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 em seu aplicativo. Em seguida, o IdP solicita que eles façam login. Assim como na autenticação de interface de usuário hospedada, um aplicativo coleta JWTs em um local de redirecionamento no aplicativo.

A autenticação com um IdP de terceiros se encaixa em um modelo em que os usuários talvez não queiram criar uma nova senha ao se inscreverem no seu aplicativo. A autenticação de terceiros pode ser adicionada com pouco esforço a um aplicativo que implementou a autenticação de interface de usuário hospedada. Na verdade, a interface de usuário hospedada e 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 interface de usuário hospedada, a autenticação federada é melhor para modelos de autorização avançados com escopos do OAuth 2.0.

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



Fluxo de autenticação federada

1. Um usuário acessa seu aplicativo.
2. Eles selecionam um link “Fazer login”.

3. O aplicativo direciona o usuário para um prompt de login com seu IdP.
4. Eles inserem seu nome de usuário e senha.
5. O IdP valida as credenciais do usuário e determina se o usuário ativou a 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 trocará silenciosamente o código por tokens IdP. O grupo de usuários valida os tokens do IdP e redireciona o usuário para o aplicativo com um novo código de autorização.
10. O aplicativo coleta o código de autorização do parâmetro de solicitação de URL que o grupo de usuários anexou ao URL de retorno de chamada.
11. O aplicativo solicita tokens com o código de autorização.
12. O endpoint do token retorna JWTs para o aplicativo.
13. O aplicativo decodifica, valida e armazena ou armazena em cache os JWTs do usuário.
14. O aplicativo exibe o componente de controle de acesso solicitado.
15. O usuário visualiza seu conteúdo.
16. Posteriormente, o token de acesso do usuário expirou e ele solicitou a visualização de um componente de acesso controlado.
17. O aplicativo determina que a sessão do usuário deve persistir. Ele solicita novos tokens do endpoint do token com o token de atualização.

Variantes e personalização

Você pode iniciar a autenticação federada na interface hospedada, na qual os usuários podem escolher em uma lista das IdPs que você atribuiu ao seu cliente de aplicativo. A interface de usuário hospedada 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 identidade terceirizado não exige interação do usuário com a interface hospedada. Seu aplicativo pode adicionar um parâmetro de solicitação à solicitação do servidor de autorização do usuário e fazer com que o usuário redirecione silenciosamente para a página de login do IdP.

Recursos relacionados

- Como adicionar acesso a grupo de usuários por meio de terceiros
- Exemplo de cenário: marcar aplicativos do Amazon Cognito como favoritos em um painel corporativo
- Escopos, M2M e autorização de API com servidores de recursos
- Referência da interface do usuário hospedada e endpoints de federação do grupo de usuários

Autenticação do pool de

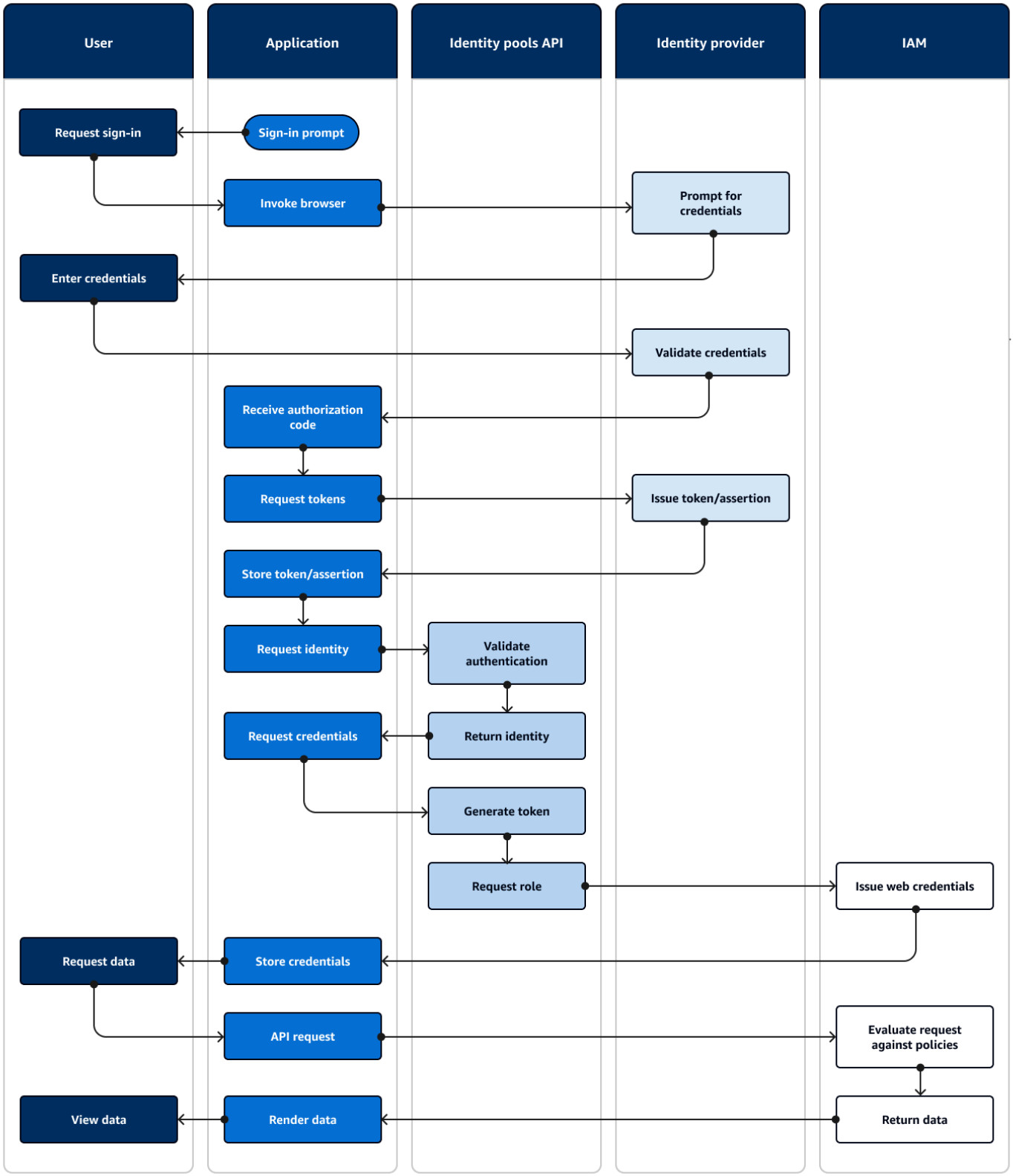
Um grupo de identidades é um componente do seu aplicativo que é diferente de um grupo de usuários em função, namespace de API e modelo de SDK. Onde os grupos de usuários oferecem autenticação e autorização baseadas em tokens, os grupos de identidades oferecem autorização para AWS Identity and Access 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 seleção de opções de autenticação para grupos 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 grupo 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 seu aplicativo realiza a autenticação inicial, ele passa a prova para um grupo 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 na autenticação de API em grupos de usuários, um aplicativo bem-sucedido inclui AWS SDKs para cada um dos serviços que você deseja acessar para o benefício de seus usuários. AWS Os SDKs aplicam 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 de grupos de identidades com um IdP.


User
Application
Identity pools API
Identity provider
IAM
Request sign-in
Sign-in prompt
Invoke browser
Prompt for credentials
Enter credentials
Validate credentials
Receive authorization code
Request tokens
Issue token/assertion
Store token/assertion
Request identity
Validate authentication
Request credentials
Return identity
Generate token
Request role
Issue web credentials
Request data
Store credentials
API request
Evaluate request against policies
View data
Render data
Return data

Fluxo de autenticação federada

1. Um usuário acessa seu aplicativo.
2. Eles selecionam um link “Fazer login”.
3. O aplicativo direciona o usuário para um prompt de login com seu IdP.
4. Eles inserem seu nome de usuário e senha.
5. O IdP valida as credenciais do usuário.
6. O IdP redireciona o usuário para o aplicativo com uma resposta SAML ou um código de autorização.
7. Se o usuário passou um código de autorização, o aplicativo troca o código por tokens IdP.
8. O aplicativo decodifica, valida e armazena ou armazena em cache as JWTs ou a afirmação do 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 de identidade.
10. O grupo de identidades valida o token ou a afirmação em relação aos provedores de identidade configurados.
11. O grupo de identidades retorna uma ID de identidade.
12. O aplicativo invoca o método que faz uma solicitação de GetCredentialsForIdentityAPI. Ele passa o token ou as afirmações do usuário e solicita uma função do IAM.
13. O pool de identidades gera um novo JWT. O novo JWT contém declarações que solicitam uma função do IAM. O grupo de identidades determina a função com base na solicitação do usuário e nos critérios de seleção de função na configuração do grupo 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 uma função do IAM.
15. O aplicativo armazena as credenciais da sessão.
16. O usuário executa uma ação no aplicativo que requer acesso a recursos protegidos. 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 à função nas credenciais. Isso os compara com a solicitação.
19. O AWS service (Serviço da AWS) retorna os dados solicitados.
20. O aplicativo 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 token/asserção de problema. A autenticação do desenvolvedor substitui todas as etapas anteriores à identidade da solicitação 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 uma função do IAM de acesso limitado.

Recursos relacionados

- Banco de identidades do Amazon Cognito
- Funções do IAM do usuário
- Conceitos de grupos de identidades
- Fluxo de autenticação dos grupos de identidades (identidades federadas)

Termos do Amazon Cognito

O Amazon Cognito fornece credenciais para aplicativos web e móveis. Ele se baseia e se baseia em termos comuns no gerenciamento de identidade e acesso. Muitos guias sobre identidade universal e termos de acesso estão disponíveis. Alguns exemplos são:

- Terminologia no corpo de conhecimento do IDPro
- AWS Serviços de identidade
- Glossário do NIST CSRC

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

Tópicos

- Geral
- Grupos de usuários
- Grupos de identidades

Geral

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

App

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)

Um 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 principais em grupos de identidade.

Servidor de autorização

Um sistema baseado na web que gera tokens web JSON. Os endpoints de federação de grupos de usuários do Amazon Cognito são o componente do servidor de autorização dos dois métodos de autenticação e autorização nos grupos de usuários. O outro método é a API de grupos de usuários.

Aplicativo confidencial, aplicativo do lado do servidor

Um aplicativo ao qual os usuários se conectam remotamente, com código em um servidor de aplicativos e acesso a segredos. Normalmente, é um aplicativo da web.

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

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

Token web JSON (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 aplicativos ou. AWS STS

Autenticação multifator (MFA)

A exigência de que os usuários forneçam autenticação adicional após fornecerem seu 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 grupo de identidades que fornece acesso ao JWT e tokens de atualização. Os grupos de usuários do Amazon Cognito automatizam as interações com provedores sociais após a autenticação dos usuários.

Provedor do OpenID Connect (OIDC)

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

Aplicativo público

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

Um modelo que concede acesso com base na designação funcional do usuário. Os grupos de identidade do Amazon Cognito implementam o RBAC com diferenciação entre as funções do IAM.

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

Um aplicativo que depende de um IdP para afirmar que os usuários são confiáveis. O Amazon Cognito atua como um SP para SPs externos IdPs e como um IdP para SPs baseados em aplicativos.

Provedor SAML

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

Identificador universalmente exclusivo (UUID)

Um rótulo de 128 bits aplicado a um objeto. Os UUIDs do Amazon Cognito são exclusivos por grupo de usuários ou grupo de identidades.

Diretório de usuários

Uma coleção de usuários e seus atributos que fornece essas informações para outros sistemas. Os grupos de usuários do Amazon Cognito são diretórios de usuários e também ferramentas para consolidação de usuários de diretórios de usuários externos.

Grupos de usuários

Quando você vê os termos na lista a seguir neste guia, eles se referem a um recurso ou configuração específica dos grupos de usuários.

API de grupos de usuários do Amazon Cognito

Um conjunto de operações de API de autenticação e autorização que você pode adicionar ao seu aplicativo com um AWS SDK. A API pode cadastrar usuários locais e usuários vinculados.

Autenticação adaptável

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

Recursos avançados de segurança

Um componente opcional que adiciona ferramentas para a segurança do usuário.

Cliente de aplicativo

Um componente que define as configurações de um grupo de usuários como um IdP para um aplicativo.

URL de retorno de chamada, URI de redirecionamento

Uma configuração em um cliente de aplicativo e um parâmetro em solicitações para endpoints de federação de grupos de usuários. O URL de retorno de chamada é o destino inicial dos usuários autenticados no seu aplicativo.

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

O 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 gatilhos Lambda que definem desafios e respostas adicionais do usuário.

autenticação de dispositivos

Um processo de autenticação que substitui o MFA por um login que usa a ID de um dispositivo confiável.

Fornecedor externo, fornecedor terceirizado

Um IdP que tem uma relação de confiança com um grupo de usuários.

Usuário federado

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

Endpoints da federação

Um conjunto de páginas da Web em seu domínio de grupo de usuários que hospedam serviços para interação IdPs e aplicativos.

Interface do usuário hospedada

Um conjunto de páginas da Web interativas em seu domínio de grupo de usuários que hospedam serviços para autenticação de usuários.

Gatilho do Lambda

Uma função na AWS Lambda qual um grupo de usuários pode ser invocado automaticamente em pontos-chave nos processos de autenticação de usuários. Você pode usar os acionadores 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

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

Personalização de tokens

O resultado de um gatilho Lambda pré-geração de token que modifica o ID ou o token de acesso de um usuário em tempo de execução.

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

Um AWS recurso com serviços de autenticação e autorização para aplicativos que funcionam com o OIDC IdPs.

Domínio do grupo de usuários

Um nome de site que você adiciona a um grupo de usuários. O domínio é o URL base para a interface hospedada e os endpoints da federação.

Verificação

O processo de confirmação de que um usuário possui um endereço de e-mail ou número de telefone. Um grupo de usuários envia um código para um usuário que inseriu um novo endereço de e-mail ou número de telefone. Quando eles enviam o código para o Amazon Cognito, eles verificam a propriedade do destino da mensagem e podem receber mensagens adicionais do grupo de usuários. Além disso, veja a confirmação.

Perfil de usuário, conta de usuário

Uma entrada para um usuário no diretório do usuário. Todos os usuários têm um perfil em seu grupo de usuários.

Grupos de identidades

Quando você vê os termos na lista a seguir neste guia, eles se referem a um recurso ou configuração específica dos grupos de identidades.

Atributos para controle de acesso

Uma implementação de controle de acesso baseado em atributos em grupos de identidades. Os grupos de identidades aplicam atributos do usuário como tags às credenciais do usuário.

Autenticação básica (clássica)

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

Identidades autenticadas pelo desenvolvedor

Um processo de autenticação que autoriza as credenciais do usuário do grupo de identidades com as credenciais do desenvolvedor.

Credenciais de desenvolvedor

As chaves da API IAM de um administrador do grupo de identidades.

Autenticação avançada

Um fluxo de autenticação que seleciona uma função do IAM e aplica as tags principais de acordo com a lógica que você define no seu grupo de identidades.

Identidade

Um UUID que vincula um usuário do aplicativo e suas credenciais de usuário ao perfil em um diretório de usuário externo que tem uma relação de confiança com um grupo de identidades.

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

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

Identidade não autenticada do

Um usuário que não fez login com um IdP do grupo de identidades. Você pode permitir que os usuários gerem credenciais de usuário limitadas para uma única função do IAM antes da autenticação.

Credenciais do usuário

Chaves de AWS API temporárias que os usuários recebem após a autenticação do grupo 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 facilitam a criação de aplicações em seu idioma preferido pelos desenvolvedores.

Documentação do SDK	Exemplos de código
AWS SDK for C++	AWS SDK for C++ exemplos de código
AWS CLI	AWS CLI exemplos de código
AWS SDK for Go	AWS SDK for Go exemplos de código
AWS SDK for Java	AWS SDK for Java exemplos de código
AWS SDK for JavaScript	AWS SDK for JavaScript exemplos de código
AWS SDK para Kotlin	AWS SDK para Kotlin exemplos de código
AWS SDK for .NET	AWS SDK for .NET exemplos de código
AWS SDK for PHP	AWS SDK for PHP exemplos de código
AWS Tools for PowerShell	Ferramentas para exemplos PowerShell de código
AWS SDK for Python (Boto3)	AWS SDK for Python (Boto3) exemplos de código
AWS SDK for Ruby	AWS SDK for Ruby exemplos de código
AWS SDK para Rust	AWS SDK para Rust exemplos de código
SDK da AWS para SAP ABAP	SDK da AWS para SAP ABAP exemplos de código
AWS SDK for Swift	AWS SDK for Swift exemplos de código

Exemplo de disponibilidade

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

Começando com AWS

Antes de começar a trabalhar com o Amazon Cognito, prepare-se com alguns recursos necessários AWS .

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

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

   Quando você se 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, é possível visualizar as atividades da conta atual e gerenciar sua conta acessando https://aws.amazon.com/ e selecionando Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, 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 AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, digite sua senha.

   Para obter ajuda ao fazer login usando o usuário raiz, consulte Signing in as the root user (Fazer login como usuário raiz) no Guia do usuário Início de Sessão da AWS .

2. Ative a autenticação multifator (MFA) para seu 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. Habilitar o IAM Identity Center.

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

   Para 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 AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com seu usuário IAM Identity Center, use a URL de login enviada ao seu endereço de e-mail quando você criou o usuário IAM Identity Center user.

  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 Create a permission set no Guia do usuário do AWS IAM Identity Center .
2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Add groups no Guia do usuário do AWS IAM Identity Center .

Conceitos básicos dos grupos de usuários

Você pode usar os guias desta seção para criar seus recursos iniciais do grupo de usuários. Para uma step-by-step explicação passo a passo, comece com um aplicativo web básico no ambiente de JavaScript desenvolvedor do React. A partir daí, você pode continuar adicionando recursos como a interface de usuário hospedada (UI hospedada) e o login federado com provedores externos de identidade social ou SAML 2.0 (). IdPs

À medida que você trabalha para expandir seu conjunto de recursos e incorporar mais componentes do Amazon Cognito, leia o capítulo de grupos de usuários do Amazon Cognito para obter descrições completas de tudo o que você pode fazer com grupos de usuários.

O exemplo de grupo de usuários e aplicativo nesta seção demonstra uma integração básica dos recursos do aplicativo com os grupos de usuários do Amazon Cognito. Posteriormente, você pode ajustar seu grupo de usuários para usar mais opções disponíveis para você. Em seguida, você pode atualizar seu aplicativo para adotar novas APIs e interagir com a interface hospedada e. IdPs

O tutorial desta seção cria um aplicativo com uma interface de usuário personalizada e autenticação baseada em API com um AWS SDK. Os aplicativos que você cria dessa forma são ideais para autenticar usuários locais. Para começar com um aplicativo com uma interface de usuário pré-criada, tratamento automático de alguns recursos do grupo de usuários e autenticação de usuários federados, vá para. Adicionar um cliente de aplicativo com a interface hospedada

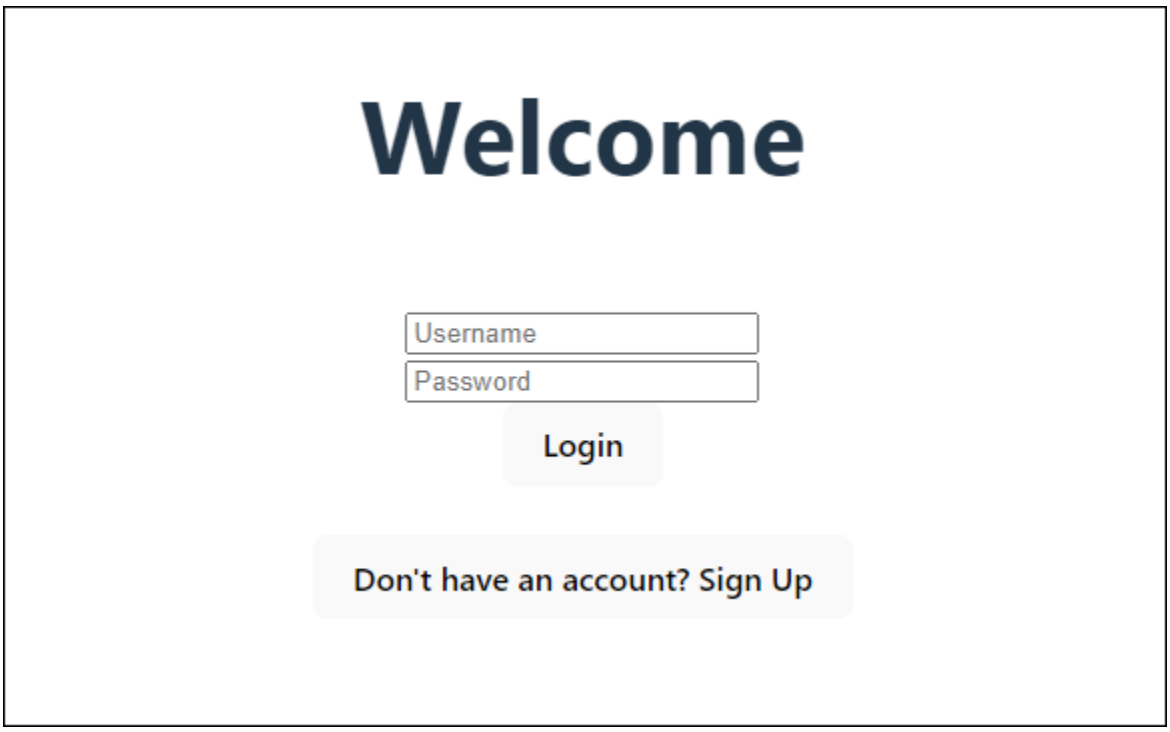
Tópicos

- Configure um exemplo de aplicativo de página única React
- Configure um aplicativo Android de exemplo com o Flutter
- Próximas etapas

Configure um exemplo de aplicativo de página única React

Neste tutorial, você criará um aplicativo de página única do React onde poderá 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 aplicativo de exemplo 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 no aplicativo que você criará.



O procedimento Criar um grupo de usuários configura você com um grupo de usuários que funciona com o aplicativo de exemplo. Você pode pular essa etapa se tiver um grupo de usuários que atenda aos seguintes requisitos:

- Os usuários podem fazer login com seu 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 Fazer distinção entre maiúsculas e minúsculas não está selecionada.
- A autenticação multifator (MFA) não é necessária. Aplicação do MFA: MFA 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 inscrever no seu grupo de usuários. Autorregistro: a opção Ativar autorregistro está selecionada.
- Seu cliente de aplicativo inicial é um cliente público que permite o login com nome de usuário e senha. Tipo de aplicativo: Cliente público, Fluxos de autenticação:`ALLOW_USER_PASSWORD_AUTH`.

Criar um grupo de usuários

Criar um novo grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha o botão Criar grupo de usuários. Talvez seja necessário selecionar Grupos de usuários no painel de navegação esquerdo para revelar essa opção.
3. No canto superior direito da página, escolha Create a user pool (Criar um grupo de usuários) para iniciar o assistente de criação de grupo de usuários.
4. Em Configurar a experiência de login, você pode escolher os provedores de identidade (IdPs) que você usará com esse grupo de usuários. Para ter mais informações, consulte Como adicionar acesso a grupo de usuários por meio de terceiros.
    a. Em Provedores de autenticação, para Tipos de provedor, certifique-se de que somente o grupo de usuários do Cognito esteja selecionado.
    b. Para opções de login do grupo de usuários do Cognito, escolha Nome do usuário. Não selecione nenhum requisito adicional de nome de usuário.
    c. Mantenha todas as outras opções como padrão e escolha Avançar.
5. Em Configurar requisitos de segurança, você pode escolher sua política de senha, requisitos de autenticação multifator (MFA) e opções de recuperação de conta de usuário. Para ter mais informações, consulte Usar atributos de segurança de grupos de usuários do Amazon Cognito.
    a. Para Política de senha, confirme se o modo de política de senha está definido para os padrões do Cognito.
    b. Em Autenticação multifator, para imposição de MFA, escolha MFA opcional.
    c. Para métodos de MFA, escolha Aplicativos autenticadores e mensagem SMS.
    d. Em Recuperação de conta de usuário, confirme se a opção Ativar recuperação de conta de autoatendimento está selecionada e se o método de entrega da mensagem de recuperação de conta de usuário está definido como Somente e-mail.
    e. Mantenha todas as outras opções como padrão e escolha Avançar.
6. Em Configurar a experiência de inscrição, você pode determinar como os novos usuários verificarão suas identidades ao se inscreverem como novos usuários e quais atributos devem ser obrigatórios ou opcionais durante o fluxo de inscrição do usuário. Para ter mais informações, consulte Como gerenciar usuários em seu grupo de usuários.

a. Confirme se a opção Ativar autorregistro está selecionada. Essa configuração abre seu grupo de usuários para que qualquer pessoa se inscreva na Internet. Isso se destina aos propósitos do aplicativo de exemplo, mas aplique essa configuração com cuidado em ambientes de produção.

b. Em Verificação e confirmação assistidas pelo Cognito, verifique se a caixa de seleção Permitir que o Cognito envie mensagens automaticamente para verificação e confirmação está marcada.

c. Confirme se os Atributos a serem verificados estão definidos como Enviar mensagem de e-mail, verificar endereço de e-mail.

d. Em Verificando alterações de atributos, confirme se as opções padrão estão selecionadas: Manter o valor do atributo original quando uma atualização está pendente é selecionado e Valores de atributos ativos quando uma atualização está pendente está definido como Endereço de e-mail.

e. Em Atributos obrigatórios, confirme se os atributos obrigatórios com base nas seleções anteriores exibem e-mail.

> **Important**
>
> Para este aplicativo de exemplo, seu grupo de usuários não deve definir phone_number como um atributo obrigatório. Se phone_number for exibido como um atributo obrigatório, revise e atualize suas escolhas anteriores:
>
> - MFA opcional, e-mail somente para método de entrega para mensagens de recuperação de conta de usuário
> - Envie uma mensagem de e-mail, verifique o endereço de e-mail dos Atributos para verificar

f. Mantenha todas as outras opções como padrão e escolha Avançar.

7. Em Configurar entrega de mensagens, você pode configurar a integração com o Amazon Simple Email Service e o Amazon Simple Notification Service para enviar mensagens de e-mail e SMS aos seus usuários para cadastro, confirmação de conta, MFA e recuperação de conta. 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. Em Provedor de e-mail, escolha Enviar e-mail com o Cognito e use o remetente de e-mail padrão fornecido pelo Amazon Cognito. Essa configuração para baixo volume de e-mails é

suficiente para testes de aplicativos. Você pode retornar depois de verificar um endereço de e-mail com o Amazon Simple Email Service (Amazon SES) e escolher Enviar e-mail com o Amazon SES.

b. Para SMS, selecione Criar uma nova função do IAM e insira um nome de função do IAM. Isso cria uma função que concede permissões ao Amazon Cognito para enviar mensagens SMS.

c. Mantenha todas as outras opções como padrão e escolha Avançar.

8. Em Integrar seu aplicativo, você pode nomear seu grupo de usuários, configurar a interface hospedada e criar um cliente de aplicativo. Para ter mais informações, consulte Adicionar um cliente de aplicativo com a interface hospedada. Os aplicativos de exemplo não usam a interface de usuário hospedada.

   a. Em Nome do grupo de usuários, insira um nome do grupo de usuários.

   b. Não selecione Usar a interface hospedada do Cognito.

   c. Em Cliente de aplicativo inicial, confirme se o tipo de aplicativo está definido como Cliente público.

   d. Em Segredo do cliente, confirme se a opção Não gerar uma chave secreta do cliente está selecionada.

   e. Insira um App client name (Nome do cliente da aplicação).

   f. Expanda Configurações avançadas do cliente de aplicativos. Adicione `ALLOW_USER_PASSWORD_AUTH` à lista de fluxos de autenticação.

   g. Mantenha todas as outras opções como padrão e escolha Avançar.

9. Revise suas escolhas na tela Revisar e criar e modifique as seleções conforme necessário. Quando estiver satisfeito com a configuração do grupo de usuários, escolha Criar grupo de usuários para continuar.

10. Na página Grupos de usuários, escolha seu novo grupo de usuários.

11. Em Visão geral do grupo de usuários, anote seu ID do grupo de usuários. Você fornecerá essa string ao criar seu aplicativo de exemplo.

12. Escolha a guia Integração de aplicativos e localize a seção Clientes e análises de aplicativos. Selecione seu novo cliente de aplicativo. Anote seu ID de cliente.

Recursos relacionados

- Grupos de usuários do Amazon Cognito

- Fluxo de autenticação de grupo de usuários
- Como usar tokens com grupos de usuários

Cria uma aplicação

Para criar esse aplicativo, você deve configurar um ambiente de desenvolvedor. Os requisitos do ambiente do desenvolvedor são:

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 na porta TCP 5173 em um navegador da Web.

Para criar um exemplo de aplicativo web React

1. Faça login em seu ambiente de desenvolvedor e navegue até o diretório principal do seu aplicativo.

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

2. Crie um novo serviço 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/frontend-client
```

4. Navegue até o `src` diretório em seu projeto.

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

5. Edite `config.ts` e substitua os seguintes valores:

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

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

6. Se você quiser acessar seu aplicativo 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 seu aplicativo.

   ```
   npm install
   ```

8. Inicie o aplicativo.

   ```
   npm run dev
   ```

9. Acesse o aplicativo em um navegador da web em `http://localhost:5173` ou`http://[IP address]:5173`.

10. Inscreva um novo usuário com um endereço de e-mail válido.

11. Recupere o código de confirmação da sua mensagem de e-mail. Insira o código de confirmação no aplicativo.

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.

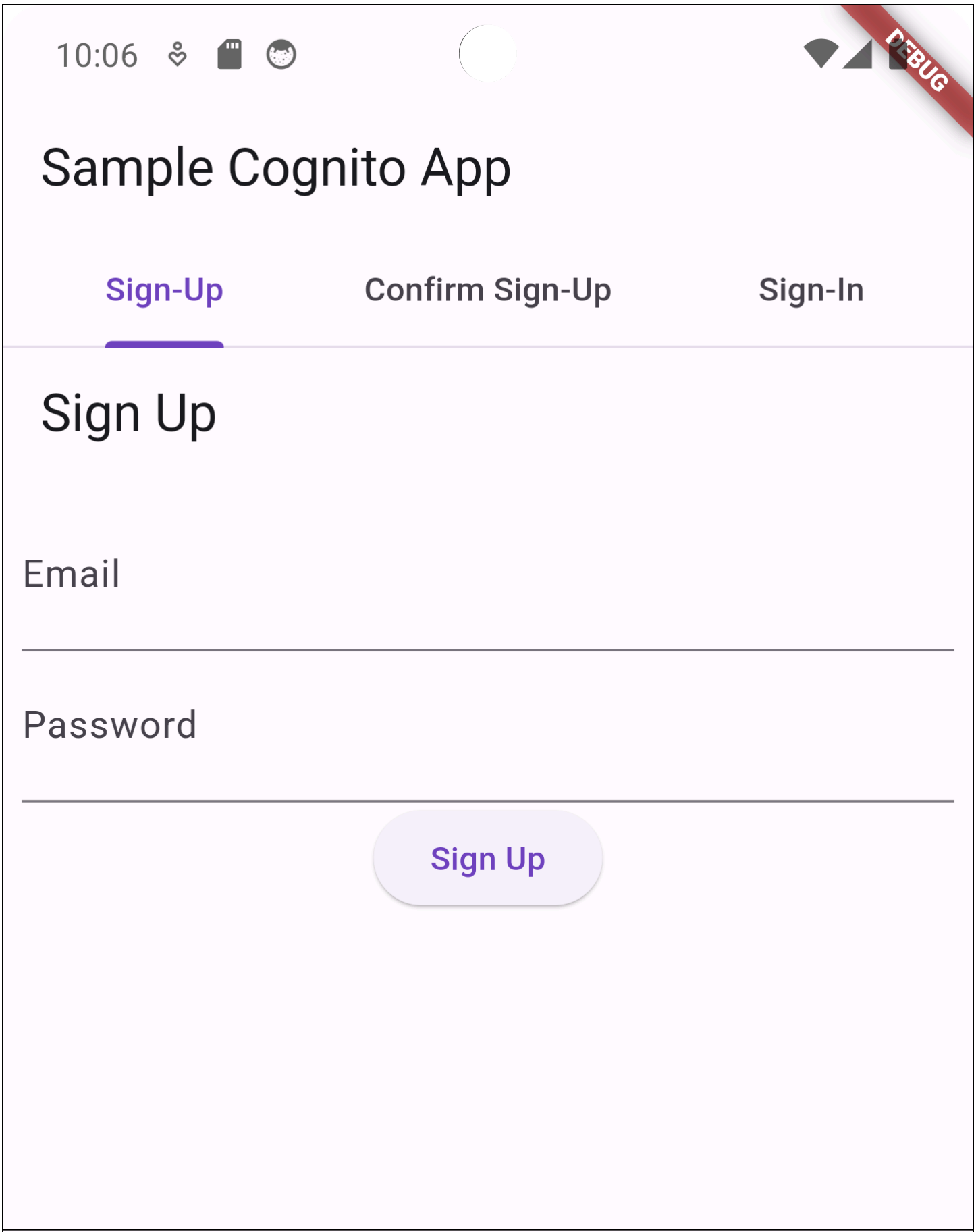
Para criar uma instância do Lightsail para esse aplicativo de exemplo

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 um blueprint, escolha 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 confirme `npm -v` se sua instância foi provisionada com Node.js e a versão mínima de npm 10.2.3.
9. Continue configurando seu aplicativo React.

Configure um aplicativo Android de exemplo com o Flutter

Neste tutorial, você criará um aplicativo móvel no Android Studio onde poderá emular um dispositivo e testar a inscrição, a confirmação e o login do usuário. Este aplicativo 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 o aplicativo em execução em um dispositivo Android virtual.


10:06
DEBUG
Sample Cognito App
Sign-Up
Confirm Sign-Up
Sign-In
Sign Up
Email
Password
Sign Up

O procedimento Criar um grupo de usuários configura você com um grupo de usuários que funciona com o aplicativo de exemplo. Você pode pular essa etapa se tiver um grupo de usuários que atenda aos seguintes requisitos:

- Os usuários podem fazer login com seu 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 Fazer distinção entre maiúsculas e minúsculas não está selecionada.
- A autenticação multifator (MFA) não é necessária. Aplicação do MFA: MFA 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 inscrever no seu grupo de usuários. Autorregistro: a opção Ativar autorregistro está selecionada.
- Seu cliente de aplicativo inicial é um cliente público que permite o login com nome de usuário e senha. Tipo de aplicativo: Cliente público, Fluxos de autenticação:`ALLOW_USER_PASSWORD_AUTH`.

Criar um grupo de usuários

Criar um novo grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha o botão Criar grupo de usuários. Talvez seja necessário selecionar Grupos de usuários no painel de navegação esquerdo para revelar essa opção.
3. No canto superior direito da página, escolha Create a user pool (Criar um grupo de usuários) para iniciar o assistente de criação de grupo de usuários.
4. Em Configurar a experiência de login, você pode escolher os provedores de identidade (IdPs) que você usará com esse grupo de usuários. Para ter mais informações, consulte Como adicionar acesso a grupo de usuários por meio de terceiros.

    a. Em Provedores de autenticação, para Tipos de provedor, certifique-se de que somente o grupo de usuários do Cognito esteja selecionado.

    b. Para opções de login do grupo de usuários do Cognito, escolha Nome do usuário. Não selecione nenhum requisito adicional de nome de usuário.

c. Mantenha todas as outras opções como padrão e escolha Avançar.

5. Em Configurar requisitos de segurança, você pode escolher sua política de senha, requisitos de autenticação multifator (MFA) e opções de recuperação de conta de usuário. Para ter mais informações, consulte Usar atributos de segurança de grupos de usuários do Amazon Cognito.

   a. Para Política de senha, confirme se o modo de política de senha está definido para os padrões do Cognito.

   b. Em Autenticação multifator, para imposição de MFA, escolha MFA opcional.

   c. Para métodos de MFA, escolha Aplicativos autenticadores e mensagem SMS.

   d. Em Recuperação de conta de usuário, confirme se a opção Ativar recuperação de conta de autoatendimento está selecionada e se o método de entrega da mensagem de recuperação de conta de usuário está definido como Somente e-mail.

   e. Mantenha todas as outras opções como padrão e escolha Avançar.

6. Em Configurar a experiência de inscrição, você pode determinar como os novos usuários verificarão suas identidades ao se inscreverem como novos usuários e quais atributos devem ser obrigatórios ou opcionais durante o fluxo de inscrição do usuário. Para ter mais informações, consulte Como gerenciar usuários em seu grupo de usuários.

   a. Confirme se a opção Ativar autorregistro está selecionada. Essa configuração abre seu grupo de usuários para que qualquer pessoa se inscreva na Internet. Isso se destina aos propósitos do aplicativo de exemplo, mas aplique essa configuração com cuidado em ambientes de produção.

   b. Em Verificação e confirmação assistidas pelo Cognito, verifique se a caixa de seleção Permitir que o Cognito envie mensagens automaticamente para verificação e confirmação está marcada.

   c. Confirme se os Atributos a serem verificados estão definidos como Enviar mensagem de e-mail, verificar endereço de e-mail.

   d. Em Verificando alterações de atributos, confirme se as opções padrão estão selecionadas: Manter o valor do atributo original quando uma atualização está pendente é selecionado e Valores de atributos ativos quando uma atualização está pendente está definido como Endereço de e-mail.

   e. Em Atributos obrigatórios, confirme se os atributos obrigatórios com base nas seleções anteriores exibem e-mail.

> **Important**
>
> Para este aplicativo de exemplo, seu grupo de usuários não deve definir phone_number como um atributo obrigatório. Se phone_number for exibido como um atributo obrigatório, revise e atualize suas escolhas anteriores:
>
> - MFA opcional, e-mail somente para método de entrega para mensagens de recuperação de conta de usuário
> - Envie uma mensagem de e-mail, verifique o endereço de e-mail dos Atributos para verificar

f. Mantenha todas as outras opções como padrão e escolha Avançar.

7. Em Configurar entrega de mensagens, você pode configurar a integração com o Amazon Simple Email Service e o Amazon Simple Notification Service para enviar mensagens de e-mail e SMS aos seus usuários para cadastro, confirmação de conta, MFA e recuperação de conta. 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. Em Provedor de e-mail, escolha Enviar e-mail com o Cognito e use o remetente de e-mail padrão fornecido pelo Amazon Cognito. Essa configuração para baixo volume de e-mails é suficiente para testes de aplicativos. Você pode retornar depois de verificar um endereço de e-mail com o Amazon Simple Email Service (Amazon SES) e escolher Enviar e-mail com o Amazon SES.

   b. Para SMS, selecione Criar uma nova função do IAM e insira um nome de função do IAM. Isso cria uma função que concede permissões ao Amazon Cognito para enviar mensagens SMS.

   c. Mantenha todas as outras opções como padrão e escolha Avançar.

8. Em Integrar seu aplicativo, você pode nomear seu grupo de usuários, configurar a interface hospedada e criar um cliente de aplicativo. Para ter mais informações, consulte Adicionar um cliente de aplicativo com a interface hospedada. Os aplicativos de exemplo não usam a interface de usuário hospedada.

   a. Em Nome do grupo de usuários, insira um nome do grupo de usuários.

   b. Não selecione Usar a interface hospedada do Cognito.

   c. Em Cliente de aplicativo inicial, confirme se o tipo de aplicativo está definido como Cliente público.

   d. Em Segredo do cliente, confirme se a opção Não gerar uma chave secreta do cliente está selecionada.
   e. Insira um App client name (Nome do cliente da aplicação).
   f. Expanda Configurações avançadas do cliente de aplicativos. Adicione `ALLOW_USER_PASSWORD_AUTH` à lista de fluxos de autenticação.
   g. Mantenha todas as outras opções como padrão e escolha Avançar.
9. Revise suas escolhas na tela Revisar e criar e modifique as seleções conforme necessário. Quando estiver satisfeito com a configuração do grupo de usuários, escolha Criar grupo de usuários para continuar.
10. Na página Grupos de usuários, escolha seu novo grupo de usuários.
11. Em Visão geral do grupo de usuários, anote seu ID do grupo de usuários. Você fornecerá essa string ao criar seu aplicativo de exemplo.
12. Escolha a guia Integração de aplicativos e localize a seção Clientes e análises de aplicativos. Selecione seu novo cliente de aplicativo. Anote seu ID de cliente.

Recursos relacionados

- Grupos de usuários do Amazon Cognito
- Fluxo de autenticação de grupo de usuários
- Como usar tokens com grupos de usuários

Cria uma aplicação

Para criar um aplicativo Android de exemplo

1. Instale o Android Studio e as ferramentas de linha de comando.
2. No Android Studio, instale o plug-in Flutter.
3. Crie um novo projeto do Android Studio a partir do conteúdo do `cognito_flutter_mobile_app` diretório neste aplicativo de exemplo.
   - Edite `assets/config.json` `<<YOUR USER POOL ID>>` e substitua `<< YOUR CLIENT ID>>` com as IDs do grupo de usuários e do cliente do aplicativo que você criou anteriormente.
4. Instale o Flutter.

a. Adicione Flutter à sua variável PATH.

b. Aceite licenças com o comando a seguir.

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

c. Verifique seu ambiente Flutter e instale os componentes ausentes.

```
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 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 Android virtual.

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

6. Inicie seu dispositivo Android virtual.

   1. Na GUI do Android Studio, selecione o

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

7. Inicie seu aplicativo em seu dispositivo virtual.

   1. Na GUI 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. Inscreva um novo usuário com um endereço de e-mail válido.
10. Recupere o código de confirmação da sua mensagem de e-mail. Insira o código de confirmação no aplicativo.

11. Faça login com seu nome de usuário e senha.

Próximas etapas

Depois de seguir os tutoriais para concluir exemplos de aplicativos, você pode ampliar o escopo da implementação do seu grupo de usuários. Você pode criar grupos de usuários adicionais, personalizar recursos de grupos de usuários para outros aplicativos ou adicionar provedores de identidade externos. Ao planejar sua mudança para colocar grupos de usuários do Amazon Cognito em aplicativos de produção, você pode avaliar exemplos e tutoriais adicionais.

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

- Como personalizar as páginas da Web integradas de cadastro e acesso
- Adicionar MFA a um grupo de usuários
- Como adicionar segurança avançada a um grupo de usuários
- Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda
- Como usar análise do Amazon Pinpoint com grupos de usuários do Amazon Cognito

Para obter uma visão geral dos modelos de autenticação e autorização do Amazon Cognito, consulte. Como a autenticação funciona com grupos de usuários e grupos de identidades do 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 os SDKs AWS Management Console e do grupo de usuários, você também pode gerenciar seus grupos de usuários usando o. AWS Command Line Interface

Tópicos

- Criar um novo grupo de usuários
- Adicionar um cliente de aplicativo com a interface hospedada
- Adicionar um acesso social a um grupo de usuários (opcional)
- Adicionar acesso com um provedor de identidade SAML a um grupo de usuários (opcional)

Criar um novo grupo de usuários

Com um grupo de usuários, seus usuários podem fazer login em aplicações Web ou móveis por meio do Amazon Cognito.

Criar um novo grupo de usuários

1. Acesse o console do Amazon Cognito. Se solicitado, insira suas AWS credenciais.
2. Escolha o botão Criar grupo de usuários. Talvez seja necessário selecionar Grupos de usuários no painel de navegação esquerdo para revelar essa opção.
3. No canto superior direito da página, escolha Create a user pool (Criar um grupo de usuários) para iniciar o assistente de criação de grupo de usuários.
4. Em Configurar a experiência de login, você pode escolher os provedores de identidade (IdPs) que você usará com esse grupo de usuários. Para ter mais informações, consulte Como adicionar acesso a grupo de usuários por meio de terceiros.
    a. Em Provedores de autenticação, para Tipos de provedor, certifique-se de que somente o grupo de usuários do Cognito esteja selecionado.
    b. Para opções de login do grupo de usuários do Cognito, escolha Nome do usuário. Não selecione nenhum requisito adicional de nome de usuário.
    c. Mantenha todas as outras opções como padrão e escolha Avançar.
5. Em Configurar requisitos de segurança, você pode escolher sua política de senha, requisitos de autenticação multifator (MFA) e opções de recuperação de conta de usuário. Para ter mais informações, consulte Usar atributos de segurança de grupos de usuários do Amazon Cognito.
    a. Para Política de senha, confirme se o modo de política de senha está definido para os padrões do Cognito.
    b. Em Autenticação multifator, para imposição de MFA, escolha MFA opcional.
    c. Para métodos de MFA, escolha Aplicativos autenticadores e mensagem SMS.
    d. Em Recuperação de conta de usuário, confirme se a opção Ativar recuperação de conta de autoatendimento está selecionada e se o método de entrega da mensagem de recuperação de conta de usuário está definido como Somente e-mail.
    e. Mantenha todas as outras opções como padrão e escolha Avançar.
6. Em Configurar a experiência de inscrição, você pode determinar como os novos usuários verificarão suas identidades ao se inscreverem como novos usuários e quais atributos devem

ser obrigatórios ou opcionais durante o fluxo de inscrição do usuário. Para ter mais informações, consulte Como gerenciar usuários em seu grupo de usuários.

a. Confirme se a opção Ativar autorregistro está selecionada. Essa configuração abre seu grupo de usuários para que qualquer pessoa se inscreva na Internet. Isso se destina aos propósitos do aplicativo de exemplo, mas aplique essa configuração com cuidado em ambientes de produção.

b. Em Verificação e confirmação assistidas pelo Cognito, verifique se a caixa de seleção Permitir que o Cognito envie mensagens automaticamente para verificação e confirmação está marcada.

c. Confirme se os Atributos a serem verificados estão definidos como Enviar mensagem de e-mail, verificar endereço de e-mail.

d. Em Verificando alterações de atributos, confirme se as opções padrão estão selecionadas: Manter o valor do atributo original quando uma atualização está pendente é selecionado e Valores de atributos ativos quando uma atualização está pendente está definido como Endereço de e-mail.

e. Em Atributos obrigatórios, confirme se os atributos obrigatórios com base nas seleções anteriores exibem e-mail.

> **Important**
>
> Para este aplicativo de exemplo, seu grupo de usuários não deve definir phone_number como um atributo obrigatório. Se phone_number for exibido como um atributo obrigatório, revise e atualize suas escolhas anteriores:
>
> - MFA opcional, e-mail somente para método de entrega para mensagens de recuperação de conta de usuário
> - Envie uma mensagem de e-mail, verifique o endereço de e-mail dos Atributos para verificar

f. Mantenha todas as outras opções como padrão e escolha Avançar.

7. Em Configurar entrega de mensagens, você pode configurar a integração com o Amazon Simple Email Service e o Amazon Simple Notification Service para enviar mensagens de e-mail e SMS aos seus usuários para cadastro, confirmação de conta, MFA e recuperação de conta. 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. Em Provedor de e-mail, escolha Enviar e-mail com o Cognito e use o remetente de e-mail padrão fornecido pelo Amazon Cognito. Essa configuração para baixo volume de e-mails é suficiente para testes de aplicativos. Você pode retornar depois de verificar um endereço de e-mail com o Amazon Simple Email Service (Amazon SES) e escolher Enviar e-mail com o Amazon SES.

b. Para SMS, selecione Criar uma nova função do IAM e insira um nome de função do IAM. Isso cria uma função que concede permissões ao Amazon Cognito para enviar mensagens SMS.

c. Mantenha todas as outras opções como padrão e escolha Avançar.

8. Em Integrar seu aplicativo, você pode nomear seu grupo de usuários, configurar a interface hospedada e criar um cliente de aplicativo. Para ter mais informações, consulte Adicionar um cliente de aplicativo com a interface hospedada. Os aplicativos de exemplo não usam a interface de usuário hospedada.

   a. Em Nome do grupo de usuários, insira um nome do grupo de usuários.

   b. Não selecione Usar a interface hospedada do Cognito.

   c. Em Cliente de aplicativo inicial, confirme se o tipo de aplicativo está definido como Cliente público.

   d. Em Segredo do cliente, confirme se a opção Não gerar uma chave secreta do cliente está selecionada.

   e. Insira um App client name (Nome do cliente da aplicação).

   f. Expanda Configurações avançadas do cliente de aplicativos. Adicione `ALLOW_USER_PASSWORD_AUTH` à lista de fluxos de autenticação.

   g. Mantenha todas as outras opções como padrão e escolha Avançar.

9. Revise suas escolhas na tela Revisar e criar e modifique as seleções conforme necessário. Quando estiver satisfeito com a configuração do grupo de usuários, escolha Criar grupo de usuários para continuar.

10. Na página Grupos de usuários, escolha seu novo grupo de usuários.

11. Em Visão geral do grupo de usuários, anote seu ID do grupo de usuários. Você fornecerá essa string ao criar seu aplicativo de exemplo.

12. Escolha a guia Integração de aplicativos e localize a seção Clientes e análises de aplicativos. Selecione seu novo cliente de aplicativo. Anote seu ID de cliente.

Para criar 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. No canto superior direito da página, escolha Create a user pool (Criar um grupo de usuários) para iniciar o assistente de criação de grupo de usuários.
4. Em Configure sign-in experience (Configurar a experiência de acesso), escolha os provedores federados que você usará com esse grupo de usuários. Para ter mais informações, consulte Como adicionar acesso a grupo de usuários por meio de terceiros.
5. Em Configure security requirements (Configurar requisitos de segurança), escolha sua política de senha, os requisitos de autenticação multifator (MFA) e as opções de recuperação de conta do usuário. Para ter mais informações, consulte Usar atributos de segurança de grupos de usuários do Amazon Cognito.
6. Em Configure sign-up experience (Configurar a experiência de cadastro), determine como os novos usuários verificarão suas identidades durante o cadastro e quais atributos devem ser obrigatórios ou opcionais durante o fluxo de cadastro do usuário. Para ter mais informações, consulte Como gerenciar usuários em seu grupo de usuários.

   > **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 na guia Experiência de inscrição do console do grupo de usuários ou atualize o valor de `AllowAdminCreateUserOnly` em uma `CreateUserPool` solicitação 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.

7. Em Configure message delivery (Configurar entrega de mensagem), configure a integração com o Amazon Simple Email Service e o Amazon Simple Notification Service para enviar mensagens de e-mail e SMS a seus usuários para cadastro, confirmação de conta, MFA e recuperação de conta. 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.

8. Em Integrate your app (Integrar sua aplicação), nomeie seu grupo de usuários, configure a interface do usuário hospedada e crie um cliente da aplicação. Para ter mais informações, consulte Adicionar um cliente de aplicativo com a interface hospedada.
9. Revise suas escolhas na tela Revisar e criar e modifique as seleções conforme necessário. Quando estiver satisfeito com a configuração do grupo de usuários, selecione Criar grupo de usuários para continuar.

Recursos relacionados

Para obter mais informações sobre grupos de usuários, consulte Grupos de usuários do Amazon Cognito.

Veja também: Fluxo de autenticação de grupo de usuários Como usar tokens com grupos de usuários e.

Adicionar um cliente de aplicativo com a interface hospedada

Depois de criar um grupo de usuários, você pode criar um cliente de aplicativo para um aplicativo que exiba as páginas da Web integradas da interface do usuário hospedada. Na interface de usuário hospedada, os usuários podem:

- Inscreva-se em um perfil de usuário.
- Faça login com um provedor de identidade terceirizado.
- Faça login com ou sem autenticação multifator.
- Redefina sua senha.

Para criar um cliente de aplicativo para interface de usuário hospedada, faça login

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. Se criar um novo grupo de usuários, você será solicitado a configurar um cliente de aplicação e a interface do usuário hospedada durante o assistente.

4. Escolha a guia App integration (Integração de aplicação) para o seu grupo de usuários.
5. Ao lado de Domínio, escolha Ações e Criar domínio personalizado ou Criar domínio do Amazon Cognito. Se já tiver configurado um domínio de grupo de usuários, escolha Excluir domínio do Amazon Cognito ou Excluir domínio personalizado antes de criar o domínio personalizado.
6. Insira um prefixo de domínio disponível para usar com um Domínio do Amazon Cognito. Para informações sobre como configurar um Domínio personalizado, consulte Uso do próprio domínio para a interface do usuário hospedada
7. Escolha Criar.
8. Retorne para a guia App integration (Integração da aplicação) para o mesmo grupo de usuários e localize App clients (Clientes da aplicação). Escolha Create an app client (Criar um cliente da aplicação).
9. Escolha um Application type (Tipo de aplicação). Algumas configurações recomendadas serão fornecidas com base na sua seleção. Uma aplicação que usa a interface do usuário hospedada é um Public client (Cliente público).
10. Insira um App client name (Nome do cliente da aplicação).
11. Para este exercício, escolha Don't generate client secret (Não gerar segredo do cliente). O segredo do cliente é usado por aplicações confidenciais que autenticam usuários de uma aplicação centralizada. Neste exercício, você apresentará uma página de acesso da interface do usuário hospedada a seus usuários e não exigirá um segredo do cliente.
12. Escolha os fluxos de autenticação que você permitirá com seu aplicativo. Certifique-se de que `USER_SRP_AUTH` tenha sido selecionado.
13. Personalize token expiration (validade do token), Advanced security configuration (Configuração avançada de segurança) e Attribute read and write permissions (Permissões de leitura e gravação do atributo) conforme necessário. Para mais informações, consulte Configuring App Client Settings (Definir configurações do cliente da aplicação).
14. Add a callback URL (Adicionar um URL de retorno de chamada) para seu cliente da aplicação. Você será direcionado para essa página após a autenticação da interface do usuário hospedada. Você não precisa adicionar uma URL de saída permitida até conseguir implementar a saída em seu aplicativo.

    Para um aplicativo de iOS ou Android, é possível usar um URL de retorno de chamada como o `myapp://`.
15. Selecione os Identity providers (Provedores de identidade) para o cliente da aplicação. No mínimo, habilite o Grupo de usuários do Amazon Cognito como provedor.

> **Note**
>
> Para fazer login com provedores de identidade externos (IdPs), como Facebook, Amazon, Google e Apple, bem como por meio do OpenID Connect (OIDC) ou SAML IdPs, primeiro configure-os conforme mostrado em Adicionar login ao grupo de usuários por meio de terceiros. Em seguida, retorne à página de configurações do cliente do aplicativo para ativá-las.

16. Escolha OAuth 2.0 Grant Types (Tipos de concessão OAuth 2.0.). Selecione Authorization code grant (Concessão de código de autorização) para retornar um código de autorização que é trocado por tokens do grupo de usuários. Como os tokens nunca são expostos diretamente a um usuário final, é menos provável que eles fiquem comprometidos. No entanto, uma aplicação personalizada é necessário no backend para trocar o código de autorização por tokens do grupo de usuários. Por motivos de segurança, recomendamos que você use o fluxo de concessão de código de autorização, junto com o Proof Key for Code Exchange (PKCE), para aplicativos móveis.

    Selecione Implicit grant (Concessão implícita) para que os JSON Web Tokens (JWT) do grupo de usuários sejam retornados do Amazon Cognito. Você pode usar esse fluxo quando não houver backend disponível para trocar um código de autorização por tokens. Ele também é útil para depurar tokens.

    > **Note**
    >
    > Você pode habilitar Authorization code grant (Concessão de código de autorização) e Implicit code grant (Concessão de código implícita) e, em seguida, usar cada concessão conforme necessário.
    > Selecione Client credentials somente se o aplicativo precisa solicitar tokens de acesso para ele mesmo, e não em nome de um usuário.

17. A menos que deseje excluir especificamente uma opção, selecione todos os OpenID Connect scopes (Escopos do OpenID Connect).
18. Selecione qualquer escopo personalizado que você tenha configurado. Normalmente os escopos personalizados são usados com clientes confidenciais.
19. Escolha Criar.

Para visualizar sua página de login

Na página do cliente do aplicativo, selecione Exibir interface hospedada para abrir uma nova guia do navegador em uma página de login pré-preenchida com os parâmetros de ID do cliente do aplicativo, escopo, concessão e URL de retorno de chamada.

Você pode exibir manualmente a página da Web de acesso da interface do usuário hospedada com o seguinte URL. Anote o `response_type`. Neste caso, response_type=code para a concessão de código de autorização.

```
https://your_domain/login?
response_type=code&client_id=your_app_client_id&redirect_uri=your_callback_url
```

Você pode visualizar a página da web de login da interface do usuário hospedada 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://your_domain/login?
response_type=token&client_id=your_app_client_id&redirect_uri=your_callback_url
```

Você pode encontrar o token que identidade do JSON web token (JWT) logo depois do parâmetro `#idtoken=` na resposta.

O URL a seguir é um exemplo de resposta de uma solicitação de concessão implícita. Sua string de token de identidade será muito maior.

```
https://www.example.com/
#id_token=123456789tokens123456789&expires_in=3600&token_type=Bearer
```

Os tokens de grupos de usuários do Amazon Cognito são assinados usando um algoritmo RS256. Você pode decodificar e verificar os tokens do grupo de usuários usando AWS Lambda. Para obter mais informações, consulte Decodificar e verificar os tokens JWT do Amazon Cognito no site. AWS GitHub

O domínio é exibido na página Domain name (Nome do domínio). O ID de cliente do aplicativo e o URL de retorno de chamada são exibidos na página General settings (Configurações gerais). Se as

alterações feitas no console não aparecerem imediatamente, aguarde alguns minutos e atualize o navegador.

Adicionar um acesso social a um grupo de usuários (opcional)

Você pode permitir que os usuários do aplicativo façam login por meio de um provedor de identidade social (IdP), como o Facebook, o Google, a Amazon e a Apple. 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 de login social.

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 um aplicativo do Facebook existente, 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 suas plataformas e escolha Avançar.
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, escolha Produtos e, em seguida, escolha Configurar a partir do login do Facebook.
14. No menu Configurar de Login com Facebook, selecione Configurações.

    Insira o URL de redirecionamento em Valid OAuth Redirect URIs (URIs válidos de redirecionamento OAuth). O URL de redirecionamento consiste no domínio do grupo de usuários com o `/oauth2/idpresponse` endpoint.

    ```
    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.

   Escolha Aplicativos e serviços na barra de navegação na parte superior da página e, em seguida, escolha 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 o domínio do grupo de usuários com o endpoint `/oauth2/idpresponse` em Allowed Return URLs (URLs permitidos de retorno).

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

11. Escolha Salvar.

Para registrar um aplicativo com o Google

Para obter mais informações sobre o OAuth 2.0 na plataforma Google Cloud, consulte Learn about authentication & authorization (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 a tela de consentimento do Oauth.
7. Insira as informações do aplicativo, um domínio do aplicativo, domínios autorizados e informações de contato do desenvolvedor. Seus domínios autorizados devem incluir `amazoncognito.com` e a raiz do seu domínio personalizado. Por exemplo: `example.com`. Escolha SAVE AND CONTINUE (Salvar e continuar).
8. 1. Em Escopos, escolha Adicionar ou remover escopos e, em seguida, escolha, no mínimo, os seguintes escopos do OAuth.

    1. `.../auth/userinfo.email`
    2. `.../auth/userinfo.profile`
    3. OpenID
9. Em Test users (Testar usuários), escolha Add Users (Adicionar usuários). Insira seu endereço de e-mail e quaisquer outros usuários de teste autorizados e, em seguida, escolha SALVAR E CONTINUAR.
10. Expanda a barra de navegação esquerda novamente, escolha APIs e serviços e, em seguida, escolha Credenciais.
11. Escolha CRIAR CREDENCIAIS e, em seguida, escolha ID do cliente OAuth.

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 Authorized redirect URIs (URIs de redirecionamento autorizadas), escolha ADD URI (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 Register a New Identifier (Registrar um novo identificador), escolha App IDs (IDs de aplicação) e selecione Continue (Continuar).
7. Na página Selecionar um tipo, escolha Aplicativo e, em seguida, escolha 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 Etapa 2: adicionar um IdP social ao seu grupo de usuários.

3. Em Capabilities (Recursos), escolha Sign In with Apple (Fazer login com a Apple) e, depois, selecione Edit (Editar).
4. Na página Sign in with Apple: App ID Configuration (Fazer login com a Apple: configuração do ID da aplicação), escolha configurar a aplicação como principal ou agrupada com outros IDs de aplicação e, depois, escolha Save (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 Register a New Identifier (Registrar um novo identificador), escolha Services IDs (IDs de serviços) e selecione Continue (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 essa ID de serviços porque você precisará desse valor depois de escolher a Apple como seu provedor de identidade emEtapa 2: adicionar um IdP social ao seu grupo de usuários.
    3. Escolha Continuar e, depois, Registrar.
13. Escolha a 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 de Website URLs (URLs de site).
    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 (URLs de retorno), insira o caminho para o endpoint `/oauth2/idpresponse` do domínio de seu grupo de usuários.

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

6. Escolha Avançar e, em seguida, escolha 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 a ID do aplicativo que você criou anteriormente como a ID principal do aplicativo. Escolha Salvar.
    4. Escolha Continue (Continuar) e, depois, Register (Registrar).
17. Na página Baixar sua chave, escolha Baixar para baixar a chave privada, anote a ID da chave exibida e 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 Etapa 2: adicionar um IdP social ao seu grupo de usuários.

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 AWS Management 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 na lista ou crie um grupo de usuários.
4. Escolha a guia Sign-in experience (Experiência de acesso). 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 do aplicativo e o segredo do cliente do aplicativo que foram gerados na seção anterior.
- Facebook — insira o ID do cliente do aplicativo e o segredo do cliente do aplicativo que foram gerados na seção anterior e escolha uma versão da API (por exemplo, versão 2.12). Recomendamos escolher a versão mais recente possível. Cada API do Facebook tem um ciclo de vida e uma data de suspensão de uso. 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.
- Faça login com a Apple — Insira o ID de serviços, ID da equipe, ID da chave e chave privada que foram gerados na seção anterior.

7. Insira os nomes dos escopos autorizados que você deseja usar. 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 with 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 após sair da página de login da Apple (elas podem ser causadas por falhas internas no Amazon Cognito ou por qualquer coisa escrita pelo desenvolvedor).

- O identificador de ID do serviço é usado em grupos de usuários e/ou 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, em seguida, o usuário faz login novamente sem remover o aplicativo do perfil de ID Apple.

8. Mapeie atributos do provedor de identidade para o grupo de usuários. Para ter mais informações, consulte Coisas a saber sobre mapeamentos.
9. Escolha Criar.
10. Na guia App client integration (Integração de cliente da aplicação), escolha um dos App clients (Clientes da aplicação) na lista e escolha Edit hosted UI settings (Editar configurações da 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.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 a interface do usuário hospedada. A página exibe a mensagem de erro `Something went wrong` (Ocorreu algum problema).

Adicionar acesso com um provedor de identidade SAML a um grupo de usuários (opcional)

Você pode permitir que os usuários do aplicativo façam login por meio de um provedor de identidade (IdP) SAML. 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 ter mais informações, consulte Usando provedores de identidade SAML com um grupo de usuários.

Você deve atualizar seu provedor de identidade SAML e configurar seu grupo de usuários. Para obter informações sobre como adicionar seu grupo de usuários como parte confiável ou aplicativo para seu provedor de identidade SAML 2.0, consulte a documentação do seu provedor de identidade SAML.

Você também deve fornecer um endpoint do Assertion Consumer Service (ACS) ao seu provedor de identidade SAML. Configure o endpoint a seguir no domínio do grupo de usuários para a vinculação POST SAML 2.0 no provedor de identidades SAML. Para obter mais informações sobre domínios de grupos de usuários, consulteComo configurar um domínio de 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 seu domínio e o valor da região para seu grupo de usuários na guia Nome de domínio do console do Amazon Cognito.

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

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

É possível localizar o ID do grupo de usuários na guia General settings (Configurações gerais) 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 seu aplicativo recuperar ou analisar respostas de asserção SAML porque o grupo de usuários recebe diretamente a resposta SAML do seu provedor de identidade 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. Escolha a guia Sign-in experience (Experiência de acesso). 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 verificar o endereço de e-mail que o usuário insere ao fazer login. Em seguida, ele os 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 identidade SAML 2.0 para enviar respostas de desconexão para o endpoint `https://<your Amazon Cognito domain>/saml2/logout` que é criado quando você configura a interface do usuário hospedada. O `saml2/logout` endpoint usa a vinculação POST.

   > **Note**
   >
   > Se essa opção for selecionada 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 SAML IdP processará a solicitação de logout assinada e desconectará 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**
>
> Recomendamos que você insira uma URL de documento de metadados se seu provedor tiver um endpoint público, em vez de fazer o upload de 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 Create (Criar).

Introdução aos grupos de identidade do Amazon Cognito

Com os grupos de identidades do Amazon Cognito, você pode criar identidades exclusivas e atribuir permissões aos usuários. O grupo de identidades pode incluir:

- 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

Com um pool de identidades, você pode obter AWS credenciais temporárias com permissões que você define para acessar diretamente outros recursos Serviços da AWS ou para acessar recursos por meio do Amazon API Gateway.

Tópicos

Criar um grupo de identidades no Amazon Cognito

Você pode criar rapidamente um grupo de identidades pelo console do Amazon Cognito ou usar a AWS Command Line Interface (CLI) ou as APIs do Amazon Cognito.

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 ter 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 receber a solicitação para fornecer informações do cliente da aplicação OAuth, selecionar um grupo de usuários do Amazon Cognito, escolher um IdP do IAM ou inserir um identificador personalizado para um provedor de desenvolvedor.

   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 você deseja comparar a declaração, 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. 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 ter mais informações, consulte Fluxo de autenticação dos grupos de identidades (identidades federadas).
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 for Java ou o. AWS SDK for .NET Para obter mais informações, consulte os tópicos a seguir.

- Configurando o SDK para o JavaScript Guia do AWS SDK for Java Desenvolvedor
- Documentação do Amplify no Amplify Dev Center
- Provedor de credenciais do Amazon Cognito no Guia do desenvolvedor do AWS SDK for .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 (bancos de identidades), 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 Provedores externos de identidade de grupos de identidades.

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

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.

Opções de configuração guiada para o Amazon Cognito

Talvez você queira 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 grupos de identidades.

Conclua um workshop

AWS O workshop studio organiza um workshop que orienta você na 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 grupos de identidades e a configuração de segurança.

Adicione o código do aplicativo a partir de exemplos

O capítulo de exemplos de código deste guia tem código de aplicativo que você pode usar com grupos de usuários e grupos de identidades. A seção de grupos de usuários do capítulo de exemplos de código tem trechos curtos que abrangem operações individuais e exemplos mais longos, por end-to-end exemplo, aplicativos em uma variedade de linguagens de programação.

Crie um aplicativo fullstack com AWS Amplify

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

Mais recursos do aplicativo Amazon Cognito em GitHub

- Exemplos de fluxo de autenticação com o.NET para Amazon Cognito
- Autenticação sem senha do Amazon Cognito
- PetStoreexemplo com Amazon Verified Permissions
- Exemplo de aplicativo React usando grupos de identidade ABAC + para acessar recursos AWS
- Autorização máquina a máquina baseada no Amazon Cognito e no API Gateway usando CDK AWS
- Criação de autorização refinada usando o Amazon Cognito, o API Gateway e o IAM

- CloudFrontautorização @edge

Mais workshops

- Implemente a autenticação sem senha com o Amazon Cognito e WebAuthn
- Identidade SaaS multilocatário com grupos de usuários do Amazon Cognito
- Análise aprofundada do Amazon Cognito JWT

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

Ao integrar a aplicação a um cliente de aplicação do Amazon Cognito, é possível invocar operações de API para autenticação e autorização de usuários. Recomendamos que você use AWS Amplifypara integrar o Amazon Cognito com seus aplicativos web e móveis. AWS Amplify é uma solução completa que permite que desenvolvedores front-end web e móveis criem, conectem e hospedem facilmente aplicativos fullstack AWS, com a flexibilidade de aproveitar a amplitude da evolução de seus casos de Serviços da AWS uso. O Amplify Auth usa principalmente o Amazon Cognito para criar atributos de autenticação.

Tópicos

- Autenticação com AWS Amplify
- Autenticação com AWS SDKs
- Autorização com o Amazon Verified Permissions

Uma implementação típica do Amazon Cognito usa uma combinação de ferramentas visuais e APIs. 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 interface de usuário hospedada é um aplicativo de login ready-to-use baseado na web para testes e implantação rápidos de grupos de usuários do Amazon Cognito. Além disso, na maioria das implantações do Amazon Cognito, você deve adicionar código nas aplicações para interagir com os grupos de usuários e bancos de identidades. Por exemplo, a aplicação pode invocar a interface de usuário hospedada 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. Para começar a usar seu próprio código de aplicação personalizado, acesse 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. Consulte as plataformas disponíveis

em Autenticação 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 a API do Amazon Cognito.

> **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 ter mais informações, consulte Usar a API de grupos de usuários e endpoints de grupo de usuários do Amazon Cognito.

Autenticação com AWS Amplify

AWS Amplify é uma solução completa para a criação de 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.

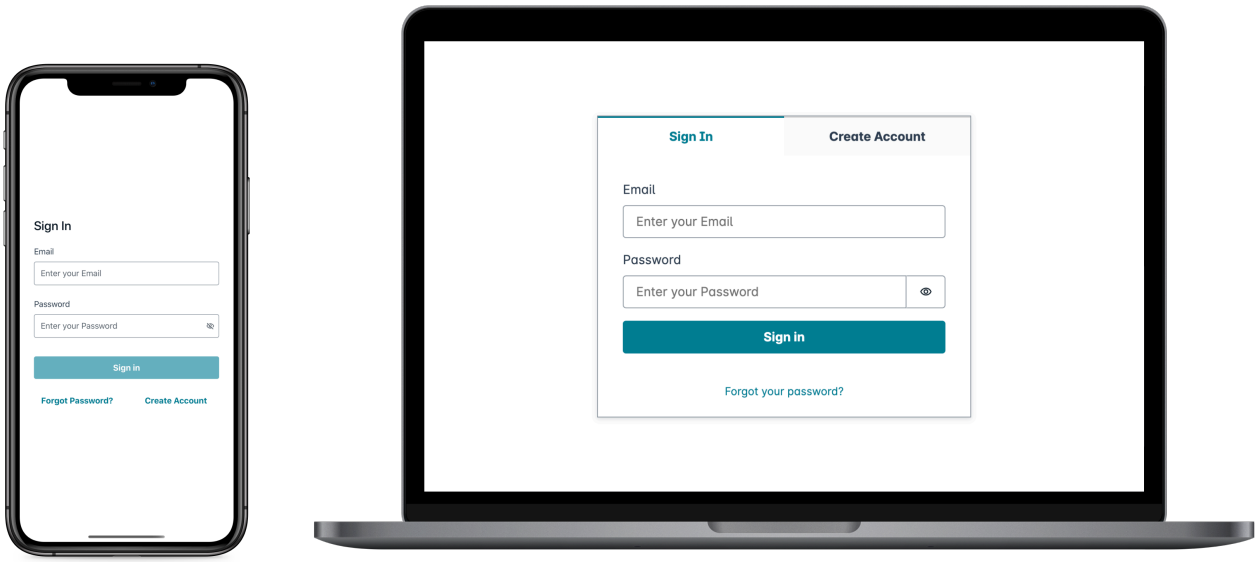
- Amplifique a autenticação para JavaScript
- Autenticação do Amplify para iOS
- Autenticação do Amplify para Android
- Autenticação do Amplify para Flutter

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

A Interface de usuário hospedada de grupos de usuários do Amazon Cognito pode atender às necessidades essenciais de um front-end de autenticação para uma aplicação web ou móvel. Para personalizar a interface de usuário (UI) além dos parâmetros que a interface de usuário hospedada 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 seções Consulte também com recursos para usar uma variedade de SDKs em plataformas compatíveis.

- 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

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 usuários `Principal` e tomam decisões de `Action` autorização para aqueles `Resource` que eles desejam acessar. Personalizações adicionais `Context` podem 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 `token_use` reivindicação em seu token corresponde aos parâmetros para os quais você passou`IsAuthorizedWithToken`. A `token_use` afirmação deve ser `access` se você a passou para o `accessToken` parâmetro e `id` se você a passou para o `identityToken` parâmetro.
5. A assinatura no token vem das chaves web JSON (JWKs) publicadas do grupo de usuários. É possível ver as 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. As permissões verificadas respondem à sua solicitação dizendo `Allow` se ela corresponde a uma ação permitida e não corresponde a uma `Deny` ação explícita; caso contrário, ela 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 declarações de usuário que você deseja apresentar às Permissões verificadas, personalize o conteúdo em seus tokens de acesso e identidade com umAcionador 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.

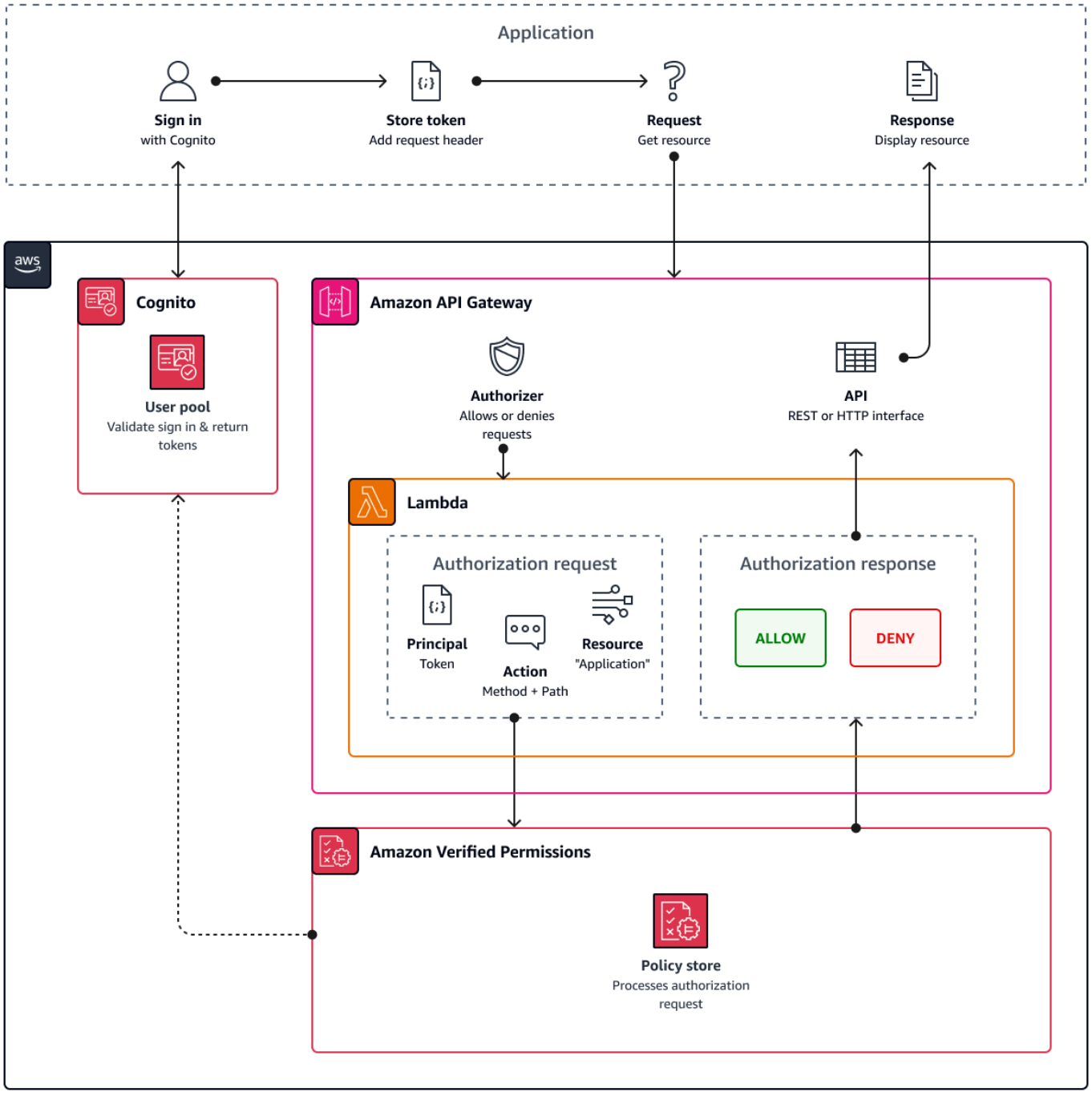
Para obter mais informações sobre como o Verified Permissions associa as declarações nos tokens do Amazon Cognito às políticas de autorização, consulte Mapping Amazon Cognito tokens to Verified Permissions schema.

Autorização de API com permissões verificadas

Seu ID ou tokens de acesso podem autorizar solicitações para as APIs REST de back-end do Amazon API Gateway com permissões verificadas. Você pode criar um repositório de políticas com links imediatos para seu grupo de usuários e sua API. Com a opção inicial Configurar com o Cognito e o API Gateway, as Permissões Verificadas adicionam 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 Permissões

verificadas. O autorizador passa o token como principal e o caminho e o método da solicitação como uma ação.

O diagrama a seguir ilustra o fluxo de autorização de uma API do API Gateway com permissões verificadas. Para obter uma análise detalhada, consulte repositórios de políticas vinculados à API no Guia do usuário de permissões verificadas da Amazon.



As permissões verificadas estruturam a autorização da API em torno de grupos 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 funções (RBAC) para suas APIs em vários contextos de aplicativos.

Escolhendo as configurações do repositório de políticas

Ao configurar uma fonte de identidade em um repositório de políticas, você deve escolher se deseja processar o acesso ou os tokens de 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: escopos do OAuth. 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 de permissões verificadas. O token de acesso aumenta a associação ao grupo com escopos que podem contribuir para a decisão de autorização. As reivindicações em um token de acesso se tornam contextuais na solicitação de autorização.

Você também deve configurar os tipos de entidade de usuário e grupo ao configurar um grupo de usuários como fonte de identidade. Os tipos de entidade são identificadores principais, de ação e de recursos que você pode consultar nas políticas de permissões verificadas. As entidades nos repositórios de políticas podem ter uma relação de associação, em que uma entidade pode ser membro de uma entidade controladora. Com a associação, você pode referenciar grupos principais, grupos de ação e grupos de recursos. No caso de grupos 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 de permissões verificadas, seu repositório de políticas tem automaticamente essa relação pai-membro.

O token de ID pode combinar o RBAC com o controle de acesso baseado em atributos (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 principais na solicitação de autorização. Suas políticas podem tomar decisões de autorização com base nos atributos principais.

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

Exemplo de política para autorização de API baseada em funções

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
  );
```

As permissões verificadas retornam uma `Allow` decisão à solicitação de autorização do seu aplicativo quando:

1. Seu aplicativo passou um ID ou token de acesso em um `Authorization` cabeçalho como um token portador.
2. Seu aplicativo passou um token com uma `cognito:groups` declaração que contém a string`MyGroup`.
3. Seu aplicativo fez uma `HTTP GET` solicitação 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 Permissões Verificadas em condições diferentes das solicitações de API. Você pode enviar qualquer decisão de controle de acesso em seu aplicativo 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
);
```

O seguinte é um exemplo 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 quiser especificar um grupo de usuários em uma política de permissões verificadas, use o seguinte formato:

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

A seguir está um exemplo

Controle de acesso baseado em atributos

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	Processar	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 obter uma lista completa dos Guias do desenvolvedor do AWS SDK e exemplos de código, 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 SDKs AWS
  - Ações para o Amazon Cognito Identity usando SDKs AWS
    - Use CreateIdentityPool com um AWS SDK ou CLI
    - Use DeleteIdentityPool com um AWS SDK ou CLI
    - Use DescribeIdentityPool com um AWS SDK ou CLI
    - Use GetCredentialsForIdentity com um AWS SDK ou CLI
    - Use GetIdentityPoolRoles com um AWS SDK ou CLI
    - Use ListIdentityPools com um AWS SDK ou CLI
    - Use SetIdentityPoolRoles com um AWS SDK ou CLI
    - Use UpdateIdentityPool com um AWS SDK ou CLI
  - Exemplos de serviços cruzados para o Amazon Cognito Identity usando SDKs AWS
    - Criar uma aplicação Amazon Transcribe
    - Criar uma aplicação de exploração do Amazon Textract
- Exemplos de código para o Amazon Cognito Identity Provider usando SDKs AWS
  - Ações para o Amazon Cognito Identity Provider usando SDKs AWS
    - 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

Exemplos de código para o Amazon Cognito Identity usando SDKs AWS

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 funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Exemplos entre serviços são amostras de aplicações que funcionam em vários 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.

Exemplos de código

- Ações para o Amazon Cognito Identity usando SDKs AWS
  - Use CreateIdentityPool com um AWS SDK ou CLI
  - Use DeleteIdentityPool com um AWS SDK ou CLI
  - Use DescribeIdentityPool com um AWS SDK ou CLI
  - Use GetCredentialsForIdentity com um AWS SDK ou CLI
  - Use GetIdentityPoolRoles com um AWS SDK ou CLI
  - Use ListIdentityPools com um AWS SDK ou CLI
  - Use SetIdentityPoolRoles com um AWS SDK ou CLI
  - Use UpdateIdentityPool com um AWS SDK ou CLI
- Exemplos de serviços cruzados para o Amazon Cognito Identity usando SDKs AWS
  - Criar uma aplicação Amazon Transcribe
  - Criar uma aplicação de exploração do Amazon Textract

Ações para o Amazon Cognito Identity usando SDKs AWS

Os exemplos de código a seguir demonstram como realizar ações individuais do Amazon Cognito Identity com AWS SDKs. Esses trechos chamam a API de identidade do Amazon Cognito e são

trechos de código de programas maiores que devem ser executados no contexto. 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 Identity.

Exemplos

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

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

Os exemplos de códigos a seguir mostram como usar `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 veja 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

Exemplo 1: Cria um novo grupo 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 AWS Tools for PowerShell cmdlet.

Swift

SDK para Swift

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

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

Crie um banco de identidades.

```
    /// 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? {
        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
    }
```

- 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 códigos a seguir mostram como usar `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 veja 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

Exemplo 1: Exclui um grupo de identidades específico.

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

- Para obter detalhes da API, consulte DeleteIdentityPoolem Referência de AWS Tools for PowerShell cmdlet.

Swift

SDK para Swift

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

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

Exclua o banco de identidades especificado.

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

    _ = try await cognitoIdentityClient.deleteIdentityPool(input: input)
}
```

- 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.

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

Os exemplos de códigos a seguir mostram como usar `DescribeIdentityPool`.

CLI

AWS CLI

Para descrever um pool de identidades

Este exemplo descreve um grupo 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

Exemplo 1: recupera informações sobre um grupo 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 AWS Tools for PowerShell cmdlet.

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 ou CLI

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 veja 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.

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

Os exemplos de códigos a seguir mostram como usar `GetIdentityPoolRoles`.

CLI

AWS CLI

Para obter funções no pool de identidades

Este exemplo obtém funções do grupo 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

Exemplo 1: Obtém as informações sobre as funções de um grupo 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 AWS Tools for PowerShell cmdlet.

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 códigos a seguir mostram como usar `ListIdentityPools`.

CLI

AWS CLI

Como 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 veja 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

Exemplo 1: Recupera uma lista de grupos 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 AWS Tools for PowerShell cmdlet.

Swift

SDK para Swift

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

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

Encontre o ID de um banco de identidades com seu nome.

```
    /// 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? {
        var token: String? = nil

        // Iterate over the identity pools until a match is found.

        repeat {
            /// `token` is a value returned by `ListIdentityPools()` if the
            /// returned list of identity pools is only a partial list. You
            /// use the `token` to tell Amazon Cognito that you want to
            /// continue where you left off previously. If you specify `nil`
            /// or you don't provide the token, Amazon Cognito will start at
            /// the beginning.

            let listPoolsInput = ListIdentityPoolsInput(maxResults: 25,
nextToken: token)
```

```
            /// 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. Each time we ask for the next
            /// page of identity pools, we pass in the token given by the
            /// previous page.

            let output = try await cognitoIdentityClient.listIdentityPools(input:
listPoolsInput)

            if let identityPools = output.identityPools {
                for pool in identityPools {
                    if pool.identityPoolName == name {
                        return pool.identityPoolId!
                    }
                }
            }

            token = output.nextToken
        } while token != nil

        return nil
    }
```

Obtenha o ID de um banco de identidades existente ou crie-o se ainda não existir.

```
    /// 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.

        guard let poolId = try await self.getIdentityPoolID(name: name) else {
            return try await self.createIdentityPool(name: name)
        }

        return poolId
    }
```

- 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.

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

Os exemplos de códigos a seguir mostram como usar `SetIdentityPoolRoles`.

CLI

AWS CLI

Para definir funções do grupo de identidades

O `set-identity-pool-roles` exemplo a seguir define uma função do grupo 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

Exemplo 1: configura o grupo de identidades específico para ter uma função do IAM não autenticada.

```
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 AWS Tools for PowerShell cmdlet.

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 **`UpdateIdentityPool`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `UpdateIdentityPool`.

CLI

AWS CLI

Para atualizar um grupo de identidades

Este exemplo atualiza um grupo de identidades. Ele define o nome como MyIdentityPool. Ele adiciona o Cognito como provedor de identidade. 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

Exemplo 1: Atualiza algumas das propriedades do Identity Pool, neste caso, o nome do Identity Pool.

```
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 AWS Tools for PowerShell cmdlet.

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 serviços cruzados para o Amazon Cognito Identity usando SDKs AWS

Os exemplos de aplicativos a seguir usam AWS SDKs para combinar o Amazon Cognito Identity com outros. Serviços da AWS Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o aplicativo.

Exemplos

- Criar uma aplicação Amazon Transcribe
- Criar uma aplicação de exploração do Amazon Textract

Criar uma aplicação Amazon Transcribe

O exemplo de código a seguir mostra como usar o Amazon Transcribe para transcrever e exibir gravações de voz no navegador.

JavaScript

SDK para JavaScript (v3)

Crie uma aplicação que use o Amazon Transcribe para transcrever e exibir gravações de voz no navegador. A aplicação usa dois buckets do Amazon Simple Storage Service (Amazon S3), um para hospedar o código da aplicação e outro para armazenar transcrições. A aplicação usa um grupo de usuários do Amazon Cognito para autenticar seus usuários. Os usuários autenticados têm permissões AWS Identity and Access Management (IAM) para acessar os AWS serviços necessários.

Para obter o código-fonte completo e instruções sobre como configurar e executar, veja o exemplo completo em GitHub.

Esse exemplo também está disponível no Guia do desenvolvedor do AWS SDK for JavaScript v3.

Serviços utilizados neste exemplo

- Identidade do Amazon Cognito
- Amazon S3
- Amazon Transcribe

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.

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 for 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 utilizados 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 SDKs AWS

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 funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Para 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.

Conceitos básicos

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 veja como configurar e executar no AWS Code Examples Repository.

Código para o arquivo CMake C MakeLists .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 for C++ da API.

Go

SDK para Go V2

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 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(context.TODO())
  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 for Go da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 veja 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 for JavaScript da API.

Exemplos de código

- Ações para o Amazon Cognito Identity Provider usando SDKs AWS
  - 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 SDKs AWS
  - 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 exige MFA usando um SDK AWS
  - Grave dados de atividades personalizados com uma função Lambda após a autenticação do usuário do Amazon Cognito usando um SDK AWS

Ações para o Amazon Cognito Identity Provider usando SDKs AWS

Os exemplos de código a seguir demonstram como realizar ações individuais do Amazon Cognito Identity Provider com AWS SDKs. 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. 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 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 códigos a seguir mostram como usar `AdminCreateUser`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- 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 veja como configurar e executar no AWS Code Examples Repository.

```
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(userPoolId string, userName string,
 userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
 &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 for 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 códigos a seguir mostram como usar `AdminGetUser`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for 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 veja 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 veja 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 for JavaScript da API.

Kotlin

SDK para Kotlin

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

```
suspend fun getAdminUser(userNameVal: String?, poolIdVal: String?) {
    val userRequest = AdminGetUserRequest {
        username = userNameVal
        userPoolId = poolIdVal
    }

    CognitoIdentityProviderClient { 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 veja 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).

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 códigos a seguir mostram como usar `AdminInitiateAuth`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for C++ da API.

CLI

AWS CLI

Como iniciar a autorização

Este exemplo inicia a autorização usando o fluxo ADMIN_NO_SRP_AUTH para o nome de usuário jane@example.com

O cliente deve ter a API de login para autenticação baseada em servidor (ADMIN_NO_SRP_AUTH) habilitada.

Use as informações da sessão no valor de retorno para chamar admin-respond-to-auth - challenge.

Comando:

```
aws cognito-idp admin-initiate-auth --user-pool-id us-west-2_aaaaaaaaa --client-
id 3n4b5urk1ft4fl3mg5e62d9ado --auth-flow ADMIN_NO_SRP_AUTH --auth-parameters
 USERNAME=jane@example.com,PASSWORD=password
```

Saída:

```
{
  "ChallengeName": "NEW_PASSWORD_REQUIRED",
  "Session": "SESSION",
  "ChallengeParameters": {
      "USER_ID_FOR_SRP": "84514837-dcbc-4af1-abff-f3c109334894",
      "requiredAttributes": "[]",
      "userAttributes": "{\"email_verified\":\"true\",\"phone_number_verified\":
\"true\",\"phone_number\":\"+01xxx5550100\",\"email\":\"jane@example.com\"}"
  }
}
```

- 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 veja 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 veja 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 for JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 { 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 veja 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).

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 códigos a seguir mostram como usar `AdminRespondToAuthChallenge`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for C++ da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 veja 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 for JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 { 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 veja 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).

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

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

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Grave dados de atividades personalizados com uma função do Lambda após a autenticação do usuário do Amazon Cognito

Go

SDK para Go V2

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

```
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(userPoolId string, userName
 string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
 &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 for 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 códigos a seguir mostram como usar `AssociateSoftwareToken`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for C ++ da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 veja 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 for JavaScript da API.

Kotlin

SDK para Kotlin

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

```
suspend fun getSecretForAppMFA(sessionVal: String?): String? {
    val softwareTokenRequest = AssociateSoftwareTokenRequest {
        session = sessionVal
    }

    CognitoIdentityProviderClient { 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 veja 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).

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 códigos a seguir mostram como usar `ConfirmDevice`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for JavaScript da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 códigos a seguir mostram como usar `ConfirmForgotPassword`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Migre automaticamente usuários conhecidos com uma função do Lambda

CLI

AWS CLI

Para confirmar uma senha esquecida

Este exemplo confirma uma senha esquecida 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 veja como configurar e executar no AWS Code Examples Repository.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// ConfirmForgotPassword confirms a user with a confirmation code and a new
 password.
func (actor CognitoActions) ConfirmForgotPassword(clientId string, code string,
 userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
 &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 for 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 códigos a seguir mostram como usar `ConfirmSignUp`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for 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 veja 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 veja 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 for JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 { 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 veja 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).

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 códigos a seguir mostram como usar `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 veja 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 códigos a seguir mostram como usar `CreateUserPoolClient`.

CLI

AWS CLI

Para criar um cliente de grupo de usuários

Este exemplo cria um novo cliente de grupo de usuários com dois fluxos de autorização explícitos: USER_PASSWORD_AUTH e ADMIN_NO_SRP_AUTH.

Comando:

```
aws cognito-idp create-user-pool-client --user-pool-id us-west-2_aaaaaaaaa
  --client-name MyNewClient --no-generate-secret --explicit-auth-flows
 "USER_PASSWORD_AUTH" "ADMIN_NO_SRP_AUTH"
```

Saída:

```
{
  "UserPoolClient": {
      "UserPoolId": "us-west-2_aaaaaaaaa",
      "ClientName": "MyNewClient",
      "ClientId": "6p3bs000no6a4ue1idruvd05ad",
      "LastModifiedDate": 1548697449.497,
      "CreationDate": 1548697449.497,
      "RefreshTokenValidity": 30,
      "ExplicitAuthFlows": [
          "USER_PASSWORD_AUTH",
          "ADMIN_NO_SRP_AUTH"
      ],
      "AllowedOAuthFlowsUserPoolClient": false
  }
}
```

- 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 veja 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 códigos a seguir mostram como usar `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 veja 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 for 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 veja como configurar e executar no AWS Code Examples Repository.

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// DeleteUser removes a user from the user pool.
func (actor CognitoActions) DeleteUser(userAccessToken string) error {
 _, err := actor.CognitoClient.DeleteUser(context.TODO(),
 &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 for 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 **`ForgotPassword`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `ForgotPassword`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- 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 veja como configurar e executar no AWS Code Examples Repository.

```
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(clientId string, userName string)
 (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
 &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 for 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 códigos a seguir mostram como usar `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

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

Go

SDK para Go V2

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

```
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(clientId string, userName string, password
 string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
 &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 for Go da API.

JavaScript

SDK para JavaScript (v3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for JavaScript da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 códigos a seguir mostram como usar `ListUserPools`.

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

CLI

AWS CLI

Como listar grupos de usuários

Este exemplo lista até vinte grupos de usuários.

Comando:

```
aws cognito-idp list-user-pools --max-results 20
```

Saída:

```
{
  "UserPools": [
      {
         "CreationDate": 1547763720.822,
         "LastModifiedDate": 1547763720.822,
         "LambdaConfig": {},
         "Id": "us-west-2_aaaaaaaaa",
         "Name": "MyUserPool"
      }
  ]
}
```

- 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 veja 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() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  fmt.Println("Couldn't load default configuration. Have you set up your AWS
 account?")
  fmt.Println(err)
  return
 }
 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(context.TODO())
  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 for Go da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 veja 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 códigos a seguir mostram como usar `ListUsers`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

CLI

AWS CLI

Como listar usuários

Este exemplo lista até vinte usuários.

Comando:

```
aws cognito-idp list-users --user-pool-id us-west-2_aaaaaaaaa --limit 20
```

Saída:

```
{
  "Users": [
      {
          "Username": "22704aa3-fc10-479a-97eb-2af5806bd327",
          "Enabled": true,
          "UserStatus": "FORCE_CHANGE_PASSWORD",
          "UserCreateDate": 1548089817.683,
          "UserLastModifiedDate": 1548089817.683,
          "Attributes": [
              {
                  "Name": "sub",
                  "Value": "22704aa3-fc10-479a-97eb-2af5806bd327"
              },
              {
                  "Name": "email_verified",
                  "Value": "true"
              },
              {
                  "Name": "email",
                  "Value": "mary@example.com"
              }
          ]
      }
  ]
}
```

- 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 veja 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 veja 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 for JavaScript da API.

Kotlin

SDK para Kotlin

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

```
suspend fun listAllUsers(userPoolId: String) {

    val request = ListUsersRequest {
        this.userPoolId = userPoolId
    }

    CognitoIdentityProviderClient { 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 veja 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).

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 códigos a seguir mostram como usar `ResendConfirmationCode`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for 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 veja 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 veja 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 for JavaScript da API.

Kotlin

SDK para Kotlin

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

```
suspend fun resendConfirmationCode(clientIdVal: String?, userNameVal: String?) {
    val codeRequest = ResendConfirmationCodeRequest {
        clientId = clientIdVal
        username = userNameVal
    }
```

```
    CognitoIdentityProviderClient { 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 veja 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).

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 códigos a seguir mostram como usar `RespondToAuthChallenge`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

CLI

AWS CLI

Como responder a um desafio de autorização

Este exemplo responde a um desafio de autorização iniciado com initiate-auth. É uma resposta ao desafio NEW_PASSWORD_REQUIRED. Ele define uma senha para o usuário jane@example.com.

Comando:

```
aws cognito-idp respond-to-auth-challenge --client-id 3n4b5urk1ft4fl3mg5e62d9ado
 --challenge-name NEW_PASSWORD_REQUIRED --challenge-responses
 USERNAME=jane@example.com,NEW_PASSWORD="password" --session "SESSION_TOKEN"
```

Saída:

```
{
  "ChallengeParameters": {},
  "AuthenticationResult": {
      "AccessToken": "ACCESS_TOKEN",
      "ExpiresIn": 3600,
      "TokenType": "Bearer",
      "RefreshToken": "REFRESH_TOKEN",
      "IdToken": "ID_TOKEN",
      "NewDeviceMetadata": {
          "DeviceKey": "us-west-2_fec070d2-fa88-424a-8ec8-b26d7198eb23",
          "DeviceGroupKey": "-wt2ha1Zd"
      }
  }
}
```

- 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 veja 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 for JavaScript da API.

Python

SDK para Python (Boto3)

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 códigos a seguir mostram como usar `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

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

C++

SDK para C++

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

```
type CognitoActions struct {
 CognitoClient *cognitoidentityprovider.Client
}



// SignUp signs up a user with Amazon Cognito.
func (actor CognitoActions) SignUp(clientId string, userName string, password
 string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(context.TODO(),
 &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 for Go da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 veja 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 for JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 { 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 veja 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).

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 códigos a seguir mostram como usar `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

Este exemplo adiciona tags a um grupo de usuários.

Comando:

```
aws cognito-idp update-user-pool --user-pool-id us-west-2_aaaaaaaaa --user-pool-
tags Team=Blue,Area=West
```

- 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 veja como configurar e executar no AWS Code Examples Repository.

```
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(userPoolId string,
 triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
 &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(context.TODO(),
&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 for 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 **`VerifySoftwareToken`** com um AWS SDK ou CLI

Os exemplos de códigos a seguir mostram como usar `VerifySoftwareToken`.

Exemplos de ações são trechos de código de programas maiores e devem ser executados em contexto. É possível ver essa ação no contexto no seguinte exemplo de código:

- Inscrever um usuário em um grupo de usuários que exija MFA

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for .NET da API.

C++

SDK para C++

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 for C++ da API.

Java

SDK para Java 2.x

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 veja 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 for JavaScript da API.

Kotlin

SDK para Kotlin

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 { 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 veja 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).

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 SDKs AWS

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. Cada cenário inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código.

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 exige MFA usando um SDK AWS
- 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

O exemplo de código a seguir mostra como confirmar 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 `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 veja como configurar e executar no AWS Code Examples Repository.

Execute um cenário interativo em um prompt de comando.

```
// 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(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(
  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(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(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(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(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(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(stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup()
  }
```

```
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]
 runner.helper.PopulateUserTable(stackOutputs["TableName"])

 runner.AddPreSignUpTrigger(stackOutputs["UserPoolId"],
 stackOutputs["AutoConfirmFunctionArn"])
 runner.resources.triggers = append(runner.resources.triggers, actions.PreSignUp)
 userName, password := runner.SignUpUser(stackOutputs["UserPoolClientId"],
 stackOutputs["TableName"])
 runner.helper.ListRecentLogEvents(stackOutputs["AutoConfirmFunction"])
 runner.resources.userAccessTokens = append(runner.resources.userAccessTokens,
  runner.SignInUser(stackOutputs["UserPoolClientId"], userName, password))

 runner.resources.Cleanup()

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Aborde o acionador `PreSignUp` com uma função do Lambda.

```
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() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Crie uma struct que realize tarefas comuns.

```
// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
 GetStackOutputs(stackName string) (actions.StackOutputs, error)
```

```
 PopulateUserTable(tableName string)
 GetKnownUsers(tableName string) (actions.UserList, error)
 AddKnownUser(tableName string, user actions.User)
 ListRecentLogEvents(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(stackName string)
 (actions.StackOutputs, error) {
 return helper.cfnActor.GetOutputs(stackName), nil
}
```

```
// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(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(tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(tableName string) (actions.UserList,
 error) {
 knownUsers, err := helper.dynamoActor.Scan(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(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(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(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(functionName)
 if err != nil {
  panic(err)
```

```
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
 events, err := helper.cwlActor.GetLogEvents(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.

```
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(userPoolId string,
 triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
 &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(context.TODO(),
 &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(clientId string, userName string, password
 string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(context.TODO(),
 &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(clientId string, userName string, password
 string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
 &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(clientId string, userName string)
 (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
 &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(clientId string, code string,
 userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
 &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(userAccessToken string) error {
 _, err := actor.CognitoClient.DeleteUser(context.TODO(),
 &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(userPoolId string, userName string,
 userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
 &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(userPoolId string, userName
 string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
 &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.

```
// 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(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(context.TODO(),
 &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(tableName string) (UserList, error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(context.TODO(), &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(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(context.TODO(), &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.

```
type CloudWatchLogsActions struct {
```

```
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(functionName string)
 (types.LogStream, error) {
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(context.TODO(),
 &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(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(context.TODO(),
 &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. AWS 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(stackName string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(context.TODO(),
 &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
}
```

Limpar recursos.

```
// 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() {
 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(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(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 for Go .
  - 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 veja como configurar e executar no AWS Code Examples Repository.

Execute um cenário interativo em um prompt de comando.

```
import (
 "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(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(
  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(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(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(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(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(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(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(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(stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup()
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]

 runner.AddMigrateUserTrigger(stackOutputs["UserPoolId"],
 stackOutputs["MigrateUserFunctionArn"])
 runner.resources.triggers = append(runner.resources.triggers,
 actions.UserMigration)
 resetNeeded, user := runner.SignInUser(stackOutputs["TableName"],
 stackOutputs["UserPoolClientId"])
 if resetNeeded {
  runner.helper.ListRecentLogEvents(stackOutputs["MigrateUserFunction"])
  runner.ResetPassword(stackOutputs["UserPoolClientId"], user)
 }

 runner.resources.Cleanup()

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
 log.Println(strings.Repeat("-", 88))
}
```

Aborde o acionador `MigrateUser` com uma função do Lambda.

```
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 output.Items == nil || 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() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  log.Panicln(err)
 }
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Crie uma struct que realize tarefas comuns.

```
// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
 GetStackOutputs(stackName string) (actions.StackOutputs, error)
 PopulateUserTable(tableName string)
 GetKnownUsers(tableName string) (actions.UserList, error)
 AddKnownUser(tableName string, user actions.User)
 ListRecentLogEvents(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(stackName string)
 (actions.StackOutputs, error) {
 return helper.cfnActor.GetOutputs(stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(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(tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(tableName string) (actions.UserList,
 error) {
 knownUsers, err := helper.dynamoActor.Scan(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(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(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(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(functionName)
 if err != nil {
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
 events, err := helper.cwlActor.GetLogEvents(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.

```
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(userPoolId string,
 triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
 &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(context.TODO(),
 &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(clientId string, userName string, password
 string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(context.TODO(),
 &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(clientId string, userName string, password
 string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
 &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(clientId string, userName string)
 (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
 &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(clientId string, code string,
 userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
 &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(userAccessToken string) error {
 _, err := actor.CognitoClient.DeleteUser(context.TODO(),
 &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(userPoolId string, userName string,
 userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
 &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(userPoolId string, userName
 string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
 &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.

```
// 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(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(context.TODO(),
 &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(tableName string) (UserList, error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(context.TODO(), &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(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(context.TODO(), &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.

```
type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(functionName string)
 (types.LogStream, error) {
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(context.TODO(),
 &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(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(context.TODO(),
 &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. AWS 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(stackName string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(context.TODO(),
 &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
}
```

Limpar recursos.

```
// 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() {
 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(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(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 for 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 exige MFA usando um SDK AWS

Os exemplos de código a seguir mostram como:

- Inscreva e confirme um usuário com nome de usuário, senha e endereço de e-mail.
- Configurar a autenticação multifator associando uma aplicação de MFA ao usuário.
- Fazer login usando uma senha e um código de MFA.

.NET

AWS SDK for .NET

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e saiba como configurar e executar no Repositório de exemplos de código da AWS.

```
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 for .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 Repositório de exemplos de código da AWS.

```
        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 for 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 Repositório de exemplos de código da AWS.

```
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 veja 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 { log } 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);
    log(`Signing up.`);
    await signUp({ clientId, username, password, email });
    log(`Signed up. A confirmation email has been sent to: ${email}.`);
    log(`Run 'confirm-sign-up ${username} <code>' to confirm your account.`);
  } catch (err) {
    log(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 { log } 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);
    log(`Confirming user.`);
    await confirmSignUp({ clientId, username, code });
    log(
      `User confirmed. Run 'admin-initiate-auth ${username} <password>' to sign
 in.`,
```

```
    );
  } catch (err) {
    log(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 { log } 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);

    log("Signing in.");
    const { ChallengeName, Session } = await adminInitiateAuth({
      clientId,
      userPoolId,
      username,
```

```
      password,
    });

    if (ChallengeName === "MFA_SETUP") {
      log("MFA setup is required.");
      return handleMfaSetup(Session, username);
    }

    if (ChallengeName === "SOFTWARE_TOKEN_MFA") {
      handleSoftwareTokenMfa(Session);
      log(`Run 'admin-respond-to-auth-challenge ${username} <totp>'`);
    }
  } catch (err) {
    log(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 { log } 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);

    log("Successfully authenticated.");
  } catch (err) {
    log(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 { log } 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);

    log("Verifying TOTP.");
    await verifySoftwareToken(totp);
    log("TOTP Verified. Run 'admin-initiate-auth' again to sign-in.");
  } catch (err) {
```

```
    console.log(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 for 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 veja 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 { 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 { 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 { 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 { 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 { 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 { 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 { 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 { region = "us-east-1" }.use
 { identityProviderClient ->
        identityProviderClient.signUp(signUpRequest)
        println("User has been signed up")
    }
}
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API 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 veja 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

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 veja como configurar e executar no AWS Code Examples Repository.

Execute um cenário interativo em um prompt de comando.

```
// 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(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(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(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(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(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(
  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(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(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(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(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(stackName string) {
 defer func() {
  if r := recover(); r != nil {
   log.Println("Something went wrong with the demo.")
   runner.resources.Cleanup()
  }
 }()

 log.Println(strings.Repeat("-", 88))
 log.Printf("Welcome\n")

 log.Println(strings.Repeat("-", 88))

 stackOutputs, err := runner.helper.GetStackOutputs(stackName)
 if err != nil {
  panic(err)
 }
 runner.resources.userPoolId = stackOutputs["UserPoolId"]
 runner.helper.PopulateUserTable(stackOutputs["TableName"])
 userName, password := runner.AddUserToPool(stackOutputs["UserPoolId"],
 stackOutputs["TableName"])

 runner.AddActivityLogTrigger(stackOutputs["UserPoolId"],
 stackOutputs["ActivityLogFunctionArn"])
 runner.SignInUser(stackOutputs["UserPoolClientId"], userName, password)
 runner.helper.ListRecentLogEvents(stackOutputs["ActivityLogFunction"])
 runner.GetKnownUserLastLogin(stackOutputs["TableName"], userName)

 runner.resources.Cleanup()

 log.Println(strings.Repeat("-", 88))
 log.Println("Thanks for watching!")
```

```
 log.Println(strings.Repeat("-", 88))
}
```

Aborde o acionador `PostAuthentication` com uma função do Lambda.

```
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() {
 sdkConfig, err := config.LoadDefaultConfig(context.TODO())
 if err != nil {
  log.Panicln(err)
 }
```

```
 h := handler{
  dynamoClient: dynamodb.NewFromConfig(sdkConfig),
 }
 lambda.Start(h.HandleRequest)
}
```

Crie uma struct que realize tarefas comuns.

```
// IScenarioHelper defines common functions used by the workflows in this
 example.
type IScenarioHelper interface {
 Pause(secs int)
 GetStackOutputs(stackName string) (actions.StackOutputs, error)
 PopulateUserTable(tableName string)
 GetKnownUsers(tableName string) (actions.UserList, error)
 AddKnownUser(tableName string, user actions.User)
 ListRecentLogEvents(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(stackName string)
 (actions.StackOutputs, error) {
 return helper.cfnActor.GetOutputs(stackName), nil
}

// PopulateUserTable fills the known user table with example data.
func (helper ScenarioHelper) PopulateUserTable(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(tableName)
 if err != nil {
  panic(err)
 }
}

// GetKnownUsers gets the users from the known users table in a structured
 format.
func (helper ScenarioHelper) GetKnownUsers(tableName string) (actions.UserList,
 error) {
 knownUsers, err := helper.dynamoActor.Scan(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(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(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(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(functionName)
 if err != nil {
  panic(err)
 }
 log.Printf("Getting some recent events from log stream %v\n",
 *logStream.LogStreamName)
 events, err := helper.cwlActor.GetLogEvents(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.

```
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(userPoolId string,
 triggers ...TriggerInfo) error {
 output, err := actor.CognitoClient.DescribeUserPool(context.TODO(),
 &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(context.TODO(),
 &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(clientId string, userName string, password
 string, userEmail string) (bool, error) {
 confirmed := false
 output, err := actor.CognitoClient.SignUp(context.TODO(),
 &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(clientId string, userName string, password
 string) (*types.AuthenticationResultType, error) {
 var authResult *types.AuthenticationResultType
 output, err := actor.CognitoClient.InitiateAuth(context.TODO(),
 &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(clientId string, userName string)
 (*types.CodeDeliveryDetailsType, error) {
 output, err := actor.CognitoClient.ForgotPassword(context.TODO(),
 &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(clientId string, code string,
 userName string, password string) error {
 _, err := actor.CognitoClient.ConfirmForgotPassword(context.TODO(),
 &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(userAccessToken string) error {
 _, err := actor.CognitoClient.DeleteUser(context.TODO(),
 &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(userPoolId string, userName string,
 userEmail string) error {
 _, err := actor.CognitoClient.AdminCreateUser(context.TODO(),
 &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(userPoolId string, userName
 string, password string) error {
 _, err := actor.CognitoClient.AdminSetUserPassword(context.TODO(),
 &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.

```
// 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(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(context.TODO(),
 &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(tableName string) (UserList, error) {
 var userList UserList
 output, err := actor.DynamoClient.Scan(context.TODO(), &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(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(context.TODO(), &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.

```
type CloudWatchLogsActions struct {
 CwlClient *cloudwatchlogs.Client
}

// GetLatestLogStream gets the most recent log stream for a Lambda function.
func (actor CloudWatchLogsActions) GetLatestLogStream(functionName string)
 (types.LogStream, error) {
 var logStream types.LogStream
 logGroupName := fmt.Sprintf("/aws/lambda/%s", functionName)
 output, err := actor.CwlClient.DescribeLogStreams(context.TODO(),
 &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(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(context.TODO(),
 &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. AWS 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(stackName string) StackOutputs {
 output, err := actor.CfnClient.DescribeStacks(context.TODO(),
 &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
}
```

Limpar recursos.

```
// 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() {
 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(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(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 for 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 SDKs AWS

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 funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

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

Ações para o Amazon Cognito Sync usando SDKs AWS

Os exemplos de código a seguir demonstram como realizar ações individuais do Amazon Cognito Sync com AWS SDKs. Esses trechos chamam a API de sincronização do Amazon Cognito e são trechos de código de programas maiores que devem ser executados no contexto. 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 ou CLI

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

Rust

SDK para Rust

> **Note**
>
> Tem mais sobre GitHub. Encontre o exemplo completo e veja 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 aplicativos multilocatários que geram um volume de solicitações que devem permanecer dentro das cotas do Amazon Cognito. Para aumentar essa capacidade à medida que sua base de clientes cresce, você pode comprar capacidade de cota adicional.

> **Note**
>
> As cotas do Amazon Cognito são aplicadas por e. Conta da AWS Região da AWS Essas cotas são compartilhadas entre todos os locatários da aplicação. Analise as cotas do serviço Amazon Cognito e certifique-se de que a cota atenda ao volume esperado e ao número esperado de inquilinos em seu aplicativo.

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 multirregional 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 em sua organização.

A multilocação em uma única região também pode trazer vantagens para seus clientes e administradores.

A lista a seguir aborda algumas das vantagens da multilocação com recursos compartilhados.

Vantagens da multilocação

Diretório comum de usuários

A multilocação oferece suporte a modelos em que os clientes têm contas em mais de um aplicativo. Você pode vincular identidades de fornecedores terceirizados 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 a mesma segurança avançada, autenticação multifator (MFA) AWS WAFe padrões 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. Quando quiser manter uma configuração de segurança consistente em aplicativos 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 Lambda em grupos de usuários e eventos do Amazon Cognito em grupos 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 antes que você possa enviar mensagens SMS aos seus usuários. Você pode enviar mensagens de e-mail com identidades e domínios verificados pelo Amazon Simple Email Service (Amazon SES) que estão contidos em uma região.

Com a multilocação, você pode compartilhar essa sobrecarga de configuração e manutenção entre todos os seus inquilinos. Como o Amazon SNS e o Amazon SES não estão disponíveis em todos Regiões da AWS, dividir seus recursos entre regiões exige uma consideração adicional.

Ao usar provedores de mensagens personalizados, você obtém a personalização comum de uma única função do Lambda para gerenciar sua entrega de mensagens.

A interface de usuário hospedada define um cookie de sessão no navegador para que ele reconheça um usuário que já tenha se autenticado. Quando você autentica usuários locais em um grupo de usuários, o cookie de sessão os autentica para todos os clientes do aplicativo 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 aplicativos com um cookie de sessão de interface de usuário hospedado.

- Separe seus usuários em grupos de usuários por locatário.
- Substitua o login da interface hospedada pelo login da API de grupos de usuários do Amazon Cognito.

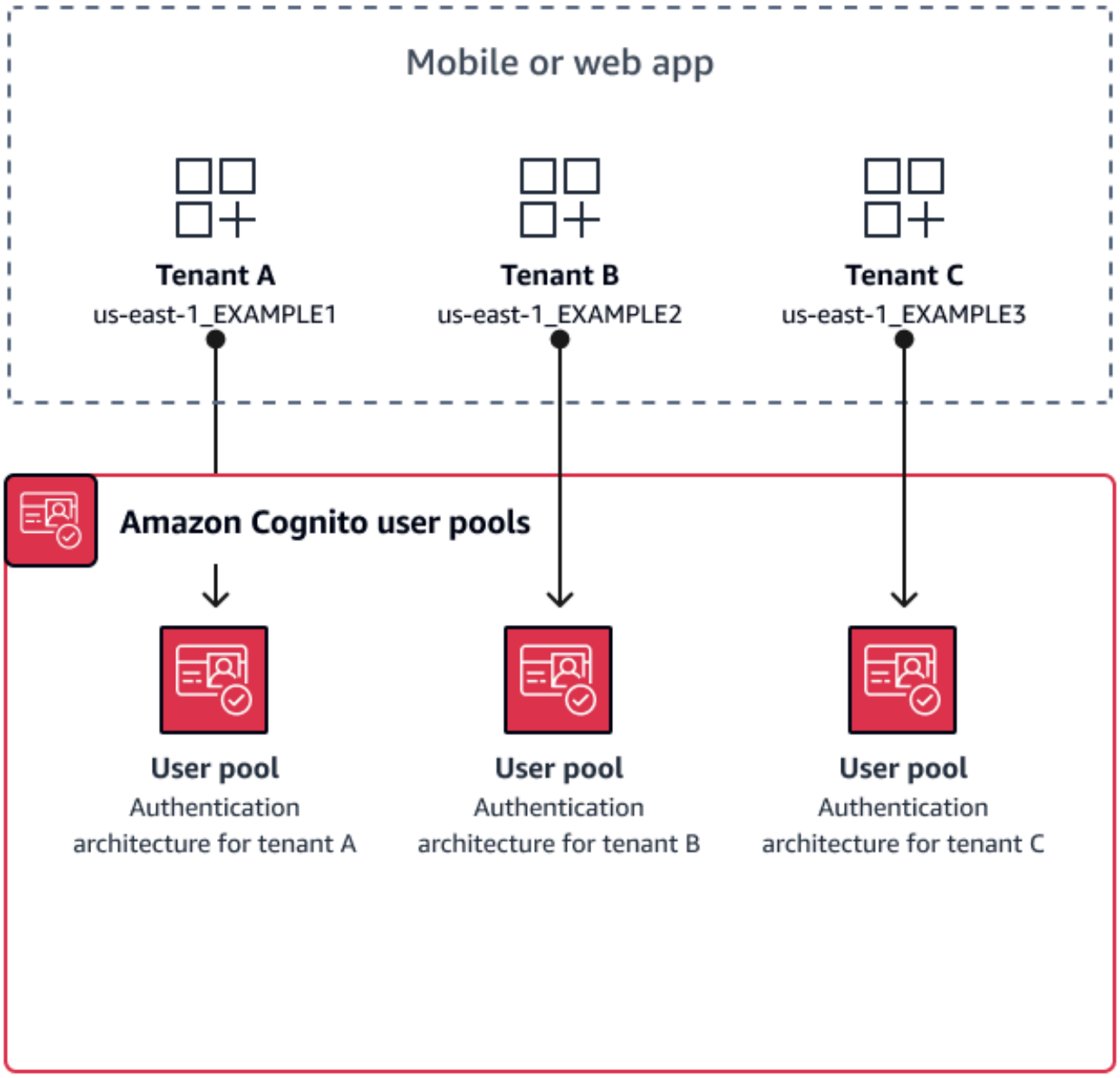
Tópicos

Melhores práticas de multilocação para grupos de usuários

Crie um grupo de usuários para cada inquilino em seu aplicativo. 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.

Usando essa abordagem, você pode configurar uma interface de usuário hospedada para cada locatário de forma independente e redirecionar os usuários para a instância específica do seu aplicativo. Você também pode usar essa abordagem para se integrar a serviços de back-end, como o Amazon API Gateway.

O diagrama a seguir mostra cada inquilino 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. Considere um exemplo em que você tem dois inquilinos educacionais. O mesmo usuário pode ser um aluno com acesso limitado em um aplicativo e um professor com um alto nível de permissões em outro. Você pode precisar de MFA em um aplicativo, mas não em outro, ou ter uma política de

senha diferente. Como os usuários locais podem entrar em vários clientes de aplicativos em grupos de usuários com a interface hospedada, a multilocação do grupo de usuários também é ideal quando você deseja que mais de um dos seus locatários faça login com a interface hospedada.

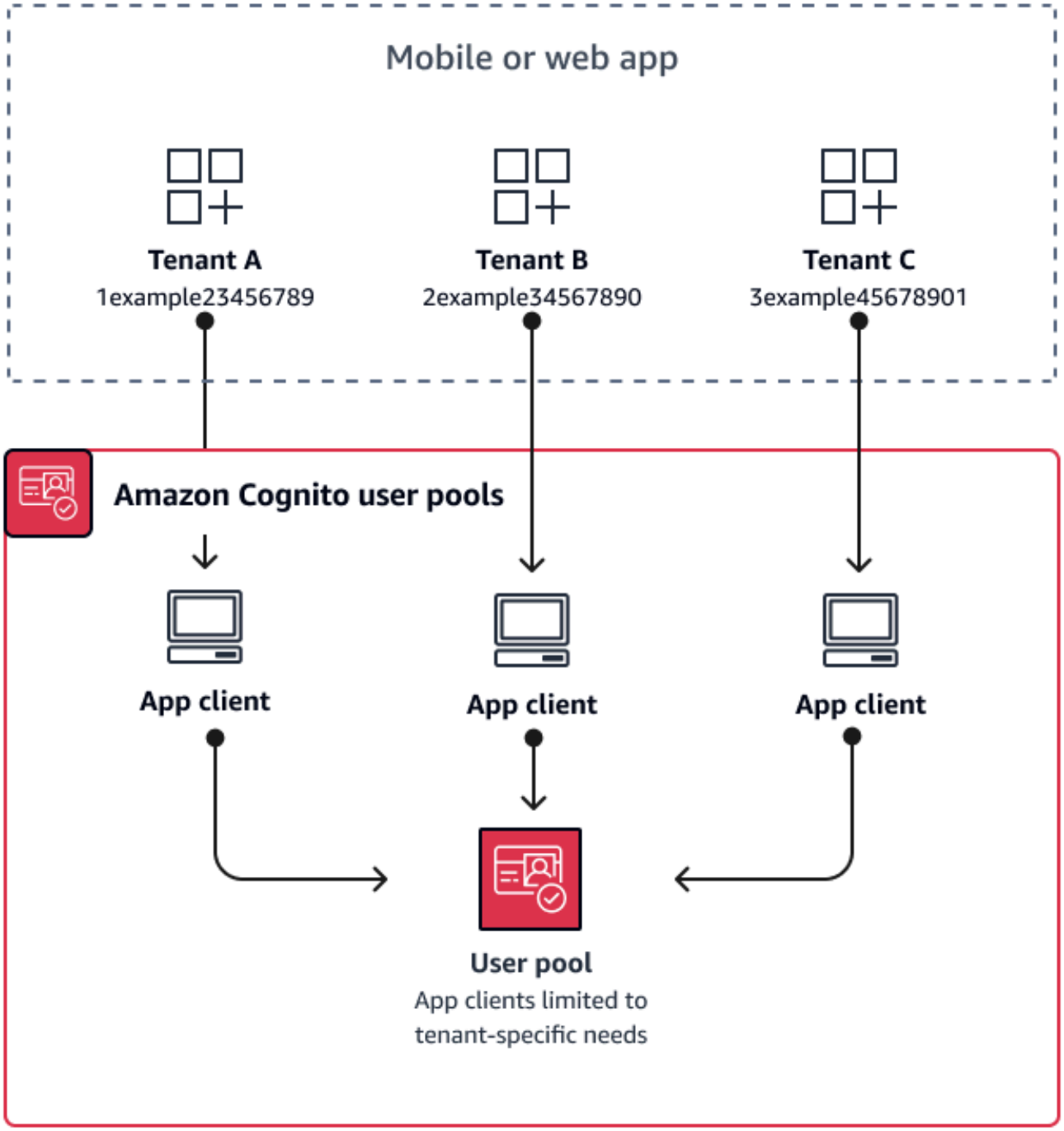
Nível de esforço

O esforço de desenvolvimento e operação para usar essa abordagem é alto. Para garantir resultados consistentes e previsíveis para sua família de aplicativos, você deve integrar os recursos do Amazon Cognito às suas ferramentas de automação e manter suas linhas de base à medida que sua arquitetura de autenticação se torna mais complexa. Quando quiser criar um único ponto de partida para seus aplicativos, você precisa criar os elementos da interface do usuário (UI) para capturar a decisão inicial que direciona os usuários para o recurso correto.

Melhores práticas de multilocação entre aplicativos e clientes

Crie um cliente de aplicativo para cada inquilino em seu aplicativo. Com a multilocação entre aplicativos e clientes, você pode atribuir qualquer usuário a clientes de aplicativos vinculados ao locatário e manter 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 inquilino com um cliente de aplicativo dedicado em um grupo de usuários compartilhado.



Quando implementar a multilocação entre aplicativos e clientes

Quando você pode escolher uma configuração universal para configurações no nível do grupo de usuários, como acionadores Lambda, política de senha e os 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 aplicativos e clientes exige um esforço moderado. Um grande desafio da multilocação entre aplicativos e clientes é a capacidade dos locatários de apresentar um cookie de interface de usuário hospedado e alternar entre aplicativos. Em uma arquitetura de multilocação entre aplicativos e clientes, evite o login na interface de usuário hospedada quando o isolamento for necessário. Você pode distribuir seu aplicativo móvel ou links para seu aplicativo web com a lógica do cliente de aplicativo 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 grupos de identidades.

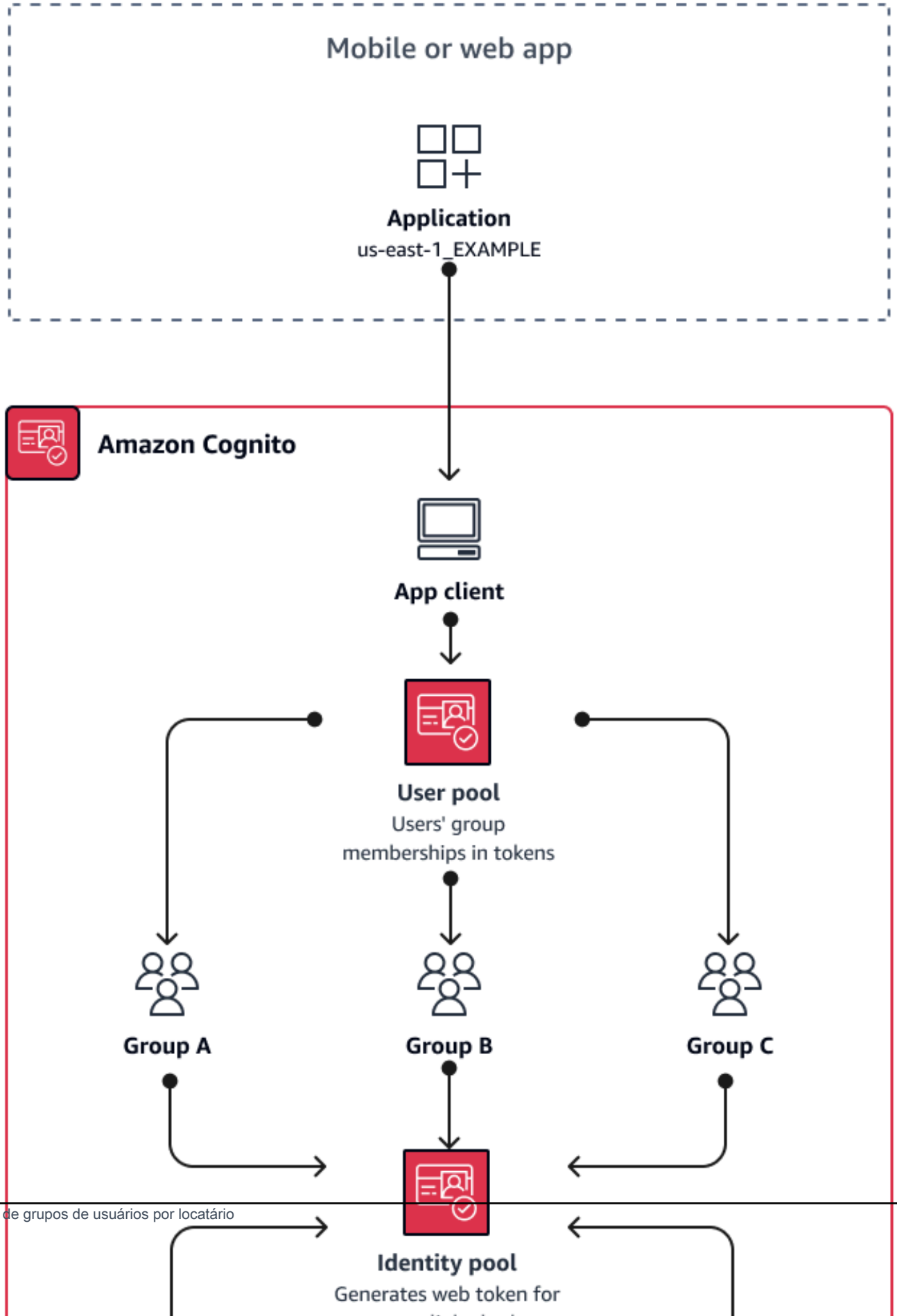
Melhores práticas de multilocação de grupos de usuários

A multilocação baseada em grupo funciona melhor quando sua arquitetura exige grupos de usuários do Amazon Cognito com grupos de identidades.

O ID do grupo de usuários e os tokens de acesso contêm uma `cognito:groups` reivindicação. Além disso, os tokens de identificação contêm `cognito:roles` e `cognito:preferred_role` reivindicações. Quando o resultado principal da autenticação em seu aplicativo são AWS credenciais temporárias de um grupo de identidades, as associações de grupos de seus usuários podem determinar a função e as permissões do IAM que eles recebem.

Como exemplo, considere três locatários que armazenam ativos de aplicativos em seu próprio bucket do Amazon S3. Atribua os usuários de cada locatário a um grupo associado, configure uma função preferencial para o grupo e conceda a essa função acesso de leitura ao bucket.

O diagrama a seguir mostra inquilinos compartilhando um cliente de aplicativo e um grupo de usuários, com grupos dedicados no grupo de usuários que determinam sua elegibilidade para uma função 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

Quando implementar a multilocação em grupo

Quando o acesso aos AWS recursos é sua principal preocupação. Grupos nos grupos de usuários do Amazon Cognito Os grupos de usuários são um mecanismo de controle de acesso baseado em funções (RBAC). Você pode configurar vários grupos em um grupo de usuários e tomar decisões complexas de RBAC com prioridade de grupo. Os grupos de identidades podem atribuir credenciais para a função com a maior prioridade, qualquer função na reivindicação do grupo ou de outras reivindicações nos tokens de um usuário.

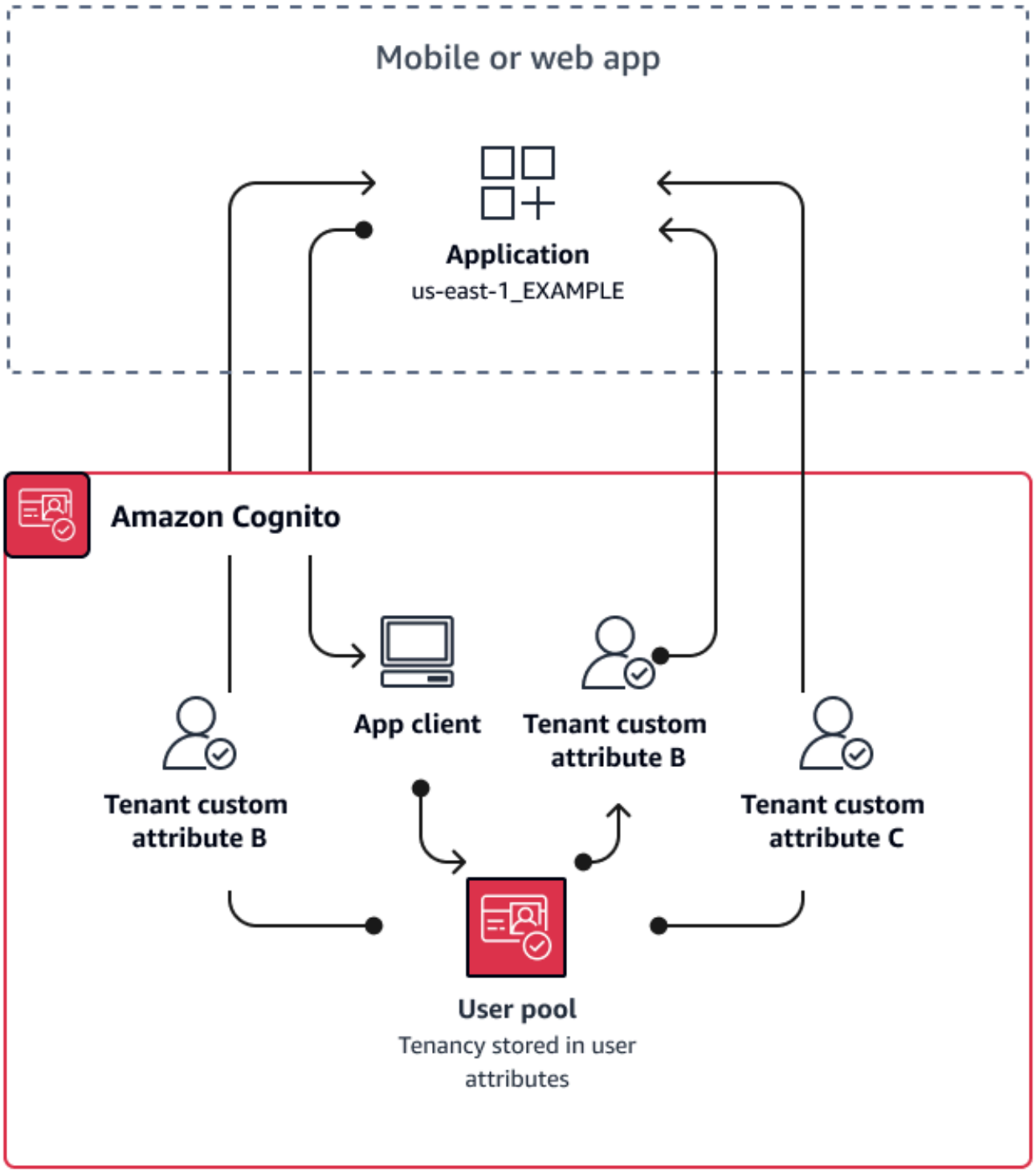
Nível de esforço

O nível de esforço para manter a multilocação apenas com a participação em grupos é baixo. No entanto, para expandir a função dos grupos de grupos de usuários além da capacidade integrada de seleção de funções do IAM, você deve criar uma lógica de aplicativo que processe a associação ao grupo nos tokens dos usuários e determine o que fazer no cliente. Você pode integrar as Permissões Verificadas da Amazon aos seus aplicativos 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 com atributos personalizados

O Amazon Cognito oferece suporte a 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`, seu aplicativo pode atribuir acesso a recursos específicos do inquilino adequadamente. Um atributo personalizado que define um ID de locatário deve ser imutável ou somente de leitura para o cliente do aplicativo.

O diagrama a seguir mostra os inquilinos compartilhando um cliente do aplicativo e um grupo de usuários, com atributos personalizados no grupo de usuários que indicam o inquilino ao qual eles pertencem.



Quando atributos personalizados determinam a locação, você pode distribuir um único aplicativo ou URL de login. Depois que o usuário fizer login, seu aplicativo poderá processar a

`custom:tenantID` solicitação, 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 identidade nas Permissões Verificadas da Amazon 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 é superficial. Um atributo de inquilino pode contribuir para os resultados da marca e do layout. Quando você deseja obter um isolamento significativo entre os inquilinos, os atributos personalizados não são a melhor escolha. Qualquer diferença entre locatários que precise ser configurada no nível do pool de usuários ou do aplicativo-cliente, como MFA ou marca de interface de usuário hospedada, exige que você crie distinções entre os locatários de uma forma que os atributos personalizados não oferecem. Com grupos de identidades, você pode até mesmo escolher a função do IAM de seus usuários na declaração 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 inquilino em seu aplicativo, o nível de esforço tende a ser alto. Se você já conhece bem uma configuração de cliente que analisa declarações do OIDC ou em Amazon Verified Permissions, essa abordagem pode exigir o menor nível de esforço.

Recomendações de segurança para locações múltiplas

Para ajudar a tornar sua aplicação mais segura, recomendamos o seguinte:

- Valide a locação em seu aplicativo com as permissões verificadas da Amazon. 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. As Permissões verificadas têm orientações adicionais para o gerenciamento de multilocação.
- 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 personalizados imutáveis ou somente para leitura para os atributos do perfil do usuário que identificam os inquilinos. Você só pode definir o valor dos atributos imutáveis ao criar um usuário ou um usuário se inscrever 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 do locatário e o cliente do aplicativo para evitar o acesso não autorizado entre inquilinos. 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 do seu aplicativo podem fazer login diretamente por meio de um grupo de usuários ou podem se federar por meio de um provedor de identidade (IdP) terceirizado. 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.

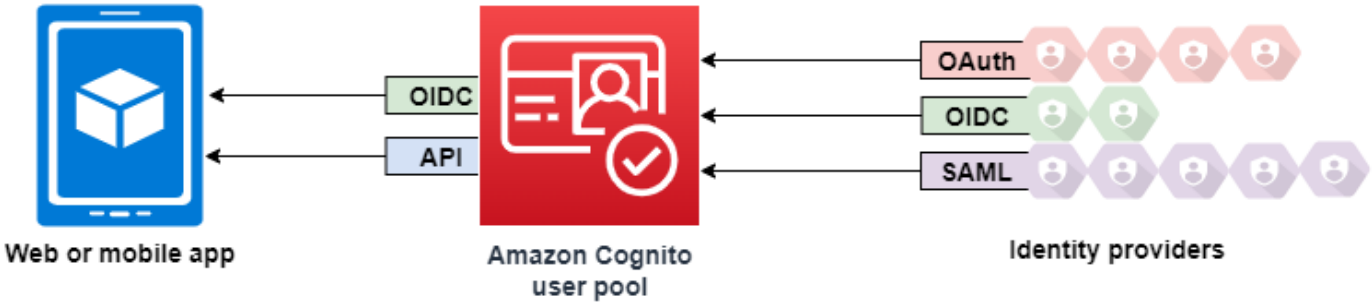
Tópicos

Autenticar com um grupo de usuários

Você pode permitir que os usuários sejam autenticados com um grupo de usuários. Os usuários do seu aplicativo podem fazer login diretamente por meio de um grupo de usuários ou podem se federar por meio de um provedor de identidade (IdP) terceirizado. 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

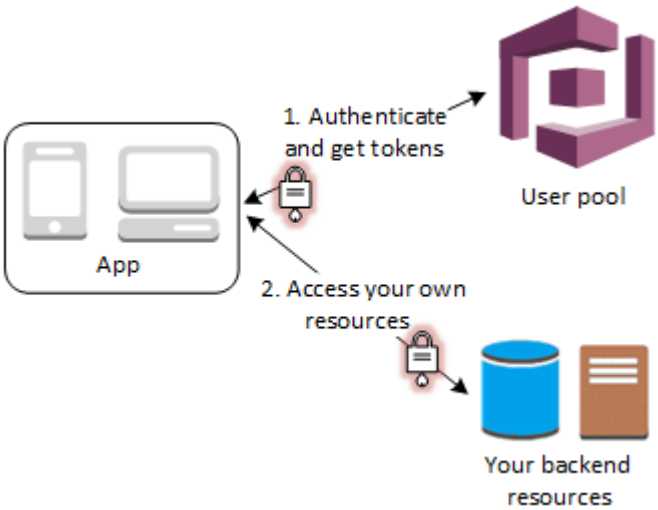
Depois de uma autenticação bem-sucedida, sua aplicação Web ou móvel receberá tokens do grupo de usuários do Amazon Cognito. Você pode usar esses tokens para recuperar AWS credenciais que permitem que seu aplicativo acesse outros AWS serviços, ou você pode optar por usá-los para controlar o acesso aos seus recursos do lado do servidor ou ao Amazon API Gateway.

Para obter mais informações, consulte Fluxo de autenticação de grupo de usuários e Como usar tokens com grupos de usuários.



Acessar os recursos no lado do servidor com um 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 do OAuth 2.0, você pode criar o próprio servidor de recursos para permitir que os usuários acessem recursos protegidos. Para ter mais informações, consulte Escopos, M2M e autorização de API com servidores de recursos.

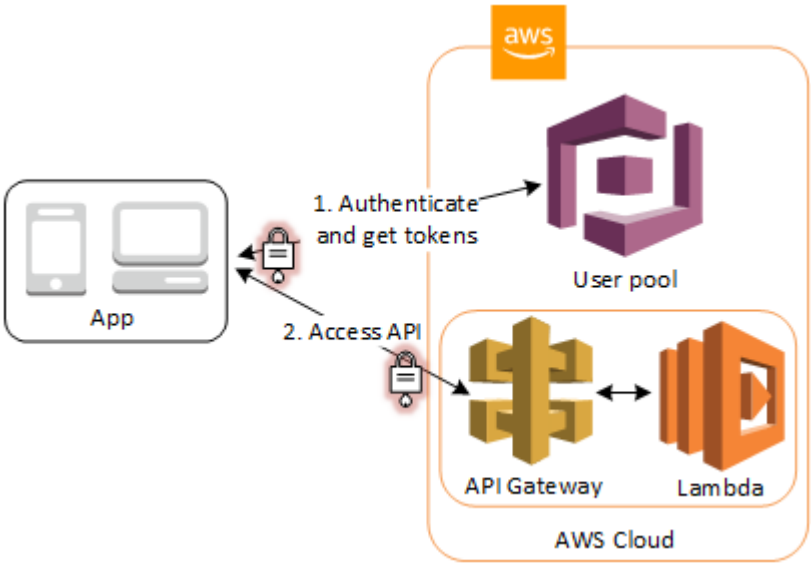
Para obter mais informações sobre a autenticação do grupo de usuários, consulte Fluxo de autenticação de grupo de usuários e Como usar tokens com grupos de usuários.

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.

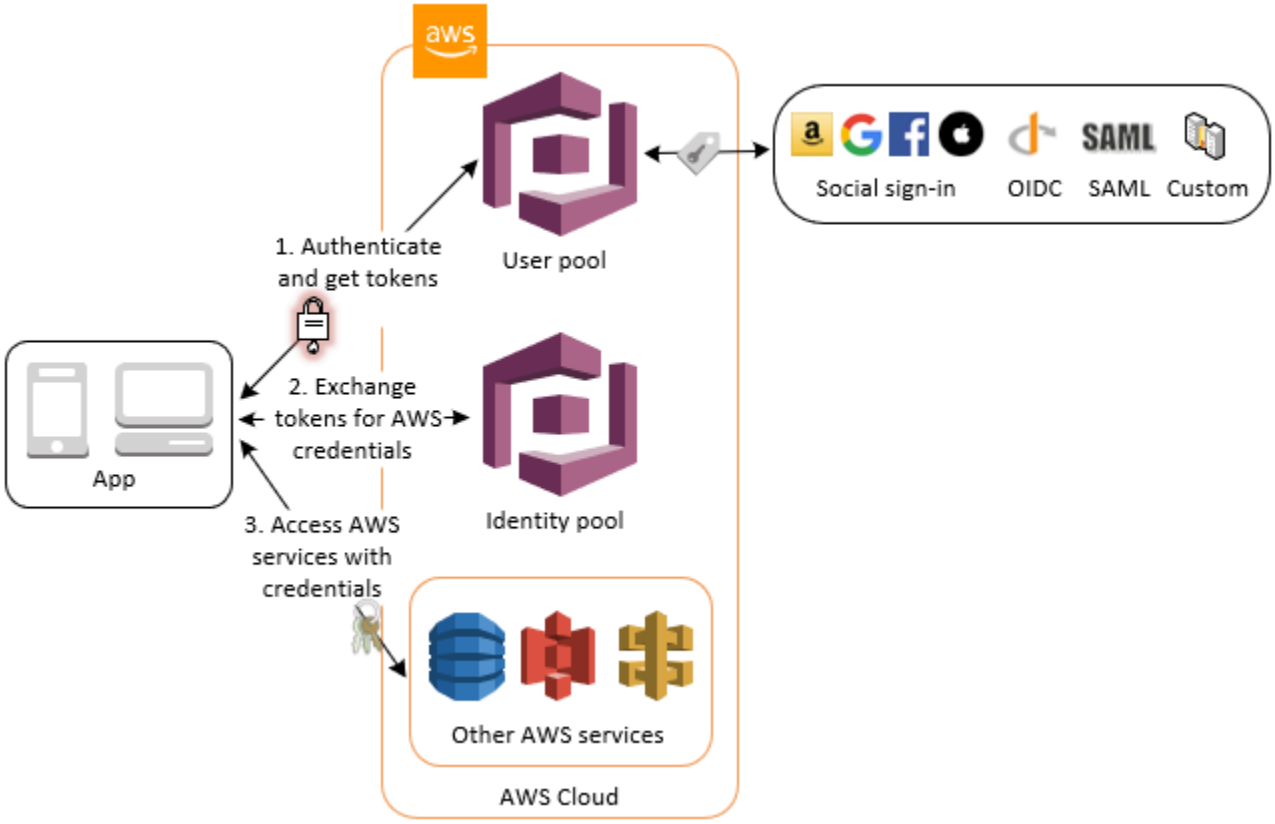
Você pode usar grupos em um grupo de usuários para controlar permissões com o API Gateway mapeando a associação ao grupo para funções do IAM. Os grupos dos quais um usuário é membro estão incluídos no token de ID fornecido por um grupo de usuários quando o usuário do aplicativo faz login. Para obter 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.



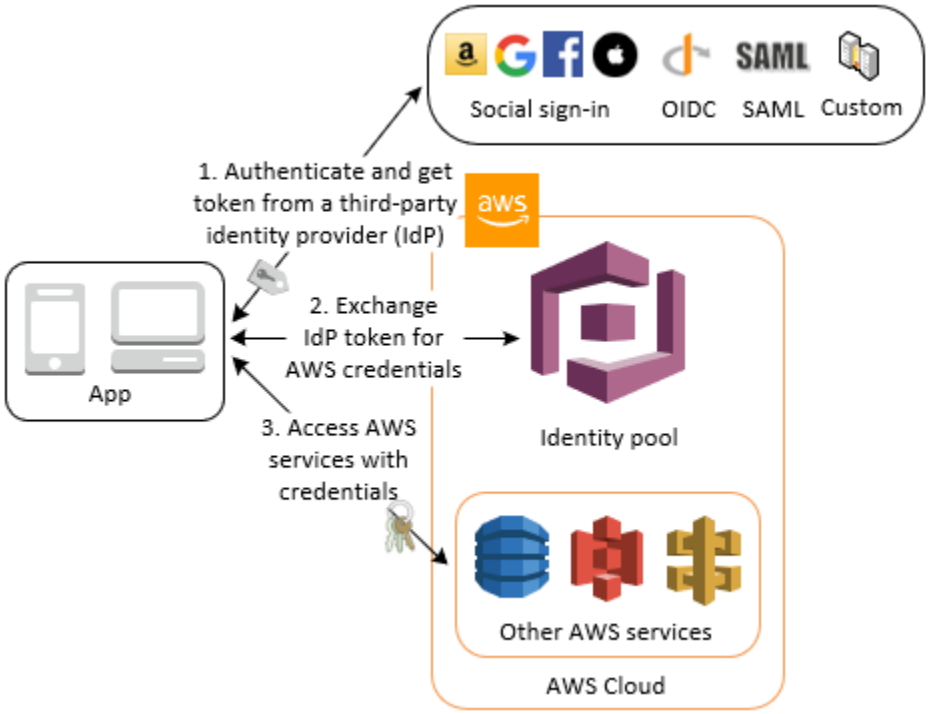
Acesse AWS serviços com um grupo de usuários e um pool de identidades

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 Introdução aos grupos de identidade do Amazon Cognito.



Autenticar com terceiros e acessar serviços da AWS com um grupo de identidades

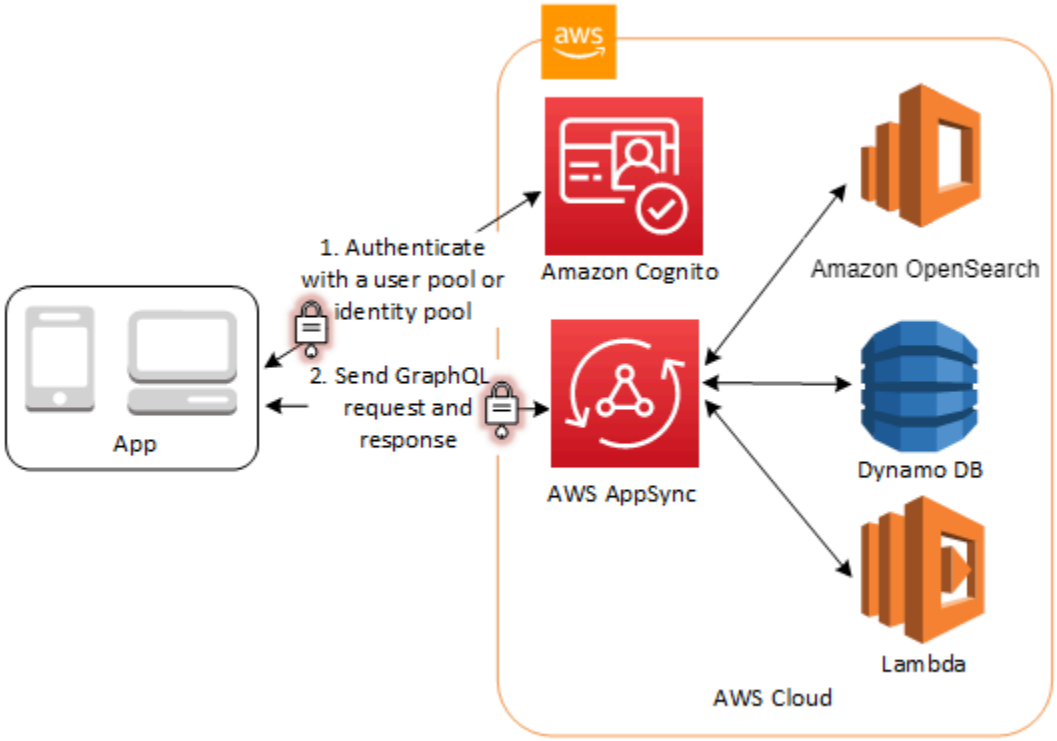
Você pode permitir que seus usuários acessem os AWS serviços por meio de um pool de identidades. Um grupo de identidades requer um token IdP de um usuário autenticado por um provedor de identidade de terceiros (ou nada se for um convidado anônimo). Em troca, o grupo de identidades concede AWS credenciais temporárias que você pode usar para acessar outros AWS serviços. Para ter mais informações, consulte Introdução aos grupos de identidade 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 ter mais informações, consulte a autorização AMAZON_COGNITO_USER_POOLS no Guia do desenvolvedor do AWS AppSync .

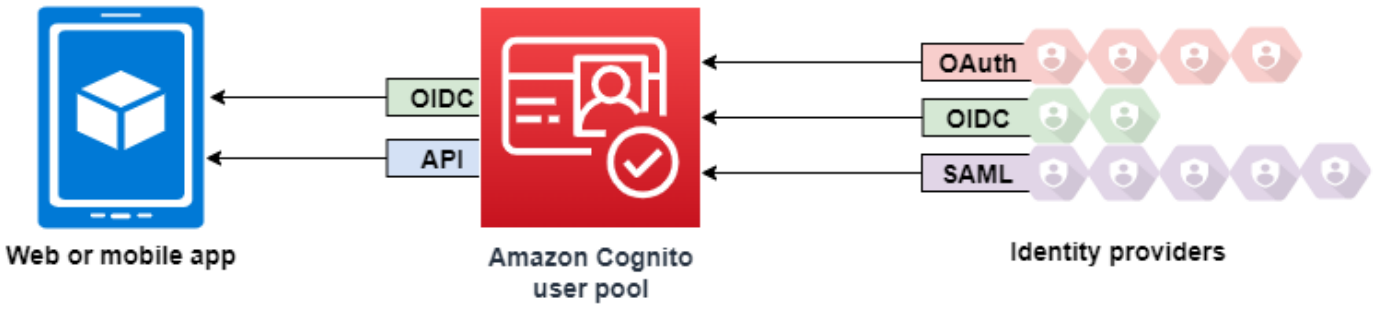
Você também pode assinar solicitações para a API AWS AppSync GraphQL com as credenciais do IAM que você recebe de um grupo de identidades. Consulte Autorização AWS_IAM.


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

- Atributos
- Autenticação com um grupo de usuários
- Usar a API de grupos de usuários e endpoints de grupo de usuários do Amazon Cognito
- Atualizar a configuração do grupo de usuários
- Configurar e usar a interface de usuário hospedada e endpoints de federação do Amazon Cognito
- Escopos, M2M e autorização de API com servidores de recursos
- Como adicionar acesso a grupo de usuários por meio de terceiros
- Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda
- Como usar análise do Amazon Pinpoint com grupos de usuários do Amazon Cognito
- Como gerenciar usuários em seu grupo 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 usar tokens com grupos de usuários
- Como acessar recursos após uma autenticação bem-sucedida do grupo de usuários
- Usar atributos de segurança de grupos de usuários do Amazon Cognito

Atributos

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 na guia Experiência de inscrição do 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, dados de `userInfo` do OAuth 2.0 e declarações do 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 da

- Como gerenciar usuários em seu grupo de usuários
- Usar a API de grupos de usuários e endpoints de grupo de usuários do Amazon Cognito
- Exemplos de código para o Amazon Cognito Identity Provider usando SDKs AWS

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 uma interface de usuário hospedada pelo Amazon Cognito ou com sua própria interface por meio da API de grupos de usuários do Amazon Cognito. O nível da aplicação por trás da interface do usuário personalizada de front-end pode autorizar solicitações no back-end com qualquer um dos vários métodos para confirmar solicitações legítimas.

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 consumidor com o login social do OAuth 2.0. O Amazon Cognito aceita login com Google, Facebook, Amazon e Apple por meio do OAuth 2.0.
2. Faça login e importe dados de usuários corporativos 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 da

- Como adicionar acesso a grupo de usuários por meio de terceiros
- Vincular usuários federados a um perfil de usuário existente

Minha achine-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 do OAuth 2.0, você pode adicionar um cliente de aplicativo que gere concessões de credenciais de cliente.

Tópicos relacionados da

- Escopos, M2M e autorização de API com servidores de recursos

Interface do usuário hospedada

Quando você não quiser criar uma interface do usuário, poderá apresentar aos usuários uma interface personalizada hospedada pelo Amazon Cognito. A interface do usuário hospedada é um conjunto de páginas da web para inscrição, login, autenticação multifator (MFA) e redefinição de senha. Você pode adicionar a interface hospedada ao seu domínio existente ou usar um identificador de prefixo em um AWS subdomínio.

Tópicos relacionados da

- Configurar e usar a interface de usuário hospedada e endpoints de federação do Amazon Cognito
- 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 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 a interface do usuário hospedada a gerencie. 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 recursos avançados de segurança, 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 as ACLs AWS WAF da web.

Tópicos relacionados da

- Adicionar MFA a um grupo de usuários
- Como adicionar segurança avançada a um grupo de usuários
- Associando 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 à interface do usuário hospedada uma aparência familiar aos usuários.

Tópicos relacionados da

- Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda
- Acionadores do Lambda de desafio personalizado de autenticação
- Como personalizar as páginas da Web integradas de cadastro e acesso

Monitoramento e análise

Os grupos de usuários do Amazon Cognito registram em log solicitações de API, incluindo solicitações à interface do usuário hospedada, no AWS CloudTrail. Você pode analisar métricas de desempenho no Amazon CloudWatch Logs, enviar registros personalizados CloudWatch com acionadores Lambda e monitorar o volume de solicitações de API no console de Service Quotas.

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 da

- Registro de chamadas da API do Amazon Cognito com AWS CloudTrail
- Rastreamento de cotas e uso em CloudWatch e Service Quotas
- Como usar análise do Amazon Pinpoint com grupos de usuários do Amazon Cognito

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.

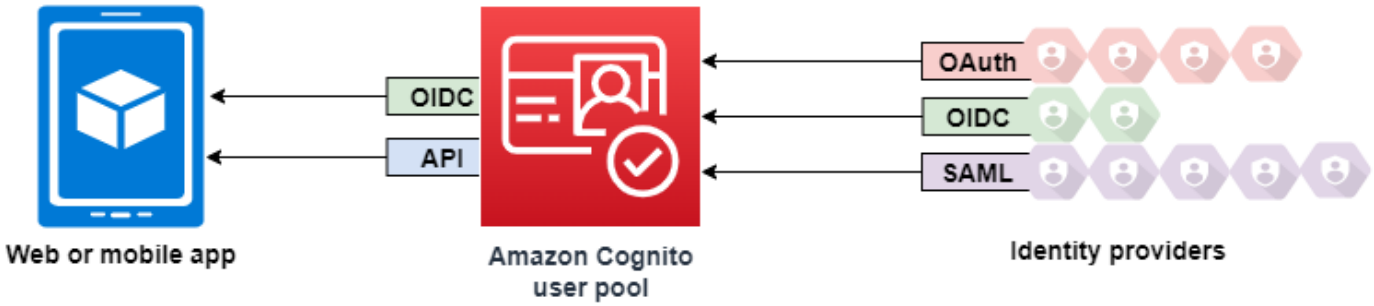
Tópicos relacionados

- Acessando Serviços da AWS usando um pool de identidades após o login
- Banco de identidades do Amazon Cognito

Autenticação com um grupo de usuários

Os usuários do seu aplicativo podem fazer login diretamente por meio de um grupo de usuários ou podem se federar por meio de um provedor de identidade (IdP) terceirizado. 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 da autenticação bem-sucedida, o Amazon Cognito retorna tokens do grupo de usuários à sua aplicação. Você pode usar os tokens para conceder aos usuários acesso aos seus próprios recursos no lado do servidor ou ao Amazon API Gateway. Ou você pode trocá-las por AWS credenciais para acessar outros AWS serviços.



O processamento e o gerenciamento de tokens do grupo de usuários para a aplicação da Web ou para dispositivos móveis são fornecidos no lado do cliente por meio dos SDKs do Amazon Cognito. Da mesma forma, o Mobile SDK for iOS e o Mobile SDK for Android atualizam automaticamente seus tokens de ID e de acesso se houver um token de atualização válido (não expirado) presente, e os tokens de ID e de acesso tiverem uma validade mínima restante de cinco minutos. Para obter informações sobre os SDKs e o código de amostra para Android e iOS JavaScript, consulte os SDKs do grupo de usuários do Amazon Cognito.

Depois que o usuário da aplicação faz login com êxito, o Amazon Cognito cria uma sessão e retorna um token de ID, de acesso e de atualização para o usuário autenticado.

JavaScript

```
// Amazon Cognito creates a session which includes the id, access, and refresh
 tokens of an authenticated user.

var authenticationData = {
        Username : 'username',
        Password : 'password',
    };
    var authenticationDetails = new
 AmazonCognitoIdentity.AuthenticationDetails(authenticationData);
    var poolData = { UserPoolId : 'us-east-1_ExaMPle',
        ClientId : '1example23456789'
    };
    var userPool = new AmazonCognitoIdentity.CognitoUserPool(poolData);
    var userData = {
        Username : 'username',
        Pool : userPool
    };
    var cognitoUser = new AmazonCognitoIdentity.CognitoUser(userData);
    cognitoUser.authenticateUser(authenticationDetails, {
```

```
        onSuccess: function (result) {
            var accessToken = result.getAccessToken().getJwtToken();

            /* Use the idToken for Logins Map when Federating User Pools with
 identity pools or when passing through an Authorization Header to an API Gateway
 Authorizer */
            var idToken = result.idToken.jwtToken;
        },

        onFailure: function(err) {
            alert(err);
        },

});
```

Android

```
// Session is an object of the type CognitoUserSession, and includes the id, access,
 and refresh tokens for a user.

String idToken = session.getIdToken().getJWTToken();
String accessToken = session.getAccessToken().getJWT();
```

iOS - swift

```
// AWSCognitoIdentityUserSession includes id, access, and refresh tokens for a user.

- (AWSTask<AWSCognitoIdentityUserSession *> *)getSession;
```

iOS - objective-C

```
// AWSCognitoIdentityUserSession includes the id, access, and refresh tokens for a
 user.

[[user getSession:@"username" password:@"password" validationData:nil scopes:nil]
 continueWithSuccessBlock:^id _Nullable(AWSTask<AWSCognitoIdentityUserSession *> *
 _Nonnull task) {
    // success, task.result has user session
    return nil;
}];
```

Tópicos

Fluxo de autenticação de grupo de usuários

O Amazon Cognito inclui vários métodos para autenticar os usuários. 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 novos tipos de desafio, além de senhas. A autenticação do Amazon Cognito normalmente exige que você implemente duas operações de API na seguinte ordem:

Public authentication

1. InitiateAuth
2. RespondToAuthChallenge

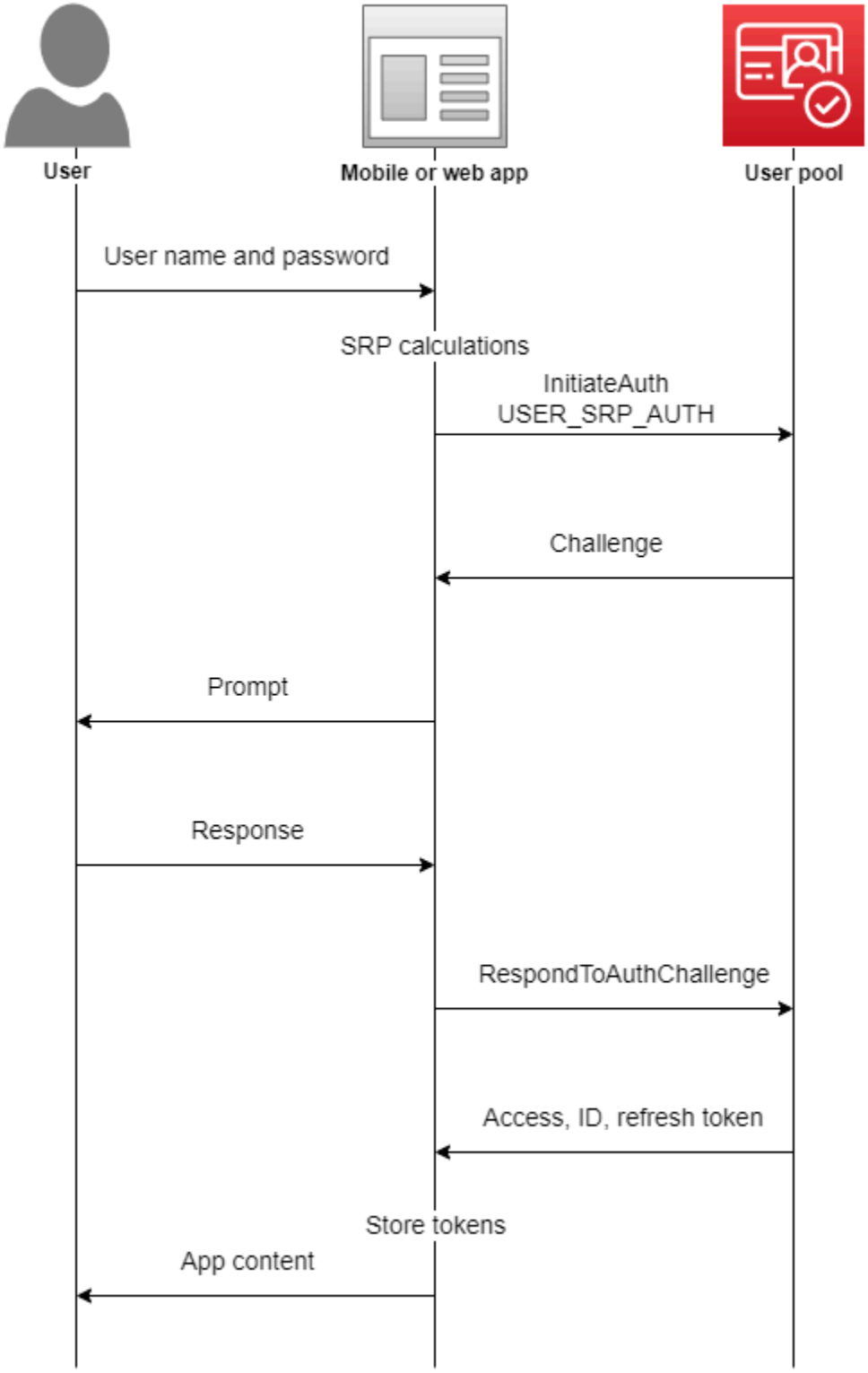
`InitiateAuth` e `RespondToAuthChallenge` são APIs não autenticadas para uso com clientes de aplicações públicas do lado do cliente.

Server-side authentication

1. AdminInitiateAuth
2. AdminRespondToAuthChallenge

`AdminInitiateAuth` e `AdminRespondToAuthChallenge` exigem credenciais do IAM e são adequadas para clientes de aplicações confidenciais do lado do servidor.

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.


User
Mobile or web app
User pool
User name and password
SRP calculations
InitiateAuth
USER_SRP_AUTH
Challenge
Prompt
Response
RespondToAuthChallenge
Access, ID, refresh token
Store tokens
App content

Normalmente, sua aplicação gera uma solicitação para coletar informações do usuário e as envia em uma solicitação de API ao Amazon Cognito. Considere um fluxo `InitiateAuth` em um grupo de usuários no qual você configurou o usuário com autenticação multifator (MFA).

1. A aplicação solicita que o usuário informe o nome de usuário e a senha.
2. Você precisa incluir o nome de usuário e a senha como parâmetros em `InitiateAuth`.
3. O Amazon Cognito retorna um desafio `SMS_MFA` e um identificador de sessão.
4. A aplicação solicita que o usuário forneça o código de MFA exibido no telefone.
5. Você precisa incluir esse código e o identificador da sessão na solicitação `RespondToAuthChallenge`.

Dependendo dos recursos do grupo de usuários, você pode acabar respondendo a vários desafios para `InitiateAuth` 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. A interface do usuário hospedada do Amazon Cognito define a duração da sessão como três minutos para autenticação multifator e oito minutos para códigos de redefinição de senha.

Amazon Cognito console

Como configurar a duração da sessão do fluxo de autenticação do cliente da aplicação (AWS Management Console)

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 Authentication flow session duration (Duração da sessão do fluxo de autenticação) 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.
4. Escolha Salvar alterações.

Amazon Cognito 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 Clientes de aplicações de grupos de usuários.

Você pode usar AWS Lambda gatilhos para personalizar a forma 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 a respectiva senha.

Comportamento de bloqueio do Amazon Cognito em tentativas fracassadas de login

Após cinco tentativas malsucedidas de login não autenticado ou autenticado pelo IAM com uma senha, o Amazon Cognito bloqueia o usuário por um 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.

Tópicos

Fluxo de autenticação no lado do cliente

O processo a seguir funciona para aplicações do lado do cliente do usuário que você cria com o AWS Amplify, o com 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**
   >
   > A aplicação gera detalhes do SRP com os recursos SRP do Amazon Cognito incorporados aos 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).

Fluxo de autenticação no lado do servidor

Se você não tiver uma aplicação de usuário, mas, em vez disso, usar uma aplicação Java, Ruby ou Node.js segura de backend ou no lado do servidor, poderá usar a API autenticada no lado do servidor para os grupos de usuários do Amazon Cognito.

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 no lado do servidor chama a operação de API `AdminInitiateAuth` (em vez de `InitiateAuth`). Essa operação requer AWS credenciais com permissões que incluem `cognito-idp:AdminInitiateAuth` e. `cognito-idp:AdminRespondToAuthChallenge` A operação retorna os parâmetros de autenticação necessários.
- Depois que a aplicação no lado do servidor tiver os parâmetros de autenticação, ela chamará a operação da API `AdminRespondToAuthChallenge` (em vez de `RespondToAuthChallenge`). A operação `AdminRespondToAuthChallenge` da API só é bem-sucedida quando você fornece AWS credenciais.

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.

As operações `AdminInitiateAuth` e `AdminRespondToAuthChallenge` da API não podem aceitar credenciais de username-and-password usuário para login de administrador, a menos que você permita explicitamente que elas façam isso de uma das seguintes formas:

- Inclua `ALLOW_ADMIN_USER_PASSWORD_AUTH` (anteriormente conhecido como `ADMIN_NO_SRP_AUTH`) no parâmetro `ExplicitAuthFlow` quando você chamar `CreateUserPoolClient` ou `UpdateUserPoolClient`.
- Adicione `ALLOW_ADMIN_USER_PASSWORD_AUTH` à lista Fluxos de autenticação para o cliente da aplicação. Configure clientes de aplicação na guia App integration (Integração de aplicação) em seu grupo de usuários, em App clients and analytics (Clientes de aplicação e análise). Para ter mais informações, consulte Clientes de aplicações de grupos de usuários.

Fluxo de autenticação personalizado

Os grupos de usuários do Amazon Cognito também possibilitam o uso de fluxos de autenticação personalizados, que podem ajudar você a criar um modelo de autenticação baseado em desafios/respostas usando gatilhos. AWS Lambda

> **Note**
>
> Você não pode usar os recursos avançados de segurança para credenciais comprometidas e autenticação adaptável com fluxos de autenticação personalizados. Para ter mais informações, consulte Como adicionar segurança avançada a um grupo de usuários.

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 integrado e desafios

O Amazon Cognito contém valores `AuthFlow` e `ChallengeName` integrados para que um fluxo de autenticação padrão possa validar um nome de usuário e a senha pelo protocolo Secure Remote Password (SRP). Os AWS SDKs têm suporte integrado para esses fluxos com o Amazon Cognito.

O fluxo inicia-se com o envio de `USER_SRP_AUTH` como o `AuthFlow` para `InitiateAuth`. Você também envia os valores `USERNAME` e `SRP_A` em `AuthParameters`. Se a chamada de `InitiateAuth` for bem-sucedida, a resposta incluiu `PASSWORD_VERIFIER` como `ChallengeName` e `SRP_B` nos parâmetros do desafio. Depois, o aplicativo chama `RespondToAuthChallenge` com o `ChallengeName` `PASSWORD_VERIFIER` e os parâmetros necessários em `ChallengeResponses`. Se a chamada para `RespondToAuthChallenge` for bem-sucedida e o usuário fizer login, o Amazon Cognito emitirá tokens. Se você tiver ativado a autenticação multifator (MFA) para o usuário, o Amazon Cognito retornará o `ChallengeName` da `SMS_MFA`. A aplicação pode fornecer o código necessário por meio de outra chamada para `RespondToAuthChallenge`.

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 administração

A prática recomendada para autenticação é usar as operações de API descritas em Fluxo de autenticação personalizado com a SRP para verificação de senha. Os AWS SDKs usam essa abordagem, e essa abordagem os ajuda a usar o SRP. No entanto, se você quiser evitar

os cálculos da SRP, um conjunto alternativo de operações da API de administrador está disponível para proteger servidores de backend. Para essas implementações administrativas de backend, use `AdminInitiateAuth` no lugar de `InitiateAuth`. Além disso, use `AdminRespondToAuthChallenge` no lugar de `RespondToAuthChallenge`. Como você pode enviar a senha como texto sem formatação, não é necessário fazer cálculos de SRP ao usar essas operações. Exemplo:

```
AdminInitiateAuth Request {
    "AuthFlow":"ADMIN_USER_PASSWORD_AUTH",
    "AuthParameters":{
        "USERNAME":"<username>",
        "PASSWORD":"<password>"
        },
    "ClientId":"<clientId>",
    "UserPoolId":"<userPoolId>"
}
```

Essas operações de autenticação de administração exigem credenciais de desenvolvedor e usam o processo de assinatura do AWS Signature Version 4 (SigV4). Essas operações estão disponíveis nos SDKs da AWS padrão, incluindo o Node.js, que é conveniente para funções do Lambda. Para usar essas operações e fazer com que elas aceitem senhas em texto não criptografado, é necessário ativá-las para a aplicação no console. Como alternativa, é possível transmitir `ADMIN_USER_PASSWORD_AUTH` para o parâmetro `ExplicitAuthFlow` em chamadas para `CreateUserPoolClient` ou `UpdateUserPoolClient`. As operações `InitiateAuth` e `RespondToAuthChallenge` não aceitam a `ADMIN_USER_PASSWORD_AUTH` `AuthFlow`.

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 comporta rastreamento de dispositivo. 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.

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 para grupos de usuários com um acionador Lambda de migração de usuário.

Clientes de aplicações de grupos de usuários

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 todos os IDs ou segredos do cliente da aplicação para que somente as aplicações clientes autorizadas 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.

Tipos de cliente de aplicação

Ao criar um cliente de aplicação no Amazon Cognito, você pode preencher as opções previamente com base nos tipos de cliente OAuth 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.

Você não pode alterar os segredos depois de criar uma aplicação. É possível criar uma aplicação com um novo segredo se você quiser alternar o segredo. Também é possível excluir um aplicativo para bloquear o acesso de aplicativos que usam esse ID de cliente de aplicativo.

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 da aplicação do Amazon Cognito podem emitir tokens JSON da web (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 pelo OAuth 2.0. 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 usar o token de ID Para 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 do OAuth 2.0. 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 usar o token de acesso Para 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 Como usar o token de atualização Para mais informações.

É possível definir a expiração desses tokens para cada cliente de aplicação na guia Integração 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.

URLs de retorno de chamada permitidos

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 - redirection endpoint (OAuth 2.0 - endpoint de redirecionamento).

O Amazon Cognito exige `HTTPS` em vez de `HTTP`, exceto `http://localhost` somente para fins de teste.

URLs de retorno de chamada do aplicativo, como `myapp://example`, também são compatíveis.

URLs de desconexão permitidos

Um URL de saída indica para onde o usuário deve ser redirecionado após fazer logoff.

Atribua 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 aplicativo 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. A aplicação pode permitir a autenticação com nome de usuário e senha, senha remota segura (SRP), autenticação personalizada com acionadores do Lambda e atualização de token. Como prática recomendada de segurança, use a autenticação SRP como principal método de login. A interface de usuário hospedada conecta automaticamente os usuários com o SRP.

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 autorização de API com 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 do `CallbackURLs` para seu cliente de aplicativo. O Amazon Cognito redireciona as sessões autenticadas para esse URL quando:

1. Seu cliente de aplicativo tem um provedor de identidade atribuído e vários URLs de retorno de chamada 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 URL de retorno de chamada definido. Nesse cenário, não é necessário definir uma URL de retorno de chamada padrão. Solicitações que não incluem um `redirect_uri` parâmetro 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-lo na página de login da interface de usuário hospedada. Você pode atribuir vários IdPs, mas deve atribuir pelo menos um. Para obter mais informações sobre o uso externo IdPs, consulteComo adicionar acesso a grupo de usuários por meio de terceiros.

Escopos do OpenID Connect

Selecione um ou mais dos seguintes escopos `OAuth` 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 do 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.

Tipos de concessão do OAuth

Uma concessão do OAuth 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 concessões do OAuth à aplicação, é necessário adicionar um domínio ao 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 fluxo do OAuth, como a concessão implícita, acontece nos navegadores dos 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 prática recomendada de segurança em aplicações de clientes públicos, ative somente o fluxo OAuth de concessão de 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 autorização de API com servidores de recursos.

As concessões de credenciais do cliente adicionam custos à sua AWS fatura. Para mais informações, consulte Preços do Amazon Cognito.

Criar um cliente de aplicação

AWS Management Console

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.
4. Selecione a guia App integration (Integração da aplicação).
5. Em App clients (Clientes da aplicação), selecione Create an app client (Criar um cliente da aplicação).
6. Selecione um App type (Tipo de aplicação): Public client (Cliente público), Confidential client (Cliente confidencial) ou Other (Outro).
7. Insira um App client name (Nome do cliente da aplicação).
8. Selecione Gerar segredo do cliente para que o Amazon Cognito gere um segredo do cliente para você. Normalmente segredos dos clientes são associados a clientes confidenciais.
9. Selecione os Authentication flows (Fluxos de autenticação) que deseja permitir no cliente da aplicação.
10. Configure a Authentication flow session duration (Duração da sessão do fluxo de autenticação). Esse é o tempo que os usuários têm para concluir cada desafio de autenticação antes que o token da sessão expire.
11. (Opcional) Se desejar configurar a validade do token, conclua as etapas a seguir:

    a. Especifique a Refresh token expiration (Validade do token de atualização) para o cliente da aplicação. O valor padrão é de 30 dias. Você pode alterá-la para qualquer valor entre 1 hora e 10 anos.

b. Especifique a Access token expiration (Validade do token de acesso) para o cliente da aplicação. O valor padrão é uma hora. Você pode alterá-la para qualquer valor entre 5 minutos e 24 horas.

c. Especifique ID token expiration (Validade do token de ID) para o cliente da aplicação. O valor padrão é uma hora. Você pode alterá-la para qualquer valor entre 5 minutos e 24 horas.

> **Important**
>
> Se você usar a interface do usuário hospedada e definir o ciclo de vida do token para menos de uma hora, o usuário será capaz de usar tokens com base na duração do cookie de sessão, que atualmente está fixada em uma hora.

12. Escolha se você vai Enable token revocation (Habilitar revogação de token) para esse cliente da aplicação. Isso aumentará o tamanho dos tokens que o Amazon Cognito emite.
13. Selecione se você vai Evitar mensagens de erro que revelem a existência do usuário para esse cliente de aplicativo. O Amazon Cognito responderá a solicitações de acesso para usuários inexistentes com uma mensagem genérica informando que o nome de usuário ou a senha estavam incorretos.
14. Se você quiser usar a interface de usuário hospedada com esse cliente de aplicativo, defina as configurações da interface do usuário hospedada.

    a. Insira um ou mais URLs de retorno de chamada permitidos. Esses são os URLs da web ou do aplicativo para os quais você deseja que o Amazon Cognito redirecione seus usuários depois que eles concluírem a autenticação.

    b. Insira um ou mais URLs de saída permitidos. Esses são os URLs que você deseja que seu aplicativo aceite em solicitações para o Endpoint de logout.

    c. Escolha um ou mais provedores de identidade nos quais você deseja conectar usuários ao seu aplicativo. Você pode escolher qualquer combinação existente IdPs. Você pode autenticar usuários somente com seu grupo de usuários ou com um ou mais terceiros IdPs que você configurou em seu grupo de usuários.

    d. Selecione os tipos de concessão do OAuth 2.0 que você deseja que seu cliente de aplicativo aceite.

    - Selecione Concessão do código de autorização para transmitir códigos para seu aplicativo que ele possa trocar por tokens com o Endpoint de token.

- Selecione Concessão implícita para passar o ID e acessar os tokens diretamente ao seu aplicativo. O fluxo de concessão implícito expõe os tokens diretamente aos seus usuários.
- Selecione Credenciais do cliente para passar tokens de acesso ao seu aplicativo com base em seu conhecimento, não das credenciais do usuário, mas do segredo do cliente. O fluxo de concessão de credenciais do cliente é mutuamente exclusivo com código de autorização e fluxos de concessão implícitos.

e. Selecione os Escopos do OpenID Connect que deseja autorizar para esse cliente de aplicativo. Você pode gerar tokens de acesso somente com o escopo `aws.cognito.signin.user.admin` por meio da API de grupos de usuários. Para escopos adicionais, você deve solicitar seus tokens de acesso ao Endpoint de token.

f. Selecione os escopos personalizados que você deseja autorizar com seu cliente de aplicativo. Os escopos personalizados são usados com mais frequência para autorizar o acesso a APIs de terceiro.

15. Defina as Permissões de leitura e gravação de atributos para esse cliente de aplicativo. Seu cliente de aplicativo pode ter permissão para leitura e gravação de tudo ou um subconjunto limitado do esquema de atributos do seu grupo de usuários.
16. Escolha Criar cliente da aplicação.
17. Anote o Client id (ID do cliente). Isso identificará o cliente da aplicação nas solicitações de cadastro e acesso.

AWS CLI

```
aws cognito-idp create-user-pool-client --user-pool-id MyUserPoolID --client-
name myApp
```

> **Note**
>
> Use o formato JSON para URLs de saída e de retorno de chamada para impedir que a CLI trate-os como arquivos parâmetro remoto:
>
> ```
> --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 conectar 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 de recursos avançados de segurança 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

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 aproveitar dois atributos.

1. Com recursos avançados de segurança, é possível monitorar a atividade do usuário em dispositivos específicos para fins de segurança e análise. Quando os usuários fazem login, a aplicação tem a opção de autenticar cada usuário e o respectivo dispositivo adicionando informações do dispositivo aos logs de atividades.
2. A memorização dos dispositivos também comporta um fluxo de autenticação de dispositivo confiável, no qual os usuários podem optar por fazer login sem MFA pelo período adequado aos requisitos de segurança da aplicação. Quando você quiser solicitar novamente ao usuário que envie um código de MFA, é possível alterar o status memorizado do dispositivo.

   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 na guia Experiência de 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.

Também é possível definir as configurações de dispositivos memorizados para o grupo de usuários em uma solicitação de API CreateUserPool ou UpdateUserPool. Para obter mais informações, consulte a propriedade DeviceConfiguration.

A API de grupos de usuários do Amazon Cognito tem operações adicionais para dispositivos memorizados.

1. ListDevices e AdminListDevices geram uma lista das chaves do dispositivo e dos respectivos metadados para um usuário.
2. GetDevice e AdminGetDevice geram a chave do dispositivo e os metadados de um único dispositivo.
3. UpdateDeviceStatus e AdminUpdateDeviceStatus definem o dispositivo de um usuário como memorizado ou não memorizado.
4. ForgetDevice e AdminForgetDevice removem do perfil o dispositivo confirmado de um usuário.

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 Usar a API de grupos de usuários e endpoints de grupo de usuários do Amazon Cognito.

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 solicitação ConfirmDevice para o Amazon Cognito que confirme o dispositivo do usuário com a chave do dispositivo, um nome amigável, um verificador de senha e um salt. 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 usuário em uma solicitação UpdateDeviceStatus.

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 recursos avançados de segurança para o cliente da aplicação e codificar o rastro 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 usuário com uma solicitação de API InitiateAuth.
2. Responda a todos os desafios de autenticação com RespondToAuthChallenge até 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 de 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 de SRP do usuário em uma solicitação de API ConfirmDevice. 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 o usuário afirmar que quer memorizar o dispositivo, gere uma solicitação de API UpdateDeviceStatus com o token de acesso do usuário, 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 use acionadores do Lambda de desafio de autenticação personalizada, transmita um parâmetro `DEVICE_KEY` na 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 usuário com uma solicitação de API InitiateAuth. 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 API RespondToAuthChallenge, envie um `ChallengeName` de `DEVICE_SRP_AUTH` e parâmetros para `USERNAME`, `DEVICE_KEY` e `SRP_A`.
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 os desafios adicionais até receber os JWTs do usuário.

O pseudocódigo a seguir demonstra como calcular valores para a resposta `DEVICE_PASSWORD_VERIFIER` ao desafio.

```
PASSWORD_CLAIM_SECRET_BLOCK = SECRET_BLOCK
TIMESTAMP = Tue Sep 25 00:09:40 UTC 2018
PASSWORD_CLAIM_SIGNATURE = Base64(SHA256_HMAC(K_USER, DeviceGroupKey + DeviceKey +
 PASSWORD_CLAIM_SECRET_BLOCK + TIMESTAMP))
K_USER = SHA256_HASH(S_USER)
S_USER = (SRP_B - k * gx)(a + ux)
x = SHA256_HASH(salt + FULL_PASSWORD)
u = SHA256_HASH(SRP_A + SRP_B)
k = SHA256_HASH(N + g)
```

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. Os IdPs de terceiros devem gerenciar separadamente os dispositivos e a MFA para os usuários que se autenticam no Amazon Cognito. Na interface de usuário hospedada, é possível solicitar o escopo `aws.cognito.signin.user.admin`, mas ela 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 uma aplicação cliente pública, envie a chave de acesso e a chave do dispositivo do usuário em uma solicitação de API GetDevice.
- Para exibir as informações do dispositivo do usuário em uma aplicação cliente confidencial, assine uma solicitação de API AdminGetDevice com credenciais AWS e envie o nome do 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 uma aplicação cliente pública, envie o token de acesso do usuário em uma solicitação de API ListDevices.
- Em uma aplicação cliente confidencial, assine uma solicitação de API AdminListDevices com credenciais AWS e envie o nome do usuário e o grupo de usuários.

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 uma aplicação cliente pública, envie a chave do dispositivo e o token de acesso do usuário em uma solicitação de API ForgetDevice.

- Em uma aplicação cliente confidencial, envie a chave do dispositivo e o token de acesso do usuário em uma solicitação de API AdminForgetDevice.

Usar a API de grupos de usuários e endpoints de grupo de usuários do Amazon Cognito

Quando quiser se inscrever, fazer login e gerenciar usuários no grupo de usuários, você terá duas opções.

1. Os endpoints do grupo de usuários incluem a interface do usuário hospedada e os endpoints de federação. 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 de inscrição, login, gerenciamento de senhas e autenticação multifator (MFA), use a interface de usuário integrada da interface de usuário hospedada. Os outros endpoints de grupo de usuários facilitam a autenticação com provedores de identidades (IdPs) de terceiros. Os serviços que eles realizam incluem o seguinte:
   a. Endpoints de retorno de chamada do provedor de serviços para solicitações autenticadas de seus IdPs, como `saml2/idpresponse` e `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 `jwks.json`. Sua aplicação usa esses endpoints ao verificar tokens ou recupera dados do perfil do usuário com AWS SDKs e bibliotecas OAuth 2.0.
2. A API de grupos de usuários do Amazon Cognito é um conjunto de ferramentas para aplicação web ou aplicativo móvel depois são coletadas informações de login em seu próprio front-end personalizado para autenticar usuários. 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 fatura da AWS.

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 a identidade federada dele a um usuário local em uma solicitação de API AdminLinkProviderForUser, 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. Ela cria e configura os recursos de grupos de usuários do Amazon Cognito. Por exemplo, você pode criar grupos de usuários, adicionar acionadores do AWS Lambda e configurar o domínio de interface de usuário hospedado. A API de grupos de usuários também realiza operações de inscrição, 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. A aplicação envia uma solicitação da API SignUp e cria um usuário no 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. A aplicação envia uma solicitação da API ConfirmSignUp 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. A aplicação envia uma solicitação da API InitiateAuth e armazena um token de ID, token de acesso e 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 a interface do usuário hospedada, consulte Referência da interface do usuário hospedada e endpoints de federação do grupo de usuários. Os usuários federados podem começar na interface de usuário hospedada e selecionar o IdP deles ou você pode ignorar a interface de usuário hospedada 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 endpoints de grupo de usuários

1. O usuário seleciona o botão “Criar uma conta” que você criou na aplicação.
2. Você apresenta ao usuário uma lista dos provedores de identidade 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 autorização do Apple OAuth. O usuário opta por permitir 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 aplicação troca o código de autorização pelo Endpoint de token e armazena um token de ID, token de acesso e token de atualização. A aplicação chama as bibliotecas do OIDC para gerenciar os tokens do usuário e manter uma sessão persistente para esse usuário.

A API e os endpoints de grupo de usuários 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.

Operações de API autenticadas e não autenticadas de grupos de usuários do Amazon Cognito

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 autenticadas pelo IAM e Operações de usuário autenticadas pelo IAM.

Operações de gerenciamento autenticadas pelo IAM

As operações de gerenciamento autenticadas pelo IAM modificam e exibem o grupo de usuários e a configuração do cliente da aplicação, como você faria no AWS Management Console.

Por exemplo, para modificar o grupo de usuários em uma solicitação da API UpdateUserPool, você deve apresentar credenciais da AWS e permissões do IAM para atualizar o recurso.

Para autorizar essas solicitações na AWS Command Line Interface (AWS CLI) ou em um AWS SDK, configure o ambiente com variáveis de ambiente ou configuração de cliente que adicione credenciais do IAM à solicitação. Para ter mais informações, consulte Acessar a AWS usando suas credenciais da AWS na 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 credenciais da AWS que incorpora no cabeçalho da solicitação. Para ter mais informações, consulte Assinar solicitações de API da AWS.

Operações de gerenciamento autenticadas pelo IAM

`AddCustomAttributes`

`CreateGroup`

`CreateIdentityProvider`

Operações de gerenciamento autenticadas pelo IAM

`CreateResourceServer`

`CreateUserImportJob`

`CreateUserPool`

`CreateUserPoolClient`

`CreateUserPoolDomain`

`DeleteGroup`

`DeleteIdentityProvider`

`DeleteResourceServer`

`DeleteUserPool`

`DeleteUserPoolClient`

`DeleteUserPoolDomain`

`DescribeIdentityProvider`

`DescribeResourceServer`

`DescribeRiskConfiguration`

`DescribeUserImportJob`

`DescribeUserPool`

`DescribeUserPoolClient`

`DescribeUserPoolDomain`

`GetCSVHeader`

`GetGroup`

`GetIdentityProviderByIdentifier`

Operações de gerenciamento autenticadas pelo IAM

`GetSigningCertificate`

`GetUICustomization`

`GetUserPoolMfaConfig`

`ListGroups`

`ListIdentityProviders`

`ListResourceServers`

`ListTagsForResource`

`ListUserImportJobs`

`ListUserPoolClients`

`ListUserPools`

`ListUsers`

`ListUsersInGroup`

`SetRiskConfiguration`

`SetUICustomization`

`SetUserPoolMfaConfig`

`StartUserImportJob`

`StopUserImportJob`

`TagResource`

`UntagResource`

`UpdateGroup`

`UpdateIdentityProvider`

Operações de gerenciamento autenticadas pelo IAM

`UpdateResourceServer`

`UpdateUserPool`

`UpdateUserPoolClient`

`UpdateUserPoolDomain`

Operações de usuário autenticadas pelo IAM

Operações de usuário autenticadas pelo IAM para se inscrever, fazer login, gerenciar credenciais, modificar e ver os usuários.

Por exemplo, você pode ter um nível de aplicação do lado do servidor que oferece suporte a um front-end da Web. A aplicação do lado do servidor é um cliente confidencial do OAuth no qual você confia, com acesso privilegiado aos seus recursos do Amazon Cognito. Para registrar um usuário na aplicação, o servidor pode incluir credenciais da AWS em uma solicitação da API AdminCreateUser. Para obter mais informações sobre os tipos de clientes do OAuth, consulte Client Types (Tipos de cliente) na Estrutura de autorização do OAuth 2.0.

Para autorizar essas solicitações na AWS CLI ou em um AWS SDK, configure o ambiente da aplicação do lado do servidor com variáveis de ambiente ou configuração do cliente que adiciona credenciais do IAM à solicitação. Para ter mais informações, consulte Acessar a AWS usando suas credenciais da AWS na 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 credenciais da AWS que incorpora no cabeçalho da solicitação. Para ter mais informações, consulte Assinar solicitações de API da AWS.

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 autenticadas pelo IAM

`AdminAddUserToGroup`

`AdminConfirmSignUp`

Operações de usuário autenticadas pelo IAM

AdminCreateUser

AdminDeleteUser

AdminDeleteUserAttributes

AdminDisableProviderForUser

AdminDisableUser

AdminEnableUser

AdminForgetDevice

AdminGetDevice

AdminGetUser

AdminInitiateAuth

AdminLinkProviderForUser

AdminListDevices

AdminListGroupsForUser

AdminListUserAuthEvents

AdminRemoveUserFromGroup

AdminResetUserPassword

AdminRespondToAuthChallenge

AdminSetUserMFAPreference

AdminSetUserPassword

AdminSetUserSettings

AdminUpdateAuthEventFeedback

Operações de usuário autenticadas pelo IAM
`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 na aplicação, você pode distribuir um cliente público do OAuth que não fornece nenhum acesso privilegiado aos segredos. É possível 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`
`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, o cliente pode incluir o token de acesso do usuário em uma solicitação da API UpdateUserAttributes.

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`		✓
`ChangePassword`	✓	
`GetUser`	✓	
`UpdateUserAttributes`	✓	

Operações de usuário autorizadas por token	AccessToken	Sessão
DeleteUserAttributes	✓	
DeleteUser	✓	
ConfirmDevice	✓	
ForgetDevice	✓	
GetDevice	✓	
ListDevices	✓	
UpdateDeviceStatus	✓	
GetUserAttributeVerificationCode	✓	
VerifyUserAttribute	✓	
SetUserSettings	✓	
SetUserMFAPreference	✓	
GlobalSignOut	✓	
AssociateSoftwareToken	✓	✓
UpdateAuthEventFeedback		✓
VerifySoftwareToken	✓	✓

Operações de usuário autorizadas por token	AccessTok en	Sessão

`RevokeToken` [1]

[1] RevokeToken usa um token de atualização como parâmetro. O token de atualização serve como token de autorização e como recurso de destino.

Atualizar a configuração do grupo de usuários

Para alterar as configurações dos grupos de usuários do Amazon Cognito no AWS Management Console, navegue pelas guias baseadas em recursos nas configurações do grupo de usuários e atualize os campos conforme descrito em outras áreas deste guia. 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.

Nome do grupo de usuários

Nome do parâmetro da API: PoolName

O nome amigável que você atribuiu ao seu grupo de usuários. Para alterar o nome de um grupo de usuários, crie outro grupo de usuários.

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 ter 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 ter mais informações, consulte Clientes de aplicações de grupos de usuários. Para alterar um segredo de cliente, crie outro cliente da aplicação no mesmo grupo de usuários.

Atributos obrigatórios

Nome do parâmetro da API: Esquema

Os atributos aos quais seus usuários devem fornecer valores no cadastro ou quando você os cria. Para ter mais informações, consulte Atributos de grupo de usuários. Para alterar os atributos necessários, crie outro grupo de usuários.

Atributos personalizados

Nome do parâmetro da API: Esquema

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 ter mais informações, consulte Atributos de grupo de usuários. 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 as mensagens SMS em seu grupo de usuários, você não poderá desativá-las.

- Se você optar por configurar mensagens SMS ao criar um grupo de usuários, não poderá desativar o SMS depois de concluir a configuração.
- Você pode ativar mensagens SMS em um grupo de usuários que você criou, mas depois disso você não pode desativar o SMS.
- O Amazon Cognito pode usar mensagens SMS para convite e recuperação de contas de usuários, verificação de atributos e autenticação multifatorial (MFA). Depois de ativar as mensagens SMS, você pode ativar ou desativar as mensagens SMS para essas funções a qualquer momento.
- A configuração de mensagens SMS inclui uma função do IAM que você delega ao Amazon Cognito para enviar mensagens com o Amazon SNS. Você pode alterar a função atribuída a qualquer momento.

Atualização de um grupo de usuários com um AWS SDK ou AWS CDK API REST

No console do Amazon Cognito, você pode alterar as configurações do grupo de usuários, um parâmetro por vez. Por exemplo, para adicionar um gatilho Lambda, você escolhe Adicionar gatilho Lambda e escolhe a função e o tipo de gatilho. 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 aplicativos 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 suas configurações até corrigi-lo. Ao encontrar esse erro, examine a resposta do erro e valide a configuração mencionada.

A AWS Cloud Development Kit (AWS CDK)API REST e os AWS SDKs dos grupos de usuários do Amazon Cognito são ferramentas para automação e configuração programática dos recursos do Amazon Cognito. As solicitações com essas ferramentas também devem, como o console do Amazon Cognito, atualizar uma configuração com uma configuração completa de recursos no corpo da solicitação. Em um alto nível, você deve executar o seguinte processo.

1. Capture a saída de uma operação que descreve a configuração do seu recurso existente.
2. Modifique a saída 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
```

Configurar e usar a interface de usuário hospedada e endpoints de federação do Amazon Cognito

Um grupo de usuários do Amazon Cognito com um domínio é um servidor de autorização compatível com OAuth-2.0 e uma interface de usuário (UI) ready-to-use hospedada para autenticação. O servidor de autorização direciona solicitações de autenticação, emite e gerencia tokens web JSON (JWTs) e fornece informações de atributos do usuário. A interface de usuário hospedada é uma coleção de interfaces da web para atividades básicas de inscrição, login, autenticação multifator e redefinição de senha no grupo de usuários. Também é um hub central para autenticação com os provedores de identidade terceirizados (IdPs) que você associa ao seu aplicativo. A aplicação pode invocar a interface de usuário hospedada e os endpoints de autorização quando você quiser autenticar e autorizar usuários. Você pode fazer com que a experiência do usuário da interface hospedada se adapte à sua marca com seu próprio logotipo e personalização de CSS. Para receber mais informações sobre os componentes da interface de usuário hospedada e do servidor de autorização, consulte Referência da interface do usuário hospedada e endpoints de federação do grupo de usuários.

> **Note**
>
> A interface de usuário hospedada do Amazon Cognito não é compatível com a autenticação personalizada com gatilhos do Lambda de desafio de autenticação personalizada.

Tópicos

- Configurando a interface de usuário hospedada com AWS Amplify
- Como configurar a interface do usuário hospedada com o console do Amazon Cognito

- Visualizar a página de login
- Fatos a saber sobre a interface de usuário hospedada dos grupos de usuários do Amazon Cognito
- Como configurar um domínio de grupo de usuários
- Como personalizar as páginas da Web integradas de cadastro e acesso
- Cadastrar-se e fazer login com a UI hospedada

Configurando a interface de usuário hospedada com AWS Amplify

Se você usa AWS Amplify para adicionar autenticação ao seu aplicativo web ou móvel, você pode configurar sua interface de usuário hospedada usando a interface de linha de comando (CLI) e as bibliotecas na AWS Amplify estrutura. Para adicionar autenticação à sua aplicação, use a CLI do AWS Amplify para adicionar a categoria `Auth` ao seu projeto. Em seguida, em seu código de cliente, você usa as AWS Amplify bibliotecas para autenticar usuários com seu grupo de usuários do Amazon Cognito.

Você pode exibir uma interface do usuário hospedada pré-compilada ou federar usuários por meio de um endpoint OAuth 2.0, que faz o redirecionamento para um provedor de login social, como o Facebook, o Google, a Amazon ou a Apple. Depois que um usuário se autentica com êxito com o provedor social, o AWS Amplify criará um novo usuário em seu grupo de usuários, se necessário, e fornecerá o token OIDC do usuário à aplicação.

Os exemplos a seguir mostram como usar AWS Amplify para configurar a interface de usuário hospedada com provedores sociais em seu aplicativo.

- AWS Amplify autenticação para JavaScript.
- AWS Amplify autenticação para Swift.
- AWS Amplify autenticação para Flutter.
- AWS Amplify autenticação para Android.

Como configurar a interface do usuário hospedada com o console do Amazon Cognito

Criar um cliente da aplicação

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 a guia App integration (Integração da aplicação).
5. Em App clients (Clientes da aplicação), selecione Create an app client (Criar um cliente da aplicação).
6. Selecione um App type (Tipo de aplicação): Public client (Cliente público), Confidential client (Cliente confidencial) ou Other (Outro). Normalmente um Public client (Cliente público) opera a partir dos dispositivos de seus usuários e usa APIs não autenticadas e autenticadas com token. Um cliente confidencial normalmente opera a partir de um aplicativo em um servidor central no qual você confia segredos do cliente e credenciais de API, e usa cabeçalhos e AWS Identity and Access Management credenciais de autorização para assinar solicitações. Se o seu caso de uso for diferente das configurações predefinidas do cliente da aplicação para um Public client (Cliente público) ou um Confidential client (Cliente confidencial), selecione Other (Outro).
7. Insira um App client name (Nome do cliente da aplicação).
8. Selecione os Authentication flows (Fluxos de autenticação) que deseja permitir no cliente da aplicação.
9. Configure a Authentication flow session duration (Duração da sessão do fluxo de autenticação). Esse é o tempo que os usuários têm para concluir cada desafio de autenticação antes que o token da sessão expire.
10. (Opcional) Configure a validade do token.

    a. Especifique a Refresh token expiration (Validade do token de atualização) para o cliente da aplicação. O valor padrão é de 30 dias. Você pode alterá-la para qualquer valor entre 1 hora e 10 anos.

    b. Especifique a Access token expiration (Validade do token de acesso) para o cliente da aplicação. O valor padrão é uma hora. Você pode alterá-la para qualquer valor entre 5 minutos e 24 horas.

    c. Especifique ID token expiration (Validade do token de ID) para o cliente da aplicação. O valor padrão é uma hora. Você pode alterá-la para qualquer valor entre 5 minutos e 24 horas.

    > **Important**
    >
    > Se você usar a interface do usuário hospedada e definir o ciclo de vida do token para menos de uma hora, o usuário será capaz de usar tokens com base na duração do cookie de sessão, que atualmente está fixada em uma hora.

11. Selecione Generate client secret (Gerar segredo do cliente) para que o Amazon Cognito gere um segredo do cliente para você. Normalmente segredos dos clientes são associados a clientes confidenciais.
12. Escolha se você vai Enable token revocation (Habilitar revogação de token) para esse cliente da aplicação. Isso aumentará o tamanho dos tokens. Para mais informações, consulte Revoking Tokens (Como revogar tokens).
13. Escolha se você vai Prevent error messages that reveal user existence (Evitar mensagens de erro que revelem a existência do usuário) para esse cliente da aplicação. O Amazon Cognito responderá a solicitações de acesso para usuários inexistentes com uma mensagem genérica informando que o nome de usuário ou a senha estavam incorretos.
14. (Opcional) Defina as Attribute read and write permissions (Permissões de leitura e gravação de atributos) para esse cliente da aplicação. Seu cliente da aplicação pode ter permissão para leitura e gravação de um subconjunto limitado do esquema de atributos do seu grupo de usuários.
15. Escolha Create (Criar).
16. Anote o Client id (ID do cliente). Isso identificará o cliente da aplicação nas solicitações de cadastro e acesso.

Configurar a aplicação

1. Na guia App integration (Integração da aplicação), selecione o cliente da aplicação em App clients (Clientes da aplicação). Revise suas informações atuais sobre a Hosted UI (interface do usuário hospedada).
2. Add a callback URL (Adicionar um URL de retorno de chamada) em Allowed callback URL(s) (URL(s) de retorno de chamada permitidos). Um URL de retorno de chamada é para onde o usuário será redirecionado após realizar o acesso com êxito.
3. Add a sign-out URL (Adicionar um URL de saída) em Allowed sign-out URL(s) (URL(s) de saída permitidos). Um URL de saída é para onde o usuário será redirecionado após sair.
4. Adicione pelo menos uma das opções listadas da lista de Identity providers (Provedores de identidade).
5. Em OAuth 2.0 grant types (Tipos de concessões OAuth 2.0), selecione Authorization code grant (Concessão de código de autorização) para retornar um código de autorização que é trocado por tokens do grupo de usuários. Como os tokens nunca são expostos diretamente a um usuário final, é menos provável que eles fiquem comprometidos. No entanto, uma aplicação personalizada é necessário no backend para trocar o código de autorização por tokens do grupo

de usuários. Por motivos de segurança, recomendamos que você use o fluxo de concessão de código de autorização juntamente com o Proof key for code Exchange (PKCE) para aplicativos móveis.

6. Em OAuth 2.0 grant types (Tipos de concessão OAuth 2.0), selecione Implicit grant (Concessão implícita) para que os JSON Web Tokens (JWT) do grupo de usuários sejam retornados para você do Amazon Cognito. Você pode usar esse fluxo quando não houver backend disponível para trocar um código de autorização por tokens. Ele também é útil para depurar tokens.
7. É possível habilitar tanto as concessões de Authorization code (Código de autorização) quanto de Implicit code (Código implícito) e, em seguida, usar cada concessão conforme necessário. Se nenhuma das concessões Authorization code (Código de autorização) ou Implicit code (Código implícito) forem selecionadas e seu cliente da aplicação tiver um segredo de cliente, você pode habilitar concessões de Client credentials (Credenciais do cliente). Só selecione Client credentials (Credenciais do cliente) se a aplicação precisar solicitar tokens de acesso em nome dela mesma, e não em nome de um usuário.
8. Selecione os OpenID Connect scopes (Escopos do OpenID Connect) que deseja autorizar para esse cliente da aplicação.
9. Escolha Salvar alterações.

Configurar um domínio

1. Acesse a guia App integration (Integração da aplicação) para o seu grupo de usuários.
2. Ao lado de Domain (Domínio), escolha Action (Ações) e, em seguida, escolha Create custom domain (Criar domínio personalizado) ou Create Cognito domain (Criar domínio do Cognito). Se já tiver configurado um domínio de grupo de usuários, escolha Delete Cognito domain (Excluir domínio do Cognito) ou Delete custom domain (Excluir domínio personalizado) antes de criar seu novo domínio personalizado.
3. Insira um prefixo de domínio disponível para usar com um Cognito domain (Domínio do Cognito). Para informações sobre como configurar um Domínio personalizado, consulte Uso do próprio domínio para a interface do usuário hospedada
4. Selecione Create (Criar).

Visualizar a página de login

No console do Amazon Cognito, selecione o botão View Hosted UI (Visualizar UI hospedada) na configuração do cliente da aplicação, em App clients and analytics (Clientes e análise de aplicações),

na guia App integration (Integração de aplicações). Esse botão levará você a uma página de login na UI hospedada 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 View hosted UI (Visualizar UI hospedada) é útil quando você deseja testar as funções básicas da UI hospedada. Você pode personalizar o URL de login com parâmetros adicionais e modificados. Na maioria dos casos, os parâmetros gerados automaticamente do link View hosted UI (Visualizar UI hospedada) 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 da interface do usuário hospedada e endpoints de federação do grupo de usuários.

A página da web de acesso da UI hospedada 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 callback url>
```

É possível recuperar o domínio do grupo de usuários pela guia Integração de aplicações. Na mesma guia, é possível identificar IDs de clientes de aplicação, seus URLs de retorno de chamada, seus escopos permitidos e outras configurações em Análise e clientes de aplicação.

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. Para ver um exemplo de URL que ignora a UI hospedada, consulte Iniciação de sessão SAML em grupos de usuários do Amazon Cognito.

Exemplo de solicitação de interface de usuário hospedada para uma concessão implícita

Você pode visualizar a página da web de login da interface do usuário hospedada 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.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://mydomain.example.com/
#id_token=eyJraaBcDeF1234567890&access_token=eyJraGhIjKlM1112131415&expires_in=3600&token_type=
```

Fatos a saber sobre a interface de usuário hospedada dos grupos de usuários do Amazon Cognito

A interface de usuário hospedada e a confirmação de usuários como administradores

Para usuários locais do grupo de usuários, a interface de usuário hospedada funciona 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. Em vez disso, quando você confirma os usuários como administradores do grupo de usuários, a interface de usuário hospedada exibe uma mensagem de erro após a inscrição. 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.

Visualizar as alterações na configuração da interface de usuário hospedada

Se as alterações nas páginas da sua interface do usuário hospedada não aparecerem imediatamente, aguarde alguns minutos e atualize a página.

Decodificar tokens do grupo de usuários

Os tokens de grupo de usuários do Amazon Cognito são assinados usando um algoritmo RS256. 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

A interface hospedada e a versão TLS

A interface de usuário hospedada requer criptografia em trânsito. Os domínios de grupos de usuários fornecidos pelo Amazon Cognito exigem uma versão mínima de TLS 1.2. Os domínios personalizados oferecem suporte, mas não exigem a versão 1.2 do TLS. Como o Amazon Cognito gerencia a configuração da interface do usuário hospedada e dos endpoints do servidor de autorização, você não pode modificar os requisitos de TLS do seu domínio do grupo de usuários.

As políticas de CORS e interface de usuário hospedada

A UI hospedada do Amazon Cognito não comporta políticas de origem de CORS. Uma política de CORS na UI hospedada impede que os usuários transmitam parâmetros de autenticação em suas solicitações. Em vez disso, implemente uma política de CORS no front-end da web da aplicação. O Amazon Cognito retorna um cabeçalho de resposta `Access-Control-Allow-Origin: *` às solicitações para os endpoints do OAuth a seguir.

1. Endpoint de token
2. Revogar endpoint
3. Endpoint do UserInfo

UI hospedada e cookies do servidor de autorização

Os endpoints do pool de usuários do Amazon Cognito definem cookies nos navegadores dos usuários. Os cookies estão em conformidade com 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 `XSRF-TOKEN` cookie para cada solicitação.
- Um `csrf-state` cookie para consistência da sessão quando um usuário é redirecionado.
- Um cookie de `cognito` sessão que preserva as tentativas de login bem-sucedidas por uma hora.

Como configurar um domínio de grupo de usuários

Após configurar um cliente do aplicativo, você pode configurar o endereço do seu cadastro e login em páginas da Web. Você pode usar um domínio hospedado do Amazon Cognito e escolher um prefixo de domínio disponível ou você pode usar seu próprio endereço da Web como domínio personalizado.

Para adicionar uma aplicação cliente e um domínio hospedado do Amazon Cognito com o AWS Management Console, consulte Adicionar uma aplicação para habilitar a interface do usuário da Web hospedada.

> **Note**
>
> Não é possível usar o texto `aws`, `amazon` ou `cognito` no prefixo do domínio.

Tópicos

Como usar o domínio do Amazon Cognito para a interface do usuário hospedada

Após configurar um cliente da aplicação, é possível configurar o endereço para suas páginas da Web de cadastro e acesso. Você pode usar o domínio hospedado do Amazon Cognito com seu próprio prefixo de domínio.

> **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 endpoint do grupo de usuários usam os seguintes formatos.
>
> ```
> auth.Region.amazoncognito.com
> auth-fips.Region.amazoncognito.com
> ```

Para adicionar um cliente de aplicativo e um domínio hospedado no Amazon Cognito com o AWS Management Console, consulte. Criar um cliente de aplicação

Tópicos

Pré-requisitos

Antes de começar, você precisa de:

- Um grupo de usuários com um cliente de aplicativo. Para ter mais informações, consulte Conceitos básicos dos grupos de usuários.

Etapa 1: configurar um domínio hospedado de grupo de usuários

Para configurar um domínio hospedado de grupo de usuários

Você pode usar a API AWS Management Console ou a AWS CLI ou para configurar um domínio de grupo de usuários.

Amazon Cognito console

Configurar um domínio

1. Acesse a guia App integration (Integração da aplicação) para o seu grupo de usuários.
2. Ao lado de Domínio, escolha Ações e Criar domínio personalizado ou Criar domínio do Amazon Cognito. Se já tiver configurado um domínio de grupo de usuários, escolha Excluir domínio do Amazon Cognito ou Excluir domínio personalizado antes de criar o domínio personalizado.
3. Insira um prefixo de domínio disponível para usar com um Domínio do Amazon Cognito. Para informações sobre como configurar um Domínio personalizado, consulte Uso do próprio domínio para a interface do usuário hospedada
4. Selecione Create (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>`
- AWS API: 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>`
- AWS API: DescribeUserPoolDomain

Para excluir um domínio

- AWS CLI: `aws cognito-idp delete-user-pool-domain`

  Exemplo: `aws cognito-idp delete-user-pool-domain --domain <domain_name>`
- AWS API: DeleteUserPoolDomain

Etapa 2: verificar a página de acesso

- 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).

Como usar o próprio domínio para a interface do usuário hospedada

Depois de configurar um cliente de aplicação, você poderá configurar o grupo de usuários com um domínio personalizado para a interface do usuário hospedada e endpoints de API de autenticação do Amazon Cognito. Com um domínio personalizado, você permite que os usuários façam login no aplicativo usando o seu próprio endereço da web.

Tópicos

- Como adicionar um domínio personalizado a um grupo de usuários
- Como alterar o certificado SSL do seu domínio personalizado

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 ter 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. Com o objetivo de evitar um impacto acidental na infraestrutura do cliente, o Amazon Cognito não aceita o uso de domínios de nível superior (TLDs) para domínios personalizados. Para obter mais informações, consulte Nomes de domínio.
- A capacidade de criar um subdomínio para seu domínio personalizado. Recomendamos que você use auth como o 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 certificado Secure Sockets Layer (SSL) gerenciado pelo ACM.

> **Note**
>
> Você deve alterar a AWS região para Leste dos EUA (Norte da Virgínia) no console do ACM antes de solicitar ou importar um certificado.

- Um aplicativo que permite que seu servidor de autorização do grupo de usuários adicione cookies às sessões do usuário. O Amazon Cognito define vários cookies necessários para a interface hospedada. Entre eles estão `cognito`, `cognito-fl` e `XSRF-TOKEN`. Embora cada cookie individual esteja em conformidade com os limites de tamanho do navegador, alterações na configuração do grupo de usuários podem fazer com que os cookies de interface do usuário hospedados aumentem de tamanho. Um serviço intermediário, como um Application Load Balancer (ALB) na frente do seu domínio personalizado, pode impor um tamanho máximo de cabeçalho ou tamanho total de cookie. Se seu aplicativo também definir seus próprios cookies, as sessões de seus usuários poderão exceder esses limites. Recomendamos que, para evitar conflitos de limite de tamanho, seu aplicativo não defina cookies no subdomínio da interface de usuário hospedada.
- 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:

```
{
    "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. Faça login no console do Amazon Cognito. Se solicitado, insira suas credenciais da AWS .
2. Escolha User Pools (Grupos de usuários).
3. Escolha o grupo de usuários ao qual você deseja adicionar seu domínio.
4. Escolha a guia App integration (Integração da aplicação).
5. Ao lado de Domain (Domínio), escolha Actions (Ações) e, em seguida, escolha Create custom domain (Criar domínio personalizado).

   > **Note**
   >
   > Se você já configurou um domínio de grupo de usuários, escolha Delete Cognito domain (Excluir domínio do Cognito) ou Delete custom domain (Excluir domínio personalizado) para excluir o domínio existente antes de criar seu novo domínio personalizado.

6. 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.

7. 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 .
8. Selecione Create (Criar).
9. O Amazon Cognito retorna você à guia App integration (Integração da aplicação). 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

Para adicionar o domínio ao seu grupo de usuários com a API do Amazon Cognito:

- Use a ação CreateUserPoolDomain.

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 credenciais da AWS .
2. Se você não tiver uma zona hospedada no Route 53, crie uma com uma raiz que seja mãe do seu domínio personalizado. Para obter mais informações, consulte

   a. Escolha Criar hosted zone.

   b. Insira o domínio pai, por exemplo *auth.example.com*, do seu domínio personalizado, por exemplo *myapp.auth.example.com*, da lista Domain Name (Nome do 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 Create hosted zone (Criar zona hospedada).

   > **Note**
   >
   > Também é possível criar uma nova zona hospedada para o domínio personalizado e criar 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 para o domínio principal do seu domínio personalizado, se você ainda não tiver um. Adicione um `A` registro DNS para o domínio principal e escolha Criar registros. Veja a seguir um exemplo de registro para o domínio *auth.example.com*.

   ```
   auth.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 Start of Authority (SOA) não é um registro DNS suficiente para fins de verificação do domínio principal.

5. Adicione um registro DNS para seu domínio personalizado. Seu registro deve apontar para o destino do alias de domínio personalizado, por exemplo`123example.cloudfront.net`. Selecione Create record (Criar registro) novamente.
6. Insira um Record name (Nome de registro) que corresponda ao seu domínio personalizado, por exemplo *`myapp`*, para criar um registro para *`myapp.auth.example.com`*.
7. Habilite a opção Alias (Alias).
8. Selecione Route traffic to (Rotear tráfego para) um Alias to CloudFront distribution (Alias para distribuição do CloudFront). Insira o Alias target (Destino do alias) fornecido pelo Amazon Cognito quando você criou seu domínio personalizado.
9. Selecione Create records (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.

AWS Management Console

Para renovar um certificado no console do Amazon Cognito:

1. Faça login no AWS Management Console 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. Escolha a guia App integration (Integração da aplicação).
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.

Como personalizar as páginas da Web integradas de cadastro e acesso

Você pode usar o AWS Management Console a AWS CLI ou a API para especificar as configurações de personalização para a experiência de interface do usuário incorporada do aplicativo. É possível fazer upload de uma imagem de logo personalizada para exibição no aplicativo. Também é possível usar Cascading Style Sheets (CSS – Folhas de estilos em cascata) para personalizar a aparência da interface do usuário.

É possível especificar configurações de personalização da interface do usuário da aplicação para um único cliente (com um `clientId` específico) ou para todos os clientes (definindo o `clientId` para `ALL`). Se você especificar `ALL`, a configuração padrão será usada para todos os clientes que nunca tiveram personalizações da interface do usuário definidas anteriormente. Se você especificar configurações de personalização da interface do usuário para um cliente específico, ele deixará de se enquadrar na configuração `ALL`.

A solicitação que define a personalização da interface do usuário não deve exceder 135 KB de tamanho. 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, o `SetUICustomization` ou a solicitação da API `request parameters too large` retornará um erro AWS Management Console. 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.

Como especificar um logo personalizado para a aplicação

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` em SetUICustomization na API, você pode converter o arquivo em uma string de texto codificada em Base64 ou, na AWS CLI, fornecer um caminho de arquivo e deixar que o Amazon Cognito o codifique para você.

Como especificar personalizações de CSS para a aplicação

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.

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.

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.

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.

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.

Como especificar as configurações de personalização de interface do usuário da aplicação para um grupo de usuários (AWS Management Console)

Você pode usar o AWS Management Console para especificar as configurações de personalização da interface do usuário para o 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. Escolha a guia App integration (Integração da aplicação).
4. Para personalizar as configurações da interface do usuário para todos os clientes da aplicação, localize Hosted UI customization (Personalização da interface do usuário hospedada) e selecione Edit (Editar).
5. Para personalizar as configurações da interface do usuário para um único cliente da aplicação, localize Clientes da aplicação e selecione o cliente da aplicação que deseja modificar. Depois, localize Personalização da interface do usuário hospedada e selecione Editar. Para trocar um cliente da aplicação da personalização padrão do grupo de usuários para uma personalização específica do cliente, selecione Use client-level settings (Usar configurações por cliente).
6. Para carregar seu próprio arquivo de imagem de logo, escolha Choose file (Escolher arquivo) ou Replace current file (Substituir arquivo atual).
7. 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.

Como especificar as configurações de personalização de interface do usuário da aplicação para um grupo de usuários (AWS CLI e API da AWS)

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`
- API da AWS: 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 em 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>;}"`
- API da AWS: SetUICustomization

Cadastrar-se e fazer login com a UI hospedada

Depois de configurar e personalizar a UI hospedada do Amazon Cognito para o grupo de usuários e clientes da aplicação, a aplicação poderá apresentá-la aos usuários. A UI hospedada comporta várias operações de autenticação do Amazon Cognito, inclusive os exemplos a seguir.

- Cadastrar-se como um novo usuário na aplicação
- Verificar um endereço de e-mail ou um número de telefone
- Configurar a autenticação multifator (MFA)
- Faça login com um nome de usuário local e senha.
- Fazer login com um provedor de identidades (IdP) de terceiros
- Redefinição de uma senha

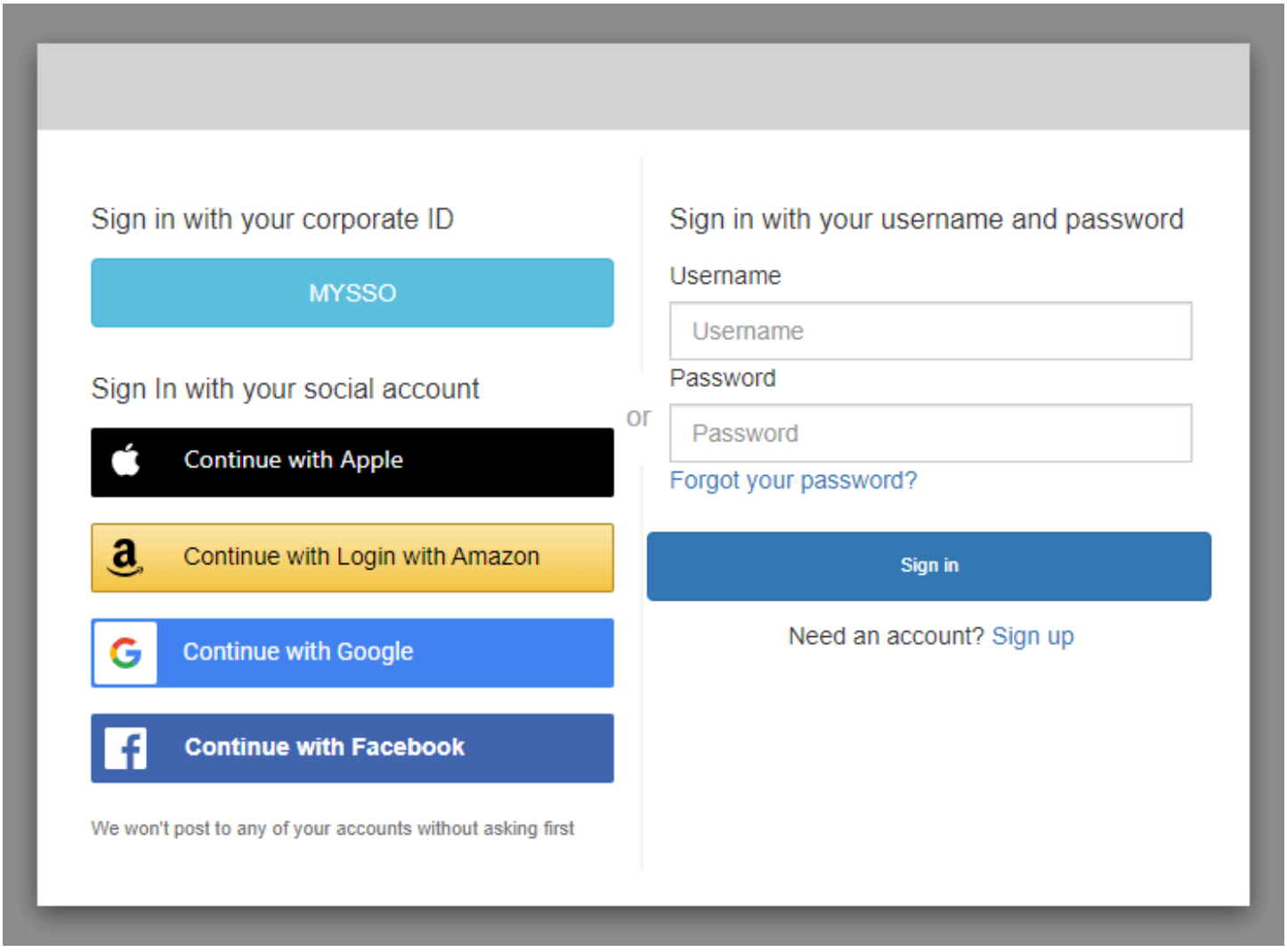
A UI hospedada do Amazon Cognito começa no Endpoint de login. O URL da página de login é uma combinação do domínio selecionado para o grupo de usuários e parâmetros que refletem as concessões do OAuth 2.0 que você deseja emitir, o cliente da aplicação, o caminho para a aplicação e os escopos do OpenID Connect (OIDC) que deseja solicitar.

```
https://<your user pool domain>/authorize?client_id=<your app client
 ID>&response_type=<code/token>&scope=<scopes to request>&redirect_uri=<your callback
 URL>
```

O URL a seguir substitui os campos de espaço reservado acima por valores de exemplo.

```
https://auth.example.com/authorize? /
client_id=1example23456789 /
&response_type=code /
&scope=aws.cognito.signin.user.admin+email+openid+profile /
&redirect_uri=https%3A%2F%2Faws.amazon.com
```

A página de login da UI hospedada do Amazon Cognito tem opções para fazer login por meio do grupo de usuários ou de qualquer provedor de identidades (IdP) que você atribuiu ao cliente da aplicação solicitado pelo usuário. Também inclui links para cadastro em uma nova conta de usuário no grupo de usuários ou para redefinição de uma senha esquecida.



Tópicos

- Como se cadastrar em uma nova conta na UI hospedada do Amazon Cognito
- Como fazer login com a UI hospedada do Amazon Cognito

- Como redefinir uma senha com a UI hospedada no Amazon Cognito

Como se cadastrar em uma nova conta na UI hospedada do Amazon Cognito

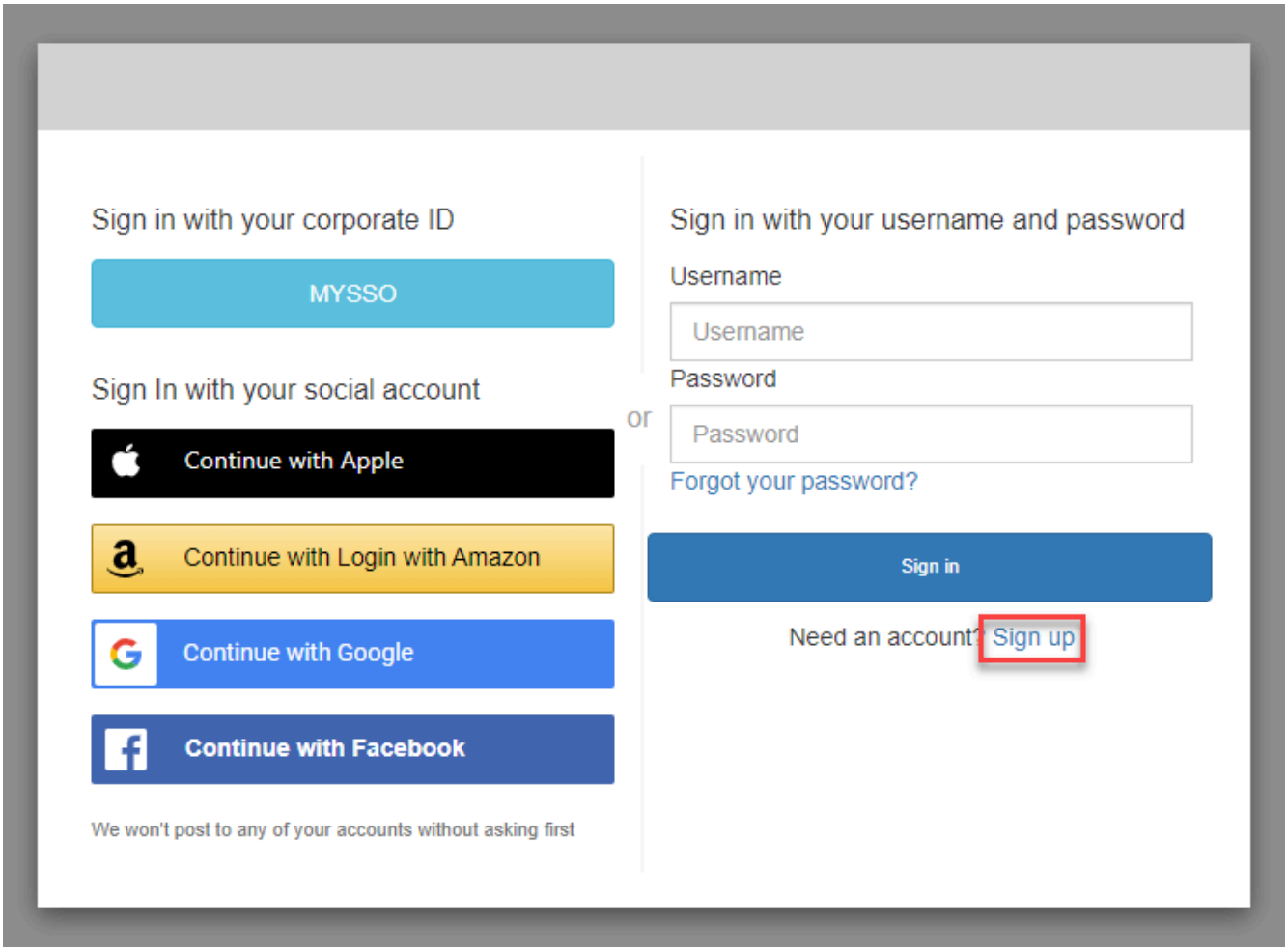
Este guia mostra como se cadastrar em uma conta de usuário em aplicações que usam o Amazon Cognito.

> **Note**
>
> Ao fazer login em uma aplicação que usa a interface do usuário (UI) hospedada do Amazon Cognito, você pode ver uma página que o proprietário da aplicação personalizou, além da configuração básica mostrada neste guia.

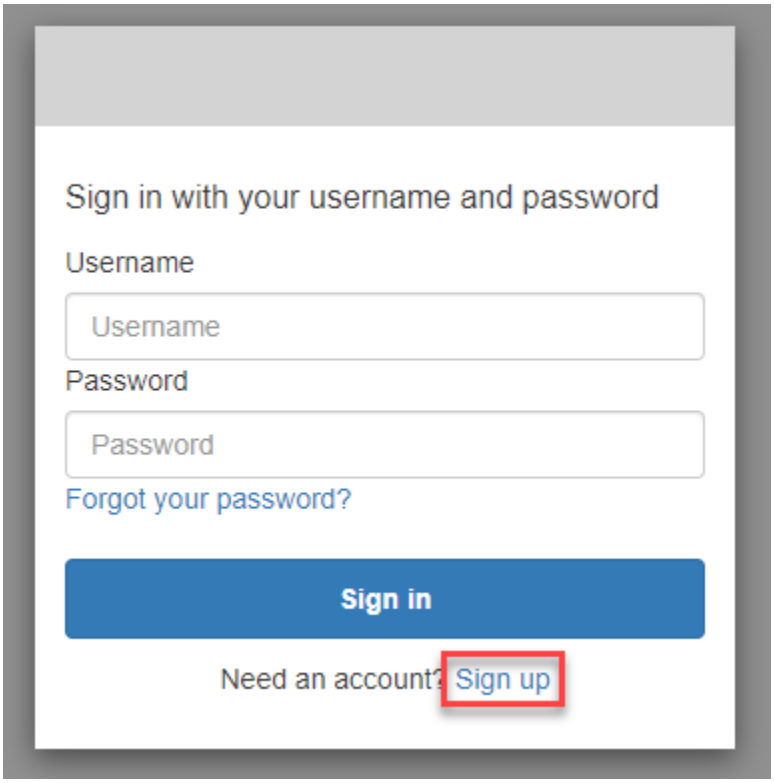
1. Selecione Sign up (Cadastrar-se) na página de login se você pretende fazer login pelo Amazon Cognito com um nome de usuário e senha, em vez de um dos provedores de login de terceiros listados pelo proprietário da aplicação.

   Se o seu provedor de login não for o Amazon Cognito, seu cadastro será concluído depois que você escolher o botão correspondente ao provedor de terceiros. Dependendo das opções selecionadas pelo proprietário da aplicação, você pode selecionar entre os provedores com os quais fazer login ou ver apenas uma solicitação de nome de usuário e senha.

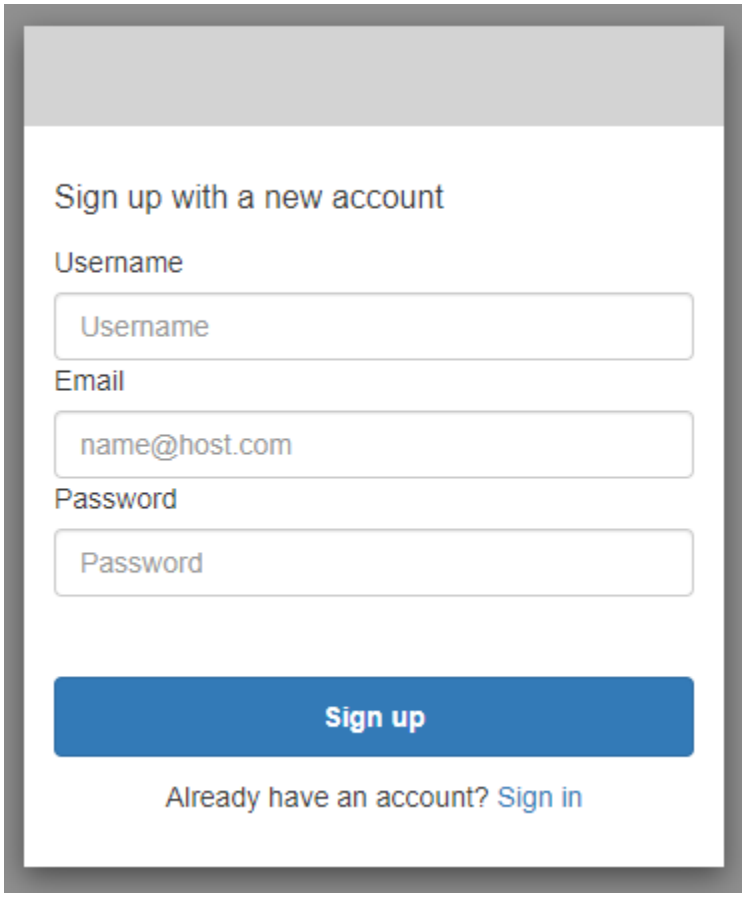
With multiple sign-in providers



With only Amazon Cognito as a sign-in provider

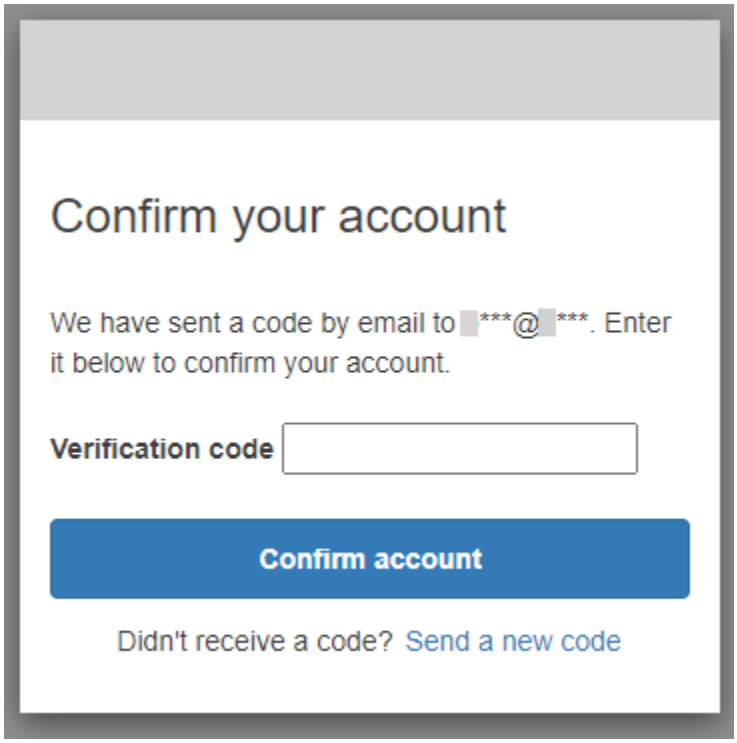


2. Na página Sign up with a new account (Cadastrar-se com uma nova conta), o proprietário da aplicação solicita as informações necessárias para cadastro. Ele pode solicitar um nome de usuário, endereço de e-mail ou número de telefone. Insira as informações necessárias e escolha uma senha.



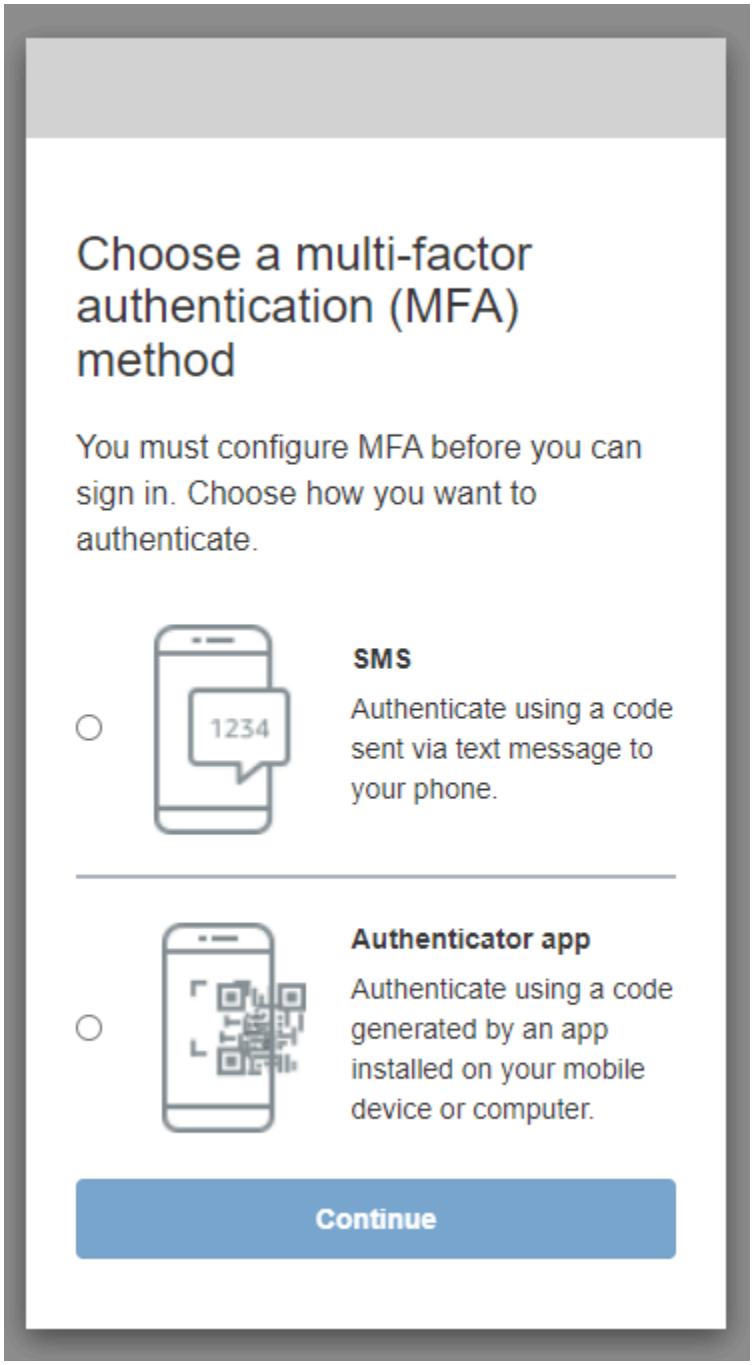
3. Na página Confirm your account (Confirmar sua conta), o proprietário da aplicação pode exigir que confirme a conta para verificar se você pode receber mensagens no endereço de e-mail ou no número de telefone fornecido.

   Você receberá um código no e-mail ou em uma mensagem de texto SMS. Insira o código no formulário para confirmar que você inseriu as informações de contato corretas.



4. O proprietário da aplicação pode exigir que você configure a autenticação multifator (MFA). Você pode receber uma solicitação para selecionar o método de MFA ou a aplicação pode passar para a próxima etapa.

   Na página Choose a multi-factor authentication (MFA) method [Selecionar um método de autenticação multifator (MFA)], escolha um método de MFA. Se você selecionar SMS, receberá senhas de MFA em mensagens de texto SMS. Se selecionar Authenticator app (Aplicação autenticadora), deverá instalar uma aplicação no dispositivo para gerar senhas de MFA com base no tempo. Você deve fazer uma escolha em até três minutos.



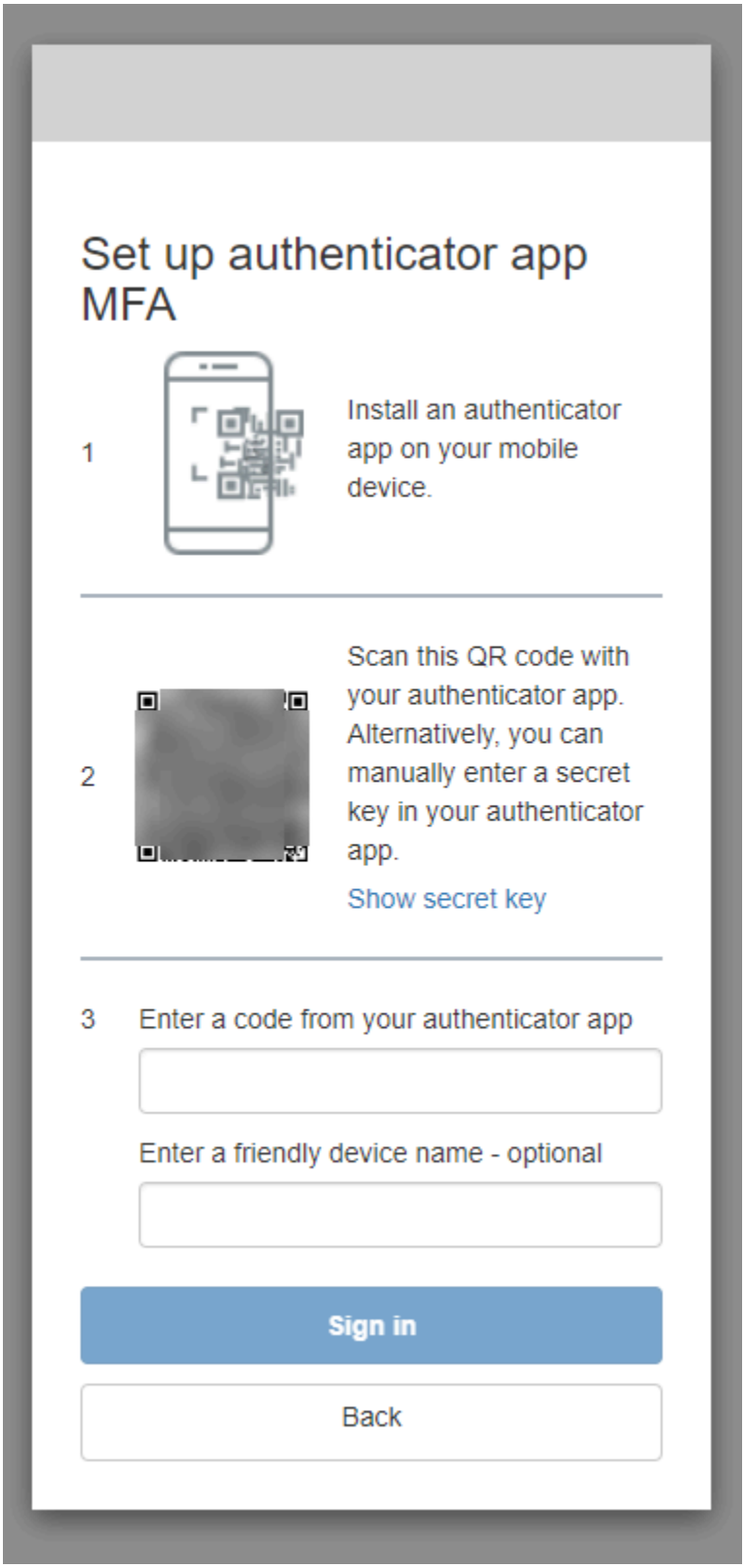
5. O Amazon Cognito solicita um código da aplicação autenticadora ou mensagem de texto SMS. Insira o código que você recebeu em até três minutos.

   Authenticator app

   1. Abra a aplicação autenticadora que você baixou.
   2. Use a câmera para digitalizar o código QR na página. Talvez seja necessário autorizar a aplicação a usar a câmera.

Se você não conseguir digitalizar o código QR, selecione Show secret key (Mostrar chave secreta) para exibir um código que você pode inserir manualmente na aplicação autenticadora.

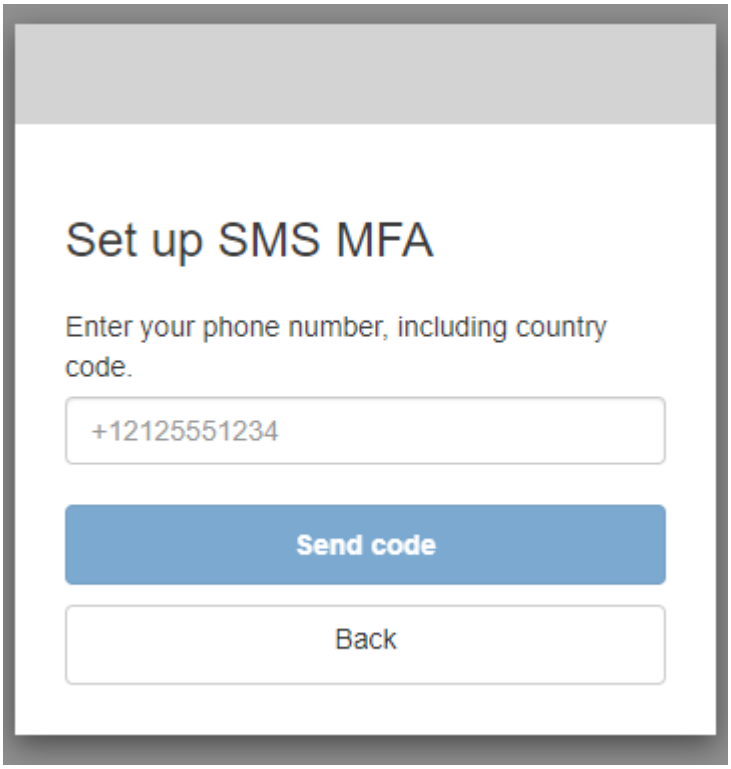
3. A aplicação autenticadora começa a exibir códigos que mudam a cada alguns segundos. Insira um código atual fornecido pela aplicação.
4. (Opcional) Na página Set up authenticator app MFA (Configurar a MFA da aplicação autenticadora), selecione um nome para o dispositivo. Quando você fizer login, o Amazon Cognito solicitará um código do dispositivo com o nome fornecido aqui.


Set up authenticator app MFA
1
Install an authenticator app on your mobile device.
2
Scan this QR code with your authenticator app. Alternatively, you can manually enter a secret key in your authenticator app.
Show secret key
3
Enter a code from your authenticator app
Enter a friendly device name - optional
Sign in
Back

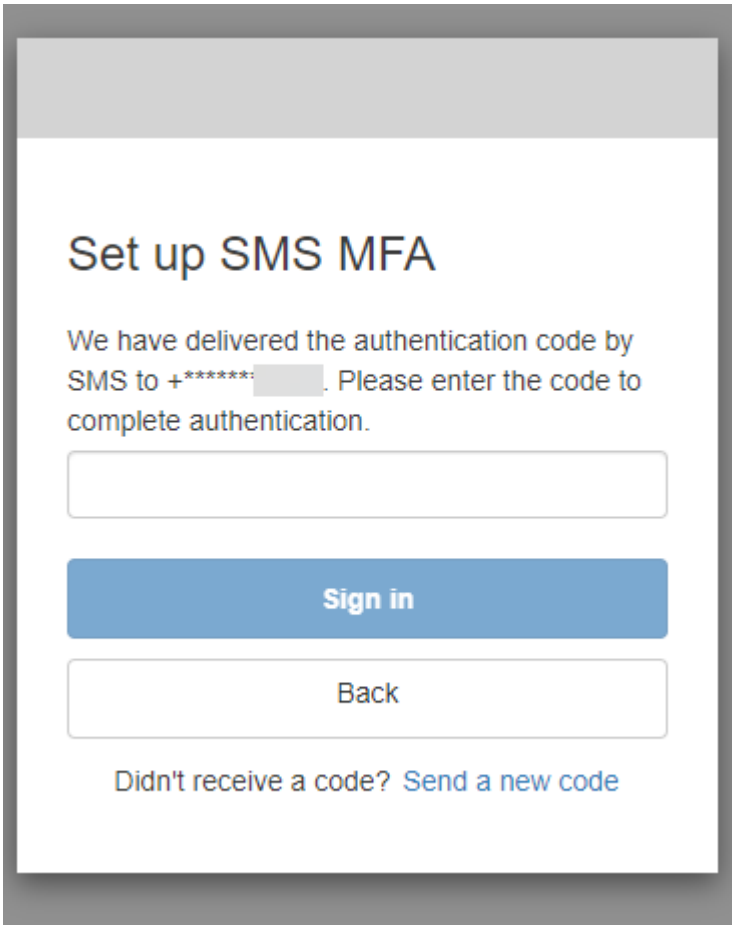
SMS text message

1. Se o proprietário da aplicação ainda não tiver obtido seu número de telefone, o Amazon Cognito o solicitará.

   Na página Set up SMS MFA (Configurar MFA por SMS), insira um número de telefone que inclua o sinal + e um código de país, por exemplo, +12125551234.



2. Você receberá uma mensagem SMS com um código. Na página Set up SMS MFA (Configurar MFA por SMS), insira o código. Se você não recebeu um código e quiser tentar novamente, selecione Send a new code (Enviar um novo código). Selecione Back (Voltar) para inserir um novo número de telefone.



6. Na primeira vez em que você se cadastra e confirma suas informações, o Amazon Cognito concede acesso à sua aplicação após a conclusão desse processo.

Como fazer login com a UI hospedada do Amazon Cognito

Este guia mostra como fazer login em aplicações que usam o Amazon Cognito.

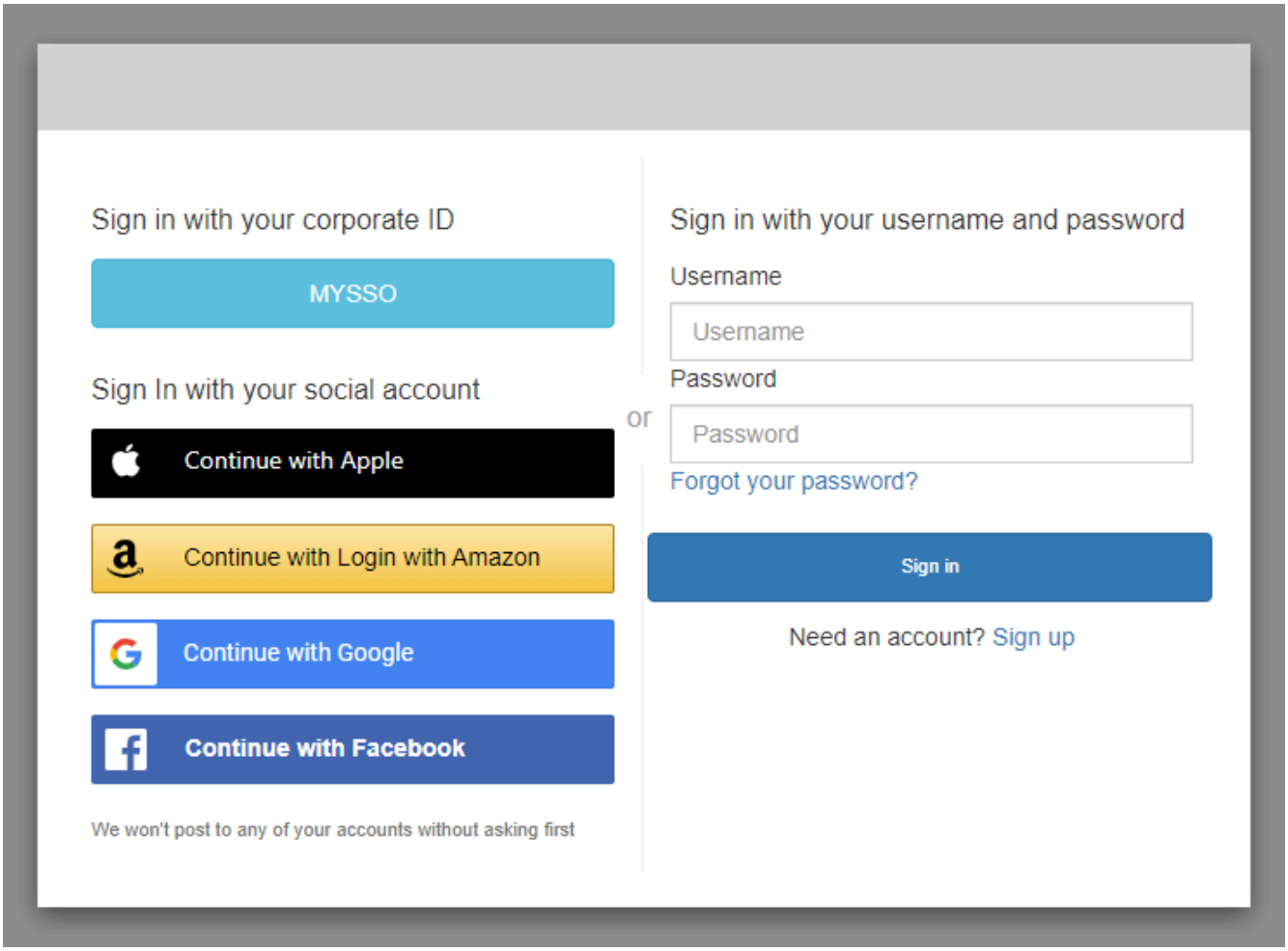
> **Note**
>
> Ao fazer login em uma aplicação que usa a interface do usuário (UI) hospedada do Amazon Cognito, você pode ver uma página que o proprietário da aplicação personalizou, além da configuração básica mostrada neste guia.

1. Dependendo das opções selecionadas pelo proprietário da aplicação, você pode selecionar entre os provedores com os quais fazer login ou ver apenas uma solicitação de nome de usuário e senha. Quando você faz login com um nome de usuário e senha nesta página, o Amazon

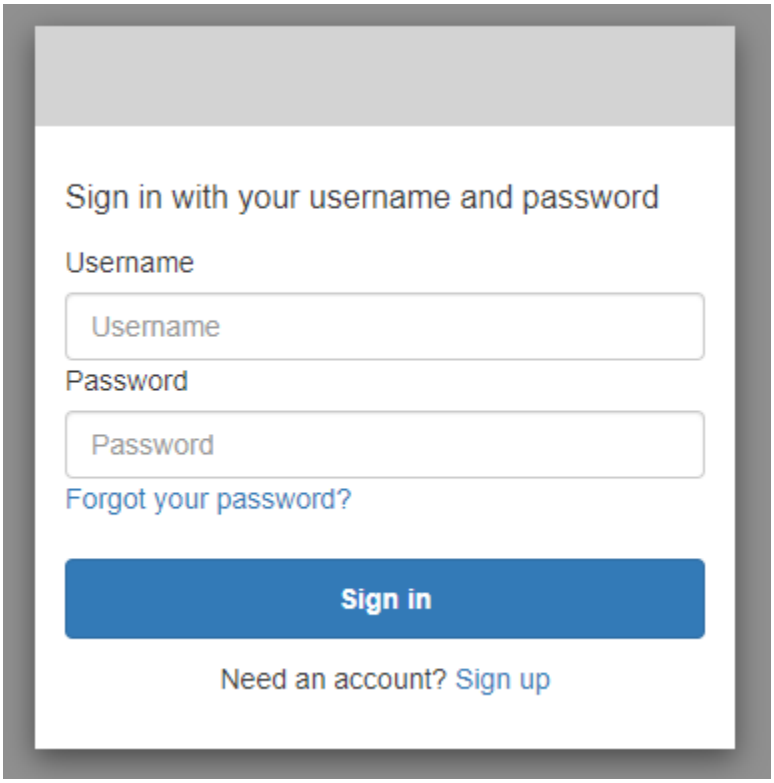
Cognito é seu provedor de login. Do contrário, o provedor de login será representado pelo botão que você escolher.

Você pode selecionar um provedor aqui ou inserir um nome de usuário e senha e ter acesso à aplicação imediatamente. Se o Amazon Cognito for seu provedor de login, o proprietário da aplicação também poderá exigir autenticação multifator.

With multiple sign-in providers

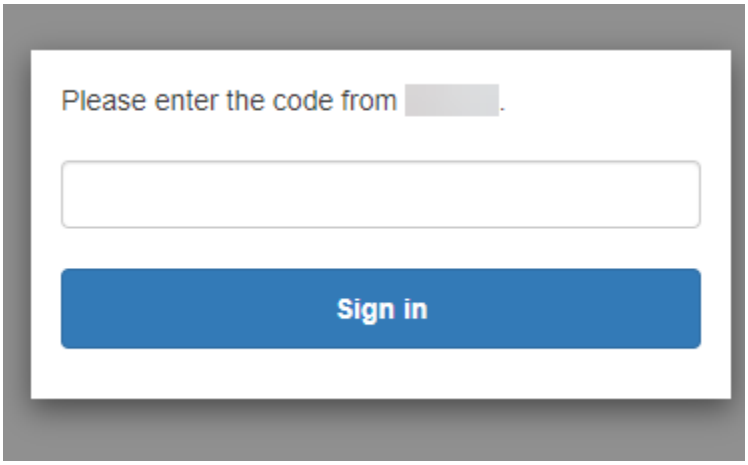


With only Amazon Cognito as a sign-in provider

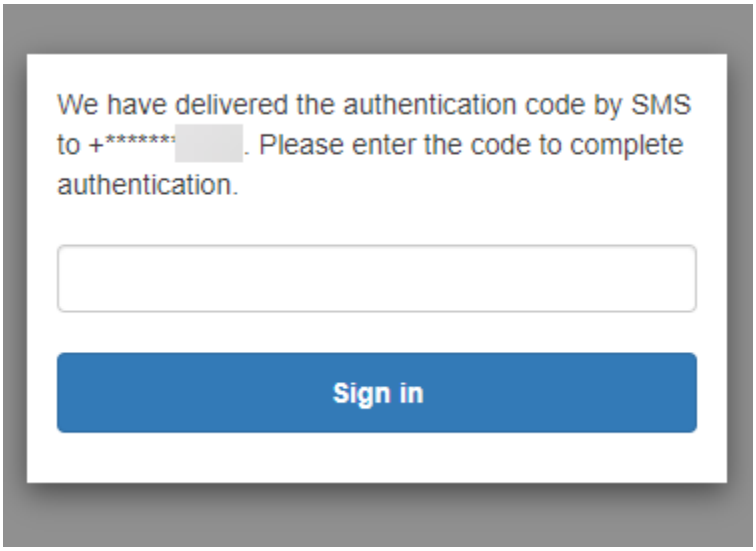


2. Você pode ter configurado a MFA quando se cadastrou na aplicação. Insira o código de MFA recebido em uma mensagem SMS ou exibido em sua aplicação autenticadora. Você deve inserir esse código em até três minutos.

   With an authenticator app



With an SMS code



3. Depois que você faz login e conclui a MFA, o Amazon Cognito concede acesso à aplicação.

Como redefinir uma senha com a UI hospedada no Amazon Cognito

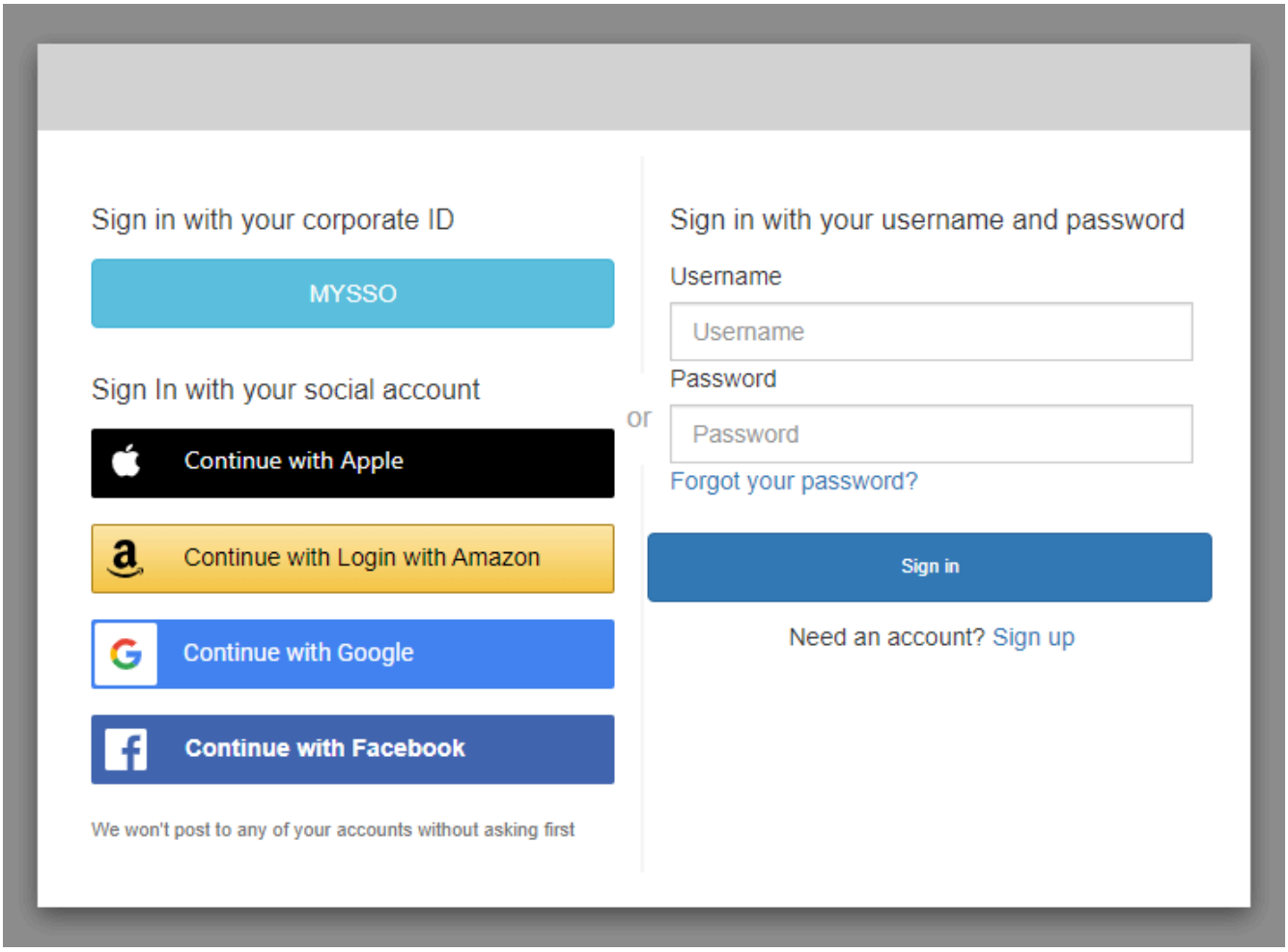
Este guia mostra como redefinir sua senha em aplicações que usam o Amazon Cognito.

> **Note**
>
> Ao fazer login em uma aplicação que usa a interface do usuário (UI) hospedada do Amazon Cognito, você pode ver uma página que o proprietário da aplicação personalizou, além da configuração básica mostrada neste guia.

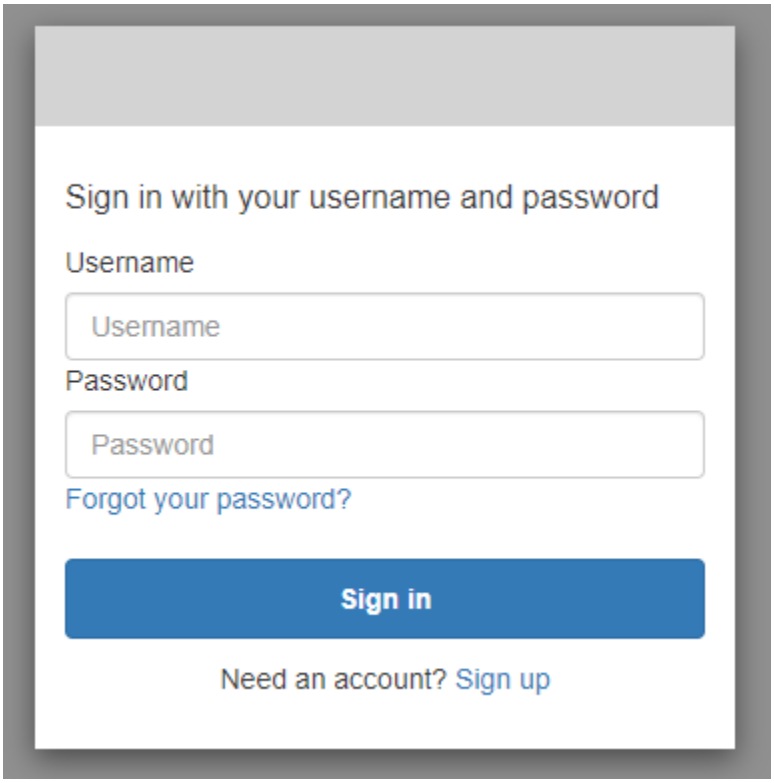
1. Dependendo das opções selecionadas pelo proprietário da aplicação, você pode selecionar entre os provedores com os quais fazer login ou ver apenas uma solicitação de nome de usuário e senha. Quando você faz login com um nome de usuário e senha nesta página, o Amazon Cognito é seu provedor de login. Do contrário, o provedor de login será representado pelo botão que você escolher.

   Se você normalmente seleciona um provedor na página de login e sua senha não está funcionando, siga o procedimento para redefini-la junto ao provedor. Se o Amazon Cognito for seu provedor de login, selecione Forgot your password? (Esqueceu a senha?)

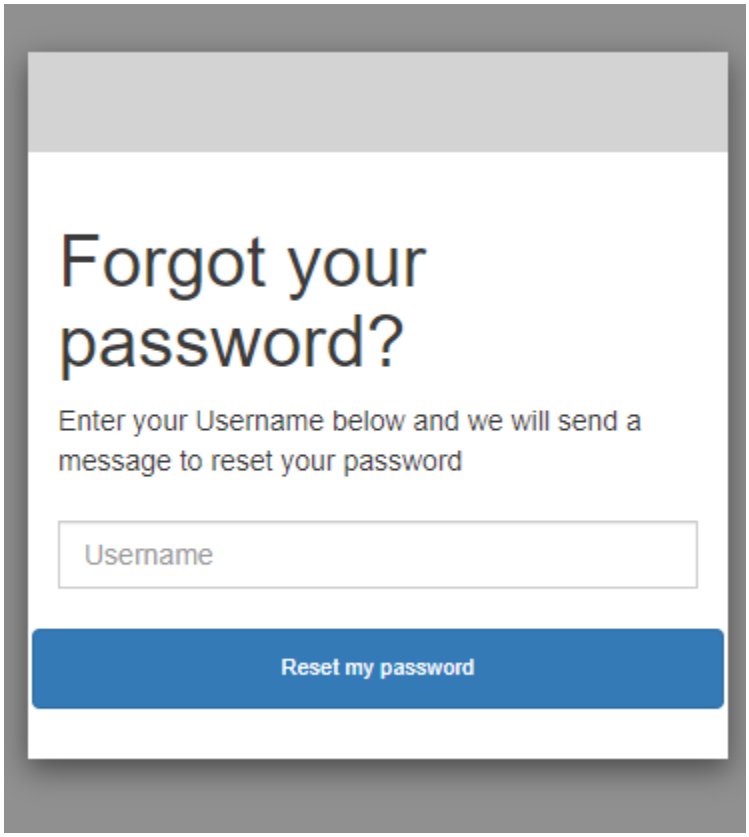
With multiple sign-in providers



With only Amazon Cognito as a sign-in provider

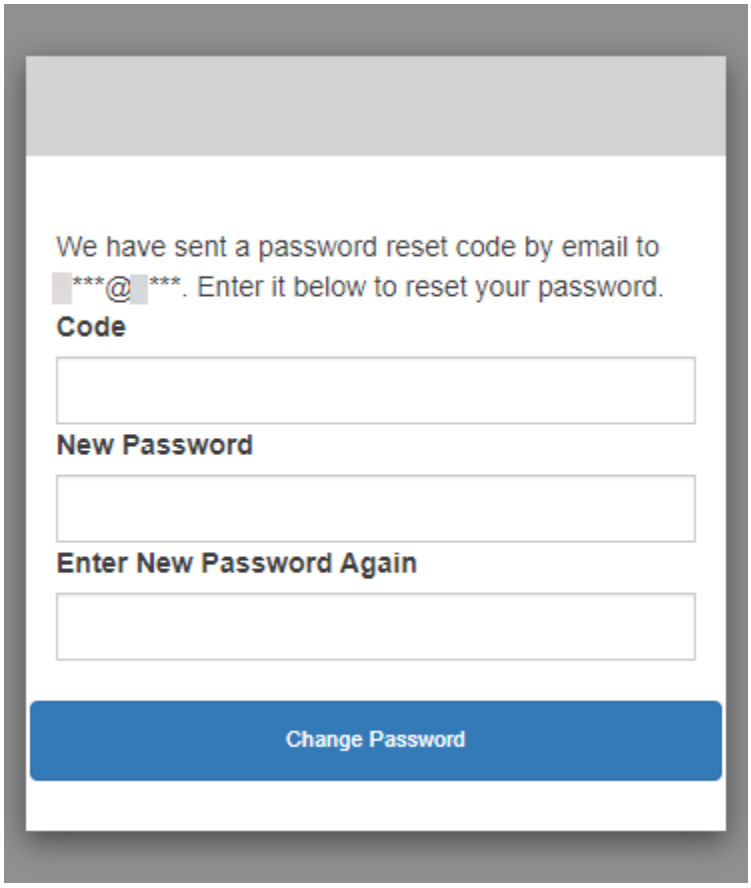


2. Na página Forgot your password (Esqueceu a senha?), o Amazon Cognito solicita as informações que você usa para fazer login. Pode ser seu nome de usuário, endereço de e-mail ou número de telefone.



3. O Amazon Cognito enviará um código para você em uma mensagem de e-mail ou uma mensagem de texto SMS.

   Insira o código que você recebeu e digite a nova senha duas vezes nos campos fornecidos. Você deve inserir o código de redefinição em até oito minutos.



4. Depois de alterar a senha, retorne à página de login e faça login com a nova senha.

Escopos, M2M e autorização de API com servidores de recursos

Depois de configurar um domínio para o grupo de usuários, o Amazon Cognito fornecerá automaticamente um servidor de autorização do OAuth 2.0 e uma interface de usuário da web hospedada com as páginas de cadastro e login que a aplicação pode apresentar aos usuários. Para obter mais informações, consulte Adicionar um cliente de aplicativo com a interface hospedada. 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 do 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 `scope` declaração do token de acesso. Para obter mais informações sobre as reivindicações nos tokens de acesso do Amazon Cognito, consulte Como usar o token de acesso.

Com o Amazon Cognito, os escopos nos tokens de acesso podem autorizar o acesso às APIs externas ou aos atributos do usuário. Você pode emitir tokens de acesso para usuários locais, usuários federados ou identidades de máquinas.

Autorização M achine-to-machine (M2M)

O Amazon Cognito oferece suporte a aplicativos 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 a APIs remotas. 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.

Um aplicativo que acessa uma API com autorização M2M deve ter um ID e um segredo do cliente. Em seu grupo de usuários, você deve criar um cliente de aplicativo que ofereça suporte à concessão de credenciais de clientes. Para oferecer suporte às credenciais do cliente, seu cliente de aplicativo deve ter um segredo de cliente e você deve ter um domínio de grupo de usuários. Nesse fluxo, a identidade da sua máquina solicita um token de acesso diretamente doEndpoint 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 como configurar clientes de aplicativos, consulteClientes de aplicações de grupos de usuários.

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, continue lendo. Para ver um exemplo de aplicativo, consulte Autorização máquina a máquina baseada no Amazon Cognito e no API Gateway usando o AWS CDK.

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, o faturamento M2M reflete as credenciais ativas do cliente, os clientes do aplicativo e o volume total de solicitações de tokens. Para mais informações, consulte Preços 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 seus aplicativos fazem. Veja uma maneira Armazenar tokens em cache de 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 da

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-padrão.

O escopo reservado da API do grupo de usuários

O escopo `aws.cognito.signin.user.admin` autoriza a 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 um usuário do grupo de usuários com, por exemplo, as operações de UpdateUserAttributesAPI GetUsere. 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 do 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 às APIs externas 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-padrão

Ao autenticar usuários com o servidor de autorização OAuth 2.0 do grupo de usuários, inclusive com a interface de usuário hospedada, 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 padrão do OAuth 2.0 autorizam a aplicação a ler as informações de usuário do Endpoint do UserInfo do grupo de usuários. O modelo do OAuth, em que você consulta os atributos do usuário por meio do endpoint `userInfo`, pode otimizar a aplicação para um alto 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 OAuth 2.0.

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 padrão, 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 do UserInfo. Uma solicitação com o parâmetro `scope=openid` exibe todos os atributos legíveis pelo cliente no token de ID e de `userInfo`.

profile

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 compatíveis com o OAuth 2.0. As APIs REST do Amazon API Gateway têm 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.

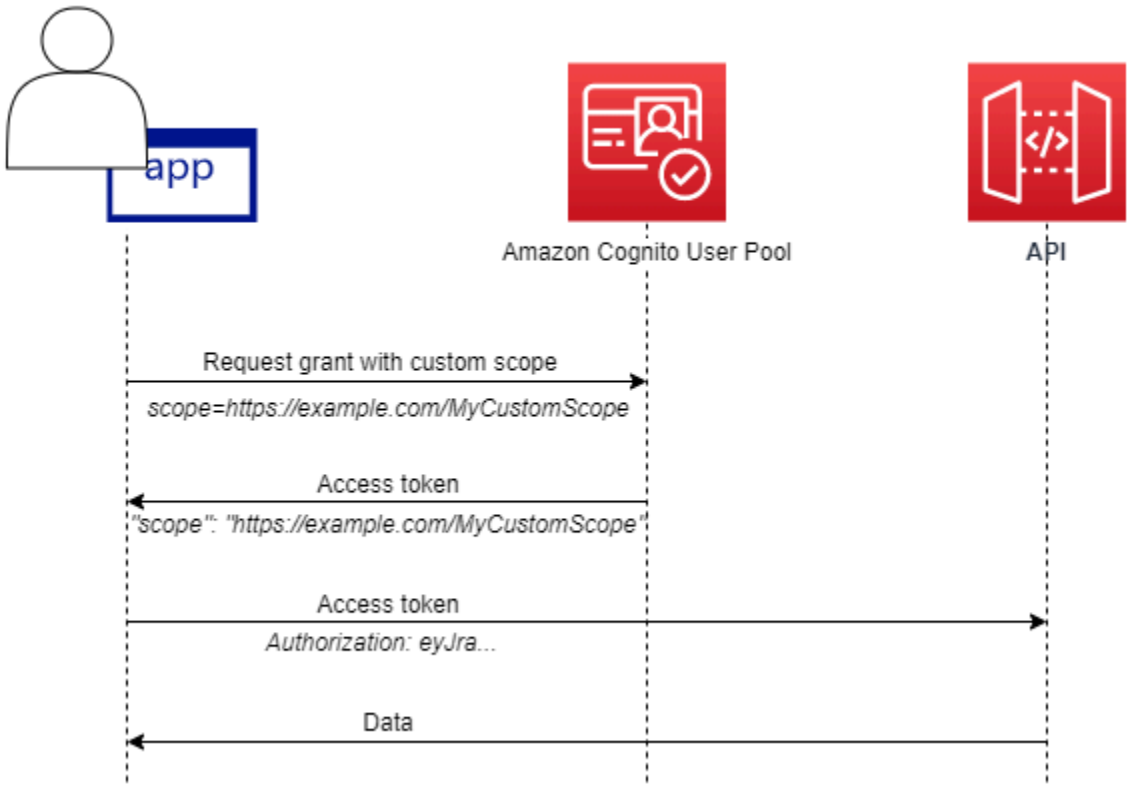
Os escopos personalizados são definidos por você e ampliam os recursos de autorização de um grupo de usuários para incluir propósitos não relacionados à consulta e modificação de usuários e seus atributos. Por exemplo, se você possui um servidor de recursos para fotos, ele pode definir dois escopos: `photos.read` para acesso de leitura das fotos e `photos.write` para acesso de gravação/exclusão. 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 um token 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

Com o Amazon Cognito, é possível criar Servidores de recursos do 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 escopos personalizados a um cliente da aplicação e solicite esses escopos nas concessões de código de autorização do OAuth 2.0, nas concessões implícitas e nas 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. Na UI hospedada, 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 usuários durante os fluxos de autenticação com o servidor de autorização OAuth 2.0 que inclui a interface de usuário hospedada. 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 AWS Management Console 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 (AWS Management Console)

Você pode usar o AWS Management Console 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. Escolha a guia App integration (Integração da aplicação) e localize Resource servers (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. Selecione Create (Criar).

Seus escopos personalizados podem ser revisados na guia App integration (Integração da aplicação) em Resource servers (Servidores de recursos), na coluna Custom scopes (Escopos personalizados). É possível habilitar escopos personalizados para clientes de aplicações na guia App integration (Integração da aplicação) em App clients (Clientes da aplicação). Selecione um cliente da aplicação, localize Hosted UI settings (Configurações de interface do usuário hospedada) e escolha Edit (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

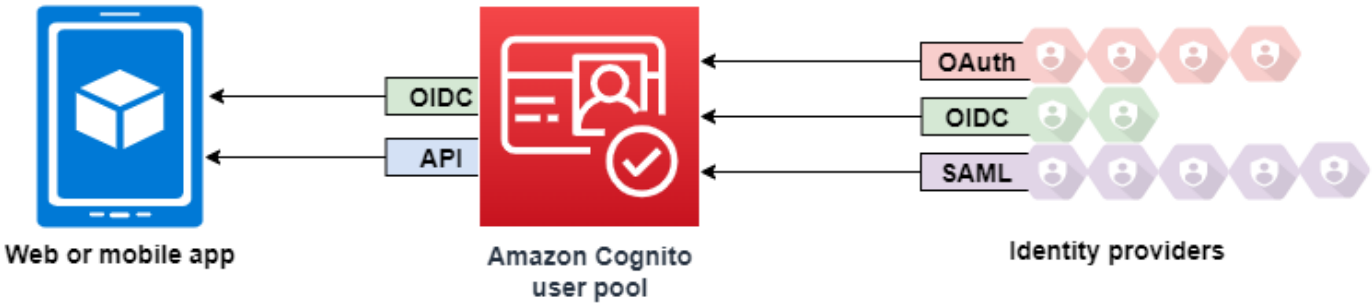
Como adicionar acesso a grupo de usuários por meio de terceiros

Os usuários do seu aplicativo podem fazer login diretamente por meio de um grupo de usuários ou podem se federar por meio de um provedor de identidade (IdP) terceirizado. 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 identidades (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 IdP OAuth 2.0 e OIDC. Ele gera tokens de acesso, tokens de ID e tokens de atualização. Para mais informações sobre tokens, consulte Como usar tokens com grupos de usuários.

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 a interface de usuário hospedada do Amazon Cognito e interface do usuário hospedada e endpoints OIDC. Para ter mais informações, consulte Como usar o próprio domínio para a interface do usuário hospedada.
- 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 a interface de usuário hospedada. Caso contrário, ela será

redirecionada para o Endpoint de login da interface de usuário hospedada. Para ver um exemplo, consulte Exemplo de cenário: marcar aplicativos do Amazon Cognito como favoritos em um painel corporativo.

- Quando a interface do usuário hospedada 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 ter mais informações, consulte Coisas a saber sobre mapeamentos.

  Exemplo: `MyIDP_bob@example.com`
- 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 um minuto para aparecerem na interface de usuário hospedada.
- O Amazon Cognito é compatível com até 20 redirecionamentos HTTP entre ele e o IdP.
- Quando o usuário faz login com a interface do usuário hospedada, 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, a interface do usuário hospedada 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 especificar mapeamentos de atributos do provedor de identidade para seu grupo de usuários
- Vincular usuários federados a um perfil de usuário existente

Como configurar provedores de identidade para seu grupo de usuários

Na guia Experiência de login, em Login do provedor de identidade federado, você pode adicionar provedores de identidade (IdPs) ao seu grupo de usuários. Para ter mais informações, consulte Como adicionar acesso a grupo de usuários por meio 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 credenciais da AWS.
2. No painel de navegação, escolha User Pools (Grupos de usuários) e escolha o grupo de usuários que deseja editar.

3. Escolha a guia Sign-in experience (Experiência de login) e localize Federated sign-in (Acesso federado).
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 Usando 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. Na guia App client integration (Integração de cliente da aplicação), escolha um dos App clients (Clientes da aplicação) na lista e escolha Edit hosted UI settings (Editar configurações da interface do usuário hospedada). 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, consulteUsando 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 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. Escolha a guia Sign-in experience (Experiência de acesso). Localize Federated sign-in (Acesso federado) e selecione Add an identity provider (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 endpoint Auto fill through issuer URL (Preenchimento automático por meio do URL do emissor) quando o Amazon Cognito puder recuperar os URLs dos endpoints `authorization`, `token`, `userInfo` e `jwks_uri`.

12. Insira o URL do emissor ou os URLs dos endpoints `authorization`, `token`, `userInfo` e `jwks_uri` de seu IdP.

> **Note**
>
> Você pode usar somente os números de porta 443 e 80 com descoberta, preenchimento automático e URL inseridos manualmente. 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 documento `openid-configuration` associado ao URL do emissor deve fornecer URLs HTTPS para os seguintes valores: `authorization_endpoint`, `token_endpoint`, `userinfo_endpoint` e `jwks_uri`. Da mesma forma, quando você escolhe Manual input (Entrada manual), você só pode inserir URLs HTTPS.

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. Selecione Create.
16. Na guia App client integration (Integração de cliente da aplicação), selecione um dos App clients (Clientes da aplicação) na lista e Edit hosted UI settings (Editar configurações da interface do usuário hospedada). Adicione o novo provedor de identidade OIDC ao cliente da aplicação em Identity providers (Provedores de identidade).
17. Escolha Salvar alterações.

Para obter mais informações sobre o OIDC IdPs, consulte. Usando 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. Configurando seu provedor de identidade SAML terceirizado

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 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. Escolha a guia Sign-in experience (Experiência de acesso). Localize Federated sign-in (Acesso federado) e selecione Add an identity provider (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 identidade SAML 2.0 para enviar respostas de desconexão para o endpoint `https://mydomain.us-east-1.amazoncognito.com/saml2/logout` que o Amazon Cognito cria quando você configura a interface do usuário hospedada. 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. Selecione Create.

> **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

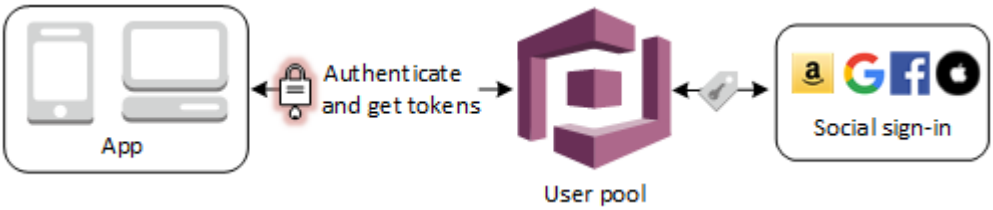
- Para obter o certificado que contém a chave pública que será usada pelo IdP para verificar a solicitação de logout assinada, escolha Mostrar certificado de assinatura em Provedores SAML ativos na caixa de diálogo SAML em Provedores de identidade da página Federação do console.

Para obter mais informações sobre SAML, IdPs consulte Usando provedores de identidade SAML com um grupo de usuários.

Usando 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 a interface do usuário hospedada para se integrar com provedores de identidade social compatíveis. Quando o Amazon Cognito cria sua interface de usuário hospedada, 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 AWS Management Console, 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

- Pré-requisitos

- Etapa 1: inscrever-se com um IdP social
- Etapa 2: adicionar um IdP social ao seu grupo de usuários
- Etapa 3: testar a configuração do IdP social

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 social.

Etapa 1: 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).
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.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.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 endpoint `/oauth2/idpresponse` para o domínio de seu grupo de usuários em Valid OAuth Redirect URIs (URIs de redirecionamento do OAuth válidos).

    ```
    https://mydomain.us-east-1.amazoncognito.com/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 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.us-east-1.amazoncognito.com
   ```

10. Insira o domínio do grupo de usuários com o endpoint `/oauth2/idpresponse` em Allowed Return URLs (URLs permitidos de retorno).

    ```
    https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
    ```

11. Selecione Save (Salvar).

Para registrar um aplicativo com o Google

Para obter mais informações sobre o OAuth 2.0 na plataforma Google Cloud, consulte Learn about authentication & authorization (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 à esquerda, escolha APIs and Services (APIs e serviços) e depois Oauth consent screen (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 Scopes (Escopos), escolha Add or remove scopes (Adicionar ou remover escopos) e selecione, no mínimo, os escopos do OAuth a seguir.

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 novamente a barra de navegação à esquerda e escolha APIs and Services (APIs e serviços) e, depois, Credentials (Credenciais).
11. Escolha CREATE CREDENTIALS (Criar credenciais) e, depois, OAuth client ID (ID do cliente do OAuth).
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.us-east-1.amazoncognito.com
    ```

14. Em Authorized redirect URIs (URIs de redirecionamento autorizadas), escolha ADD URI (Adicionar URI). Insira o caminho para o endpoint `/oauth2/idpresponse` do domínio de seu grupo de usuários.

    ```
    https://mydomain.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.

Para registrar uma aplicação na 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.

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 Register a New Identifier (Registrar um novo identificador), escolha App IDs (IDs de aplicação) e selecione Continue (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 Etapa 2: adicionar um IdP social ao seu grupo de usuários.
    3. Em Capabilities (Recursos), escolha Sign In with Apple (Fazer login com a Apple) e, depois, selecione Edit (Editar).
    4. Na página Sign in with Apple: App ID Configuration (Fazer login com a Apple: configuração do ID da aplicação), escolha configurar a aplicação como principal ou agrupada com outros IDs de aplicação e, depois, escolha Save (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 Register a New Identifier (Registrar um novo identificador), escolha Services IDs (IDs de serviços) e selecione Continue (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 Etapa 2: adicionar um IdP social ao seu grupo de usuários.
    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 de Website URLs (URLs de site).
4. Em Domains and subdomains (Domínios e subdomínios), insira o domínio do grupo de usuários sem um prefixo `https://`.

   ```
   mydomain.us-east-1.amazoncognito.com
   ```

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

   ```
   https://mydomain.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). Selecione Save (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 Etapa 2: adicionar um IdP social ao seu grupo de usuários.

Etapa 2: adicionar um IdP social ao seu grupo de usuários

Para configurar um IdP social do grupo de usuários com o AWS Management 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.
4. Escolha a guia Sign-in experience (Experiência de login). Localize Federated sign-in (Acesso federado) e selecione Add an identity provider (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 with 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 após sair da página de acesso com a Apple (podem ter origem em falhas internas dentro do Amazon Cognito ou de qualquer elemento escrito pelo desenvolvedor)
- O identificador do ID do serviço é usado nos grupos de usuários e/ou em 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. Selecione Create (Criar).
10. Na guia App client integration (Integração de cliente da aplicação), escolha um dos App clients (Clientes da aplicação) na lista e escolha Edit hosted UI settings (Editar configurações da interface do usuário hospedada). Adicione o novo IdP social ao cliente de aplicação em Identity providers (Provedores de identidade).
11. Escolha Salvar alterações.

Etapa 3: 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.us-east-1.amazoncognito.com/login?
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

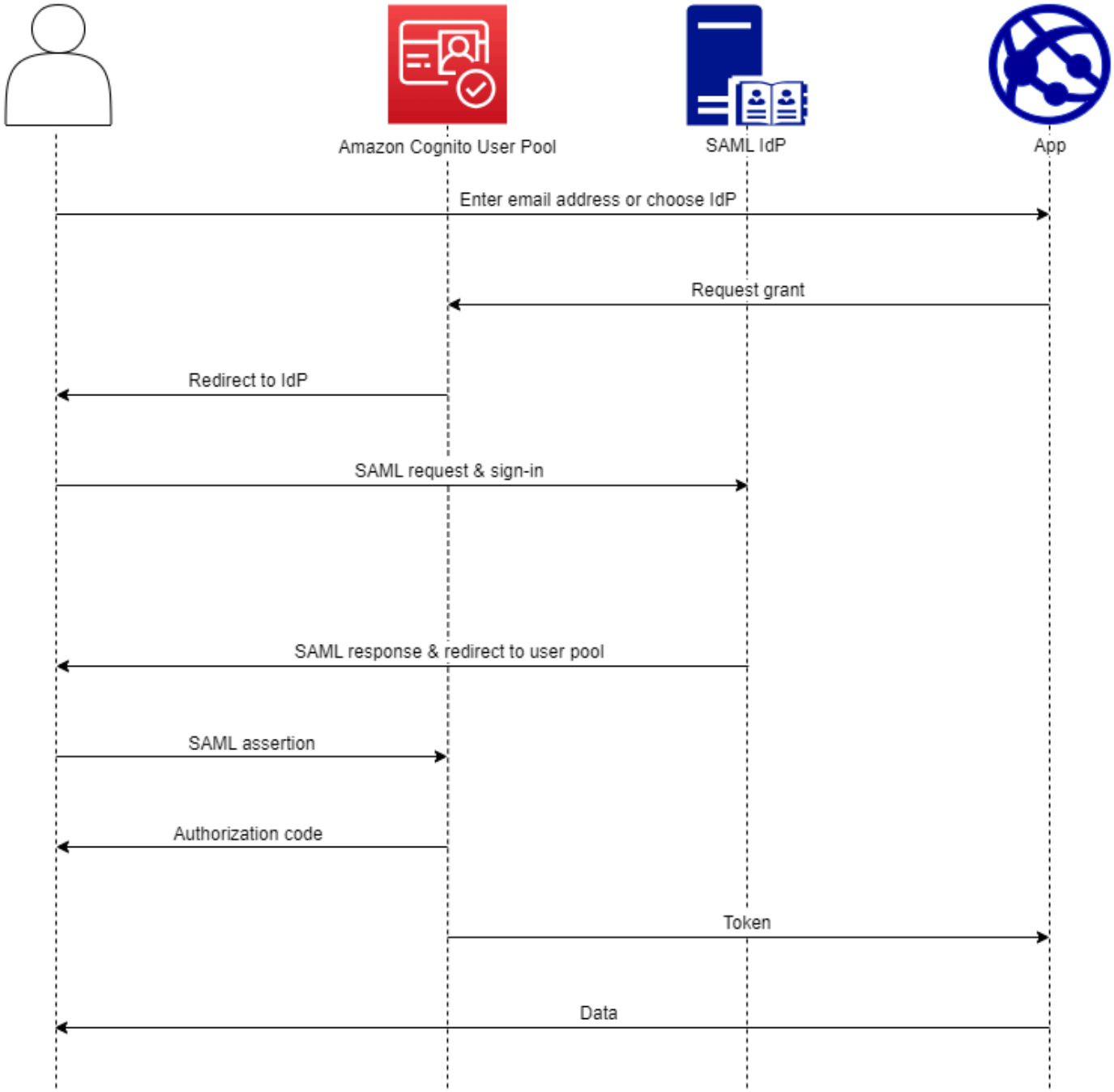
Você pode encontrar o domínio na página do console Domain name (Nome do domínio) do grupo de usuários. O client_id está na página App client settings (Configurações de cliente de aplicação). Use o URL de retorno de chamada para o parâmetro redirect_uri. Esse é o URL da página para a qual o usuário será 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 a interface do usuário hospedada. A página exibe a mensagem de erro `Something went wrong` (Ocorreu algum problema).

Usando provedores de identidade SAML com um grupo de usuários

Você pode optar por fazer com que seus usuários de aplicativos móveis e da web entrem por meio de um provedor de identidade SAML (IdP), como o Microsoft Active Directory Federation Services (ADFS) ou Shibboleth. Você deve escolher um IdP SAML compatível com o padrão SAML 2.0.

Com a interface hospedada e os endpoints de federação, 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 todos os seus aplicativos. 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 AWS Management Console a API de AWS CLI grupos de usuários do Amazon Cognito. Para criar seu primeiro IdP SAML no AWS Management Console, consulte. 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 grupos 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 grupo de identidades. Um grupo de usuários pode

ser um IdP de terceiros para um grupo de identidades. Para ter mais informações, consulte Banco de identidades do Amazon Cognito.

Referência rápida para configuração do IdP

Você deve configurar seu SAML IdP para aceitar solicitações e enviar respostas ao seu grupo de usuários. A documentação do seu IdP SAML conterá informações sobre como adicionar seu grupo de usuários como uma parte confiável ou aplicativo para seu IdP SAML 2.0. A documentação a seguir fornece os valores que você deve fornecer para o ID da entidade SP e o URL do serviço ao consumidor de asserção (ACS).

Referência rápida de valores SAML do grupo de usuários

ID da entidade SP

```
urn:amazon:cognito:sp:us-east-1_EXAMPLE
```

URL DO ANÚNCIO

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

Você deve configurar seu grupo de usuários para oferecer suporte ao seu provedor de identidade. As etapas de alto nível 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 Configurando seu provedor de identidade SAML terceirizado.
2. Adicione um novo IdP ao seu grupo de usuários. Faça upload dos metadados do SAML ou forneça o URL dos metadados. Consulte Adicionar e gerenciar provedores de identidade SAML em um grupo de usuários.
3. Atribua o IdP aos seus clientes de aplicativos. Consulte Clientes de aplicações de grupos de usuários

Tópicos

- Coisas que você deve saber sobre o SAML IdPs nos grupos de usuários do Amazon Cognito
- Diferenciação entre maiúsculas e minúsculas dos nomes de usuário SAML

- Adicionar e gerenciar provedores de identidade SAML em um grupo de usuários
- Iniciação de sessão SAML em grupos de usuários do Amazon Cognito
- Usando o login SAML iniciado pelo SP
- Usando o login SAML iniciado pelo IdP
- Fluxo de saída do SAML
- Assinatura e criptografia SAML
- Nomes e identificadores do provedor de identidade SAML
- Configurando seu provedor de identidade SAML terceirizado

Coisas que você deve saber sobre o SAML IdPs nos grupos de usuários do Amazon Cognito

O Amazon Cognito processa declarações de 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 oferece suporte ao single sign-on (SSO) iniciado pelo SP e pelo IdP, conforme descrito nas seções 5.1.2 e 5.1.4 da Visão geral técnica do SAML V2.0.

Forneça um certificado de assinatura de IdP válido

O certificado de assinatura nos metadados do seu provedor de SAML não deve expirar quando você configura o IdP do SAML em seu grupo de usuários.

Grupos de usuários oferecem suporte a 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 não deve ter mais de 4.096 caracteres.

Mantenha o parâmetro de estado do relé

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 `relayState` token é uma referência opaca 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).

Identifique 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.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.

Nenhuma afirmação repetida

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

Você deve fornecer ao IdP o ID do grupo de usuários no provedor de serviços (SP)`urn`, também chamado de URI do público ou ID da entidade do 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 seu ID do grupo de usuários em Visão geral do grupo de usuários no console do Amazon Cognito.

Mapeie todos os atributos necessários

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 Como especificar mapeamentos de atributos do provedor de identidade para seu grupo de usuários.

O formato de afirmação tem requisitos específicos

Seu IdP do SAML deve incluir as seguintes declarações na declaração do SAML.

1. Uma `NameID` reclamação. O Amazon Cognito associa uma declaração de SAML ao usuário de destino por. `NameID` Se houver `NameID` mudanças, o Amazon Cognito considera que a afirmação é para um novo usuário. O atributo definido `NameID` na configuração do IdP deve ter um valor persistente.

```
<saml2:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:persistent">
  carlos
</saml2:NameID>
```

2. 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>
```

3. Para login único iniciado pelo SP, um `Response` elemento com um `InResponseTo` valor do ID de solicitação SAML original.

```
<saml2p:Response Destination="https://mydomain.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 afirmações de SAML iniciadas pelo IdP não devem conter um valor. `InResponseTo`

4. Um `SubjectConfirmationData` elemento com um `Recipient` valor do `saml2/idpresponse` endpoint do grupo de usuários e, para o SAML iniciado pelo SP, um `InResponseTo` valor que corresponde ao ID da solicitação SAML original.

```
<saml2:SubjectConfirmationData InResponseTo="_dd0a3436-bc64-4679-a0c2-
cb4454f04184" NotOnOrAfter="Date-time stamp" Recipient="https://mydomain.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`. Uma solicitação SAML contém informações sobre seu grupo de usuários, incluindo seu endpoint ACS. Opcionalmente, você pode aplicar uma assinatura criptográfica a essas solicitações.

Assine solicitações e criptografe 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 suportar uma resposta de SAML criptografada faz com que o Amazon Cognito gere um novo par de chaves e um novo certificado de criptografia para esse provedor. Para visualizar e baixar os certificados com a chave pública, escolha seu IdP na guia Experiência de login do console do Amazon Cognito.

Para estabelecer confiança com as solicitações SAML do grupo de usuários, forneça ao IdP uma cópia do certificado de assinatura SAML 2.0 do grupo de usuários. Seu IdP pode ignorar as solicitações 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 passa para o seu 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 login único (SSO) para qualquer IdP externo do SAML. Quando você fornece uma cópia do certificado, seu IdP pode verificar a integridade das solicitações SAML de seus usuários.

2. Seu IdP SAML pode criptografar respostas SAML com o certificado de criptografia. Quando você configura um IdP com criptografia SAML, seu IdP só deve enviar respostas criptografadas.

Codifique caracteres não alfanuméricos

O Amazon Cognito não aceita caracteres UTF-8 de 4 bytes, como # ou, que seu 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 `InvalidParameterException` for exibido durante a criação de um IdP SAML com um URL do endpoint de metadados HTTPS, por exemplo, “Error retrieving metadata from *<metadata endpoint>*”, verifique se o endpoint de metadados está com o SSL configurado corretamente e se há um certificado SSL válido associado a ele. Para obter mais informações sobre a validação de certificados, consulte O que é um certificado SSL/TLS? .

Clientes de aplicativos 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 está impedido de adicionar o diretório de usuários no grupo de usuários e todos os provedores de identidade externos que não sejam SAML a um cliente de aplicativo configurado dessa forma.

As respostas de logout devem usar a vinculação POST

O `/saml2/logout` endpoint aceita `LogoutResponse` como `HTTP POST` solicitações. Os grupos de usuários não aceitam respostas de logout com `HTTP GET` vinculação.

Diferenciação entre maiúsculas e minúsculas dos nomes de usuário SAML

Quando um usuário federado tenta fazer login, o provedor de identidade SAML (IdP) passa uma mensagem exclusiva para o Amazon `NameId` Cognito na declaração de SAML do usuário. O Amazon Cognito identifica um usuário federado SAML por meio da respectiva declaração `NameId`. Independentemente das configurações de distinção entre maiúsculas e minúsculas do seu grupo de usuários, o Amazon Cognito reconhece um usuário federado que retorna de um IdP SAML quando ele passa sua declaração exclusiva e com distinção entre maiúsculas e minúsculas. `NameId` 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

- Adicionar e gerenciar provedores de identidade SAML em um grupo de usuários
- Iniciação de sessão SAML em grupos de usuários do Amazon Cognito
- Usando o login SAML iniciado pelo SP
- Usando o login SAML iniciado pelo IdP
- Fluxo de saída do SAML
- Assinatura e criptografia SAML

- Nomes e identificadores do provedor de identidade SAML
- Configurando seu provedor de identidade SAML terceirizado

Adicionar e gerenciar provedores de identidade SAML em um grupo de usuários

Os procedimentos a seguir demonstram como criar, modificar e excluir provedores de SAML em um grupo de usuários do Amazon Cognito.

AWS Management Console

Você pode usar o AWS Management Console para criar e excluir provedores de identidade SAML (IdPs).

Antes de criar um IdP SAML, você deve ter o documento de metadados SAML obtido do IdP de terceiros. Para obter instruções sobre como obter ou gerar o documento de metadados do SAML necessário, consulte Configurando seu provedor de identidade SAML terceirizado.

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. Escolha a guia Sign-in experience (Experiência de acesso). Localize Federated sign-in (Acesso federado) e escolha Add an identity provider (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 `identity_provider` solicitação para Autorizar endpoint o.
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 seu IdP SAML 2.0 para enviar respostas de desconexão ao endpoint `https://`*`mydomain.us-east-1.amazoncognito.com`*`/saml2/logout` que é criado quando você configura a interface do usuário hospedada. O endpoint `saml2/logout` usa uma associação POST.

> **Note**
>
> Se essa opção for selecionada e seu IdP SAML esperar uma solicitação de logout assinada, você também deverá fornecer ao seu IdP SAML o 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 ter 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**
>
> Recomendamos que você insira uma URL de documento de metadados se seu provedor tiver um endpoint público em vez de fazer o upload de um arquivo. O Amazon Cognito atualiza automaticamente os metadados da URL de metadados. Normalmente, a atualização de metadados ocorre a cada seis horas ou antes de os metadados expirarem, o que ocorrer primeiro.

11. Mapeie atributos entre seu provedor de SAML e seu grupo de usuários para mapear os atributos do provedor de 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. Selecione Create (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 mensagem de resposta do IdP para SP.

  ```
  https://mydomain.us-east-1.amazoncognito.com/saml2/idpresponse
  ```

- Seu IdP SAML deve `NameID` preencher todos os atributos necessários para seu grupo de usuários na declaração SAML. `NameID`é usado para identificar exclusivamente seu usuário federado SAML no grupo de usuários. Seu IdP deve transmitir o ID do nome 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 seu IdP pode usar para validar solicitações de logout do SAML, escolha a guia Experiência de login do seu grupo de usuários, selecione seu IdP e, em Exibir 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. Escolha a guia Experiência de login e localize o login do provedor de identidade federado.
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 oferece suporte ao login único (SSO) iniciado pelo provedor de serviços (iniciado pelo SP) e ao SSO iniciado pelo IdP. Como melhor prática 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 seus usuários do SAML 2.0 se autenticam com um fluxo iniciado pelo SP, eles devem sempre primeiro 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. Quando você oferece suporte à autenticação iniciada pelo IdP, o Amazon Cognito

não pode verificar se solicitou a resposta SAML recebida porque o Amazon Cognito não inicia a autenticação com uma solicitação SAML. No 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 solicitações entre sites (CSRF).

Para obter um exemplo de como criar SAML iniciado pelo SP em um ambiente em que você não deseja que seus usuários interajam com a interface de usuário hospedada do grupo de usuários, consulte. Exemplo de cenário: marcar aplicativos do Amazon Cognito como favoritos em um painel corporativo

Tópicos

Exemplo de cenário: marcar aplicativos do Amazon Cognito como favoritos em um painel corporativo

Você pode criar marcadores em seus painéis SAML ou OIDC IdP que fornecem aos grupos de usuários do Amazon Cognito acesso SSO a aplicativos web. Você pode criar um link com o Amazon Cognito de uma forma que não exija que os usuários façam login com a interface do usuário hospedada. Para fazer isso, adicione um marcador de login ao seu portal que redireciona para o grupo de usuários do Autorizar endpoint Amazon Cognito no seguinte formato.

```
https://mydomain.us-east-1.amazoncognito.com/authorize?
response_type=code&identity_provider=MySAMLIdP&client_id=1example23456789&redire
www.example.com
```

> **Note**
>
> Você também pode usar um parâmetro `idp_identifier`, em vez de um parâmetro `identity_provider`, em sua solicitação ao endpoint de autorização. Um identificador de IdP é um nome ou domínio de e-mail alternativo que você pode configurar ao criar um provedor de identidade em seu grupo de usuários. Consulte Nomes e identificadores do provedor de identidade SAML.

Quando você usa os parâmetros apropriados em sua solicitação ao `/authorize`, o Amazon Cognito inicia silenciosamente o fluxo de login iniciado pelo SP e redireciona o usuário para fazer login em seu IdP.

Para começar, adicione um SAML IdP ao seu grupo de usuários. Crie um cliente de aplicação que use seu IdP SAML para fazer login e tenha o URL para sua aplicação como um URL de retorno de chamada autorizado. Para obter mais informações sobre clientes de aplicação, consulte Clientes de aplicações de grupos de usuários.

Antes de implantar esse acesso autenticado no seu portal, teste a entrada iniciada pelo SP no seu aplicativo a partir da sua interface de usuário hospedada. Para obter mais informações sobre como configurar um IdP SAML no Amazon Cognito, consulte Configurando seu provedor de identidade SAML terceirizado.

O diagrama a seguir mostra um fluxo de autenticação que emula a SSO iniciada pelo IdP. Seus usuários podem se autenticar com o Amazon Cognito por meio de um link no portal de sua empresa.

Depois de atender aos requisitos, crie um marcador para o seu Autorizar endpoint que inclua um `identity_provider` ou um `idp_identifier` parâmetro. A autenticação do usuário prossegue da seguinte forma.

1. Seu usuário faz login no painel de IdP de SSO. As aplicações empresarias que o usuário está autorizado a acessar preenchem esse painel.
2. O usuário escolhe o link para a aplicação que se autentica no Amazon Cognito. Em muitos portais de SSO, você pode adicionar um link de aplicação personalizado. Qualquer recurso que você possa usar para criar um link para um URL público em seu portal de SSO funcionará.
3. Seu link de aplicação personalizado no portal de SSO direciona o usuário para o grupo de usuários Autorizar endpoint. O link inclui parâmetros para `response_type`, `client_id`, `redirect_uri` e `identity_provider`. O parâmetro `identity_provider` é o nome que você atribuiu ao IdP em seu grupo de usuários. Você também pode usar um parâmetro `idp_identifier`, em vez do parâmetro `identity_provider`. Um usuário acessa seu endpoint de federação a partir de um link que contém um parâmetro `idp_identifier` ou`identity_provider`. Esse usuário ignora a página de login e navega diretamente para se autenticar em seu IdP. Para obter mais informações sobre como nomear SAML IdPs, consulte. Nomes e identificadores do provedor de identidade SAML

   URL de exemplo

   ```
   https://mydomain.us-east-1.amazoncognito.com/authorize?
   ```

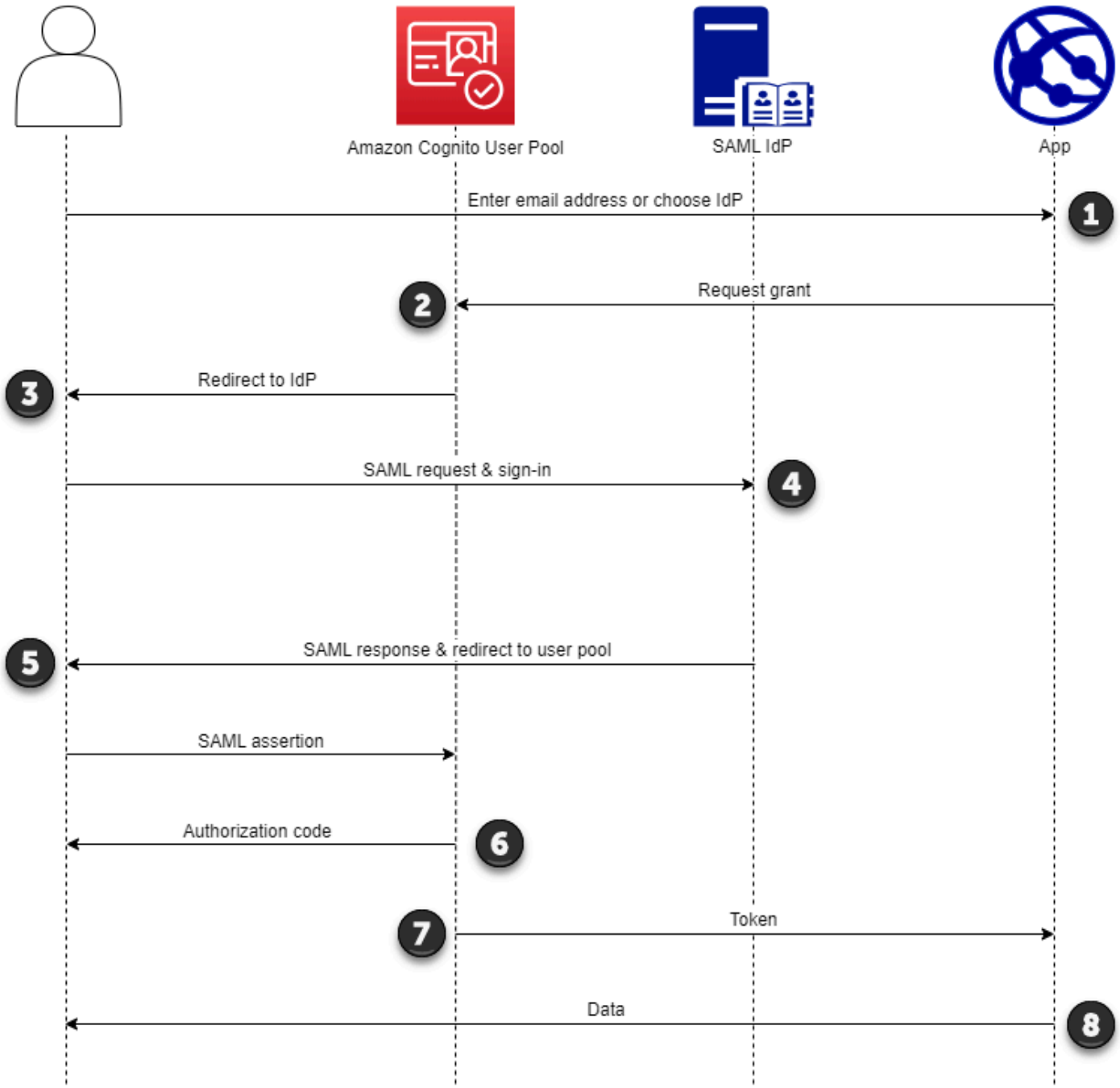
```
response_type=code&
identity_provider=MySAMLIdP&
client_id=1example23456789&
redirect_uri=https://www.example.com
```

4. O Amazon Cognito redireciona a sessão do usuário para o IdP com uma solicitação SAML.
5. O usuário pode ter recebido um cookie de sessão do IdP quando fez login no painel. Seu IdP usa esse cookie para validar o usuário silenciosamente e redirecioná-lo para o endpoint `idpresponse` do Amazon Cognito com uma resposta SAML. Se não houver nenhuma sessão ativa, o IdP autenticará novamente o usuário antes que ele publique a resposta SAML.
6. O Amazon Cognito valida a resposta SAML e cria ou atualiza o perfil do usuário com base na declaração SAML.
7. O Amazon Cognito redireciona o usuário para sua aplicação interna com um código de autorização. Você configurou o URL da aplicação interna como um URL de redirecionamento autorizado para o cliente de aplicação.
8. Sua aplicação troca o código de autorização por tokens do Amazon Cognito. Para ter mais informações, consulte Endpoint de token.

Usando 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 o maior controle sobre quem apresenta as solicitações de login. Você também pode permitir o login iniciado pelo IDP sob certas condições. Para ter mais informações, consulte Iniciação de sessão SAML em grupos de usuários do Amazon Cognito.

O processo a seguir mostra como os usuários fazem login no 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 um aplicativo personalizado ou invocar a interface de usuário hospedada na visualização da web. Você pode configurar sua interface de usuário hospedada para exibir uma lista IdPs ou solicitar apenas um endereço de e-mail.

2. Seu aplicativo invoca o endpoint de redirecionamento do grupo de usuários e solicita uma sessão com o ID do cliente que corresponde ao aplicativo 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 em 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 POST.

   > **Note**
   >
   > O Amazon Cognito cancela sessões que não recebem uma resposta em 5 minutos e redireciona o usuário para a interface hospedada. Quando seu usuário experimenta esse resultado, ele recebe uma mensagem `Something went wrong` de erro.

6. Depois de verificar a declaração do SAML e mapear os atributos do usuário a partir 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. Seu aplicativo 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 do 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 aplicativo de pool de usuários.

Você também pode escolher a duração dos tokens de atualização. Depois que o token de atualização do usuário expirar, ele deverá entrar novamente. Se eles se autenticaram por meio de um SAML IdP, a duração da sessão de seus usuários é definida pela expiração dos tokens, não pela expiração da sessão com o IdP. Seu aplicativo precisa armazenar o token de atualização de

cada usuário e renovar a sessão quando ela expirar. A interface de usuário hospedada mantém as sessões do usuário em um cookie do navegador válido por 1 hora.

Usando o login SAML iniciado pelo IdP

Ao configurar seu provedor de identidade para login no SAML 2.0 iniciado pelo IdP, você pode apresentar asserções de SAML ao `saml2/idpresponse` endpoint 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 identidade externo do grupo de usuários compatível com o cliente do aplicativo solicitado. As etapas a seguir descrevem o processo geral de configuração e login com um provedor SAML 2.0 iniciado pelo IdP.

1. Crie ou designe um grupo de usuários e um cliente de aplicativo.
2. Crie um IdP SAML 2.0 em seu grupo de usuários.
3. Configure seu IdP para suportar 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 de SAML iniciado pelo IdP a um cliente de aplicativo 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 seu usuário para o `saml2/idpresponse` endpoint do grupo de usuários com a declaração SAML.
7. Receba tokens web JSON (JWTs).

Para aceitar declarações de SAML não solicitadas em seu grupo de usuários, você deve considerar seus efeitos na segurança do seu aplicativo. 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 `InResponseTo` reclamação e deve ter sido emitida nos últimos 6 minutos.

Você deve enviar solicitações com SAML iniciado pelo IdP para o seu. `/saml2/idpresponse` Para solicitações de autorização de interface de usuário hospedadas e iniciadas pelo SP, você deve fornecer parâmetros que identifiquem o cliente do aplicativo solicitado, os escopos, o URI de redirecionamento e outros detalhes como parâmetros da sequência de caracteres de consulta nas solicitações. `HTTP GET` No entanto, para declarações de SAML iniciadas pelo IdP, os detalhes da sua solicitação devem ser formatados como um `RelayState` parâmetro no corpo de uma solicitação. `HTTP POST` O corpo da solicitação também deve conter sua declaração de SAML como parâmetro. `SAMLResponse`

Veja a seguir um exemplo de solicitação 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]
```

AWS Management Console

Para configurar um IdP para SAML iniciado pelo IdP

1. Crie um grupo de usuários, um cliente de aplicativo e um provedor de identidade SAML.
2. Desassocie todos os provedores de identidade social e OIDC do seu cliente de aplicativo, se houver algum associado.
3. Navegue até a guia Experiência de login do seu grupo de usuários.
4. Em Login do provedor de identidade federado, edite ou adicione um provedor SAML.
5. Em Login de SAML iniciado por Idp, escolha Aceitar asserções de SAML iniciadas pelo SP e iniciadas pelo IdP.
6. Escolha Salvar alterações.

API/CLI

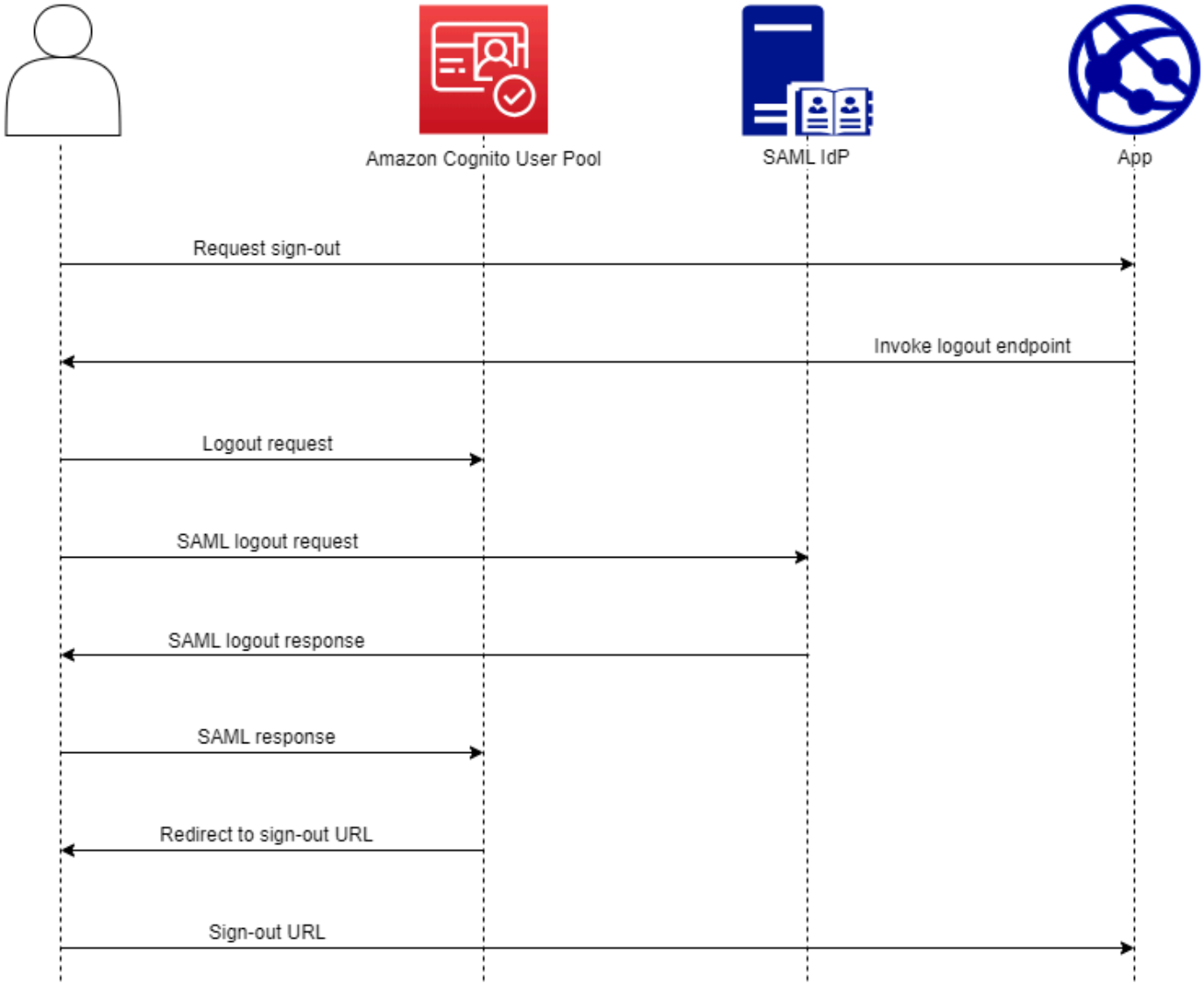
Para configurar um IdP para SAML iniciado pelo IdP

Configure o SAML iniciado pelo IdP com o `IDPInit` parâmetro em uma solicitação de API CreateIdentityProviderou UpdateIdentityProviderAPI. Veja a seguir um exemplo `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"
}
```

Fluxo de saída do SAML

O Amazon Cognito oferece suporte ao logout único do SAML 2.0. Quando você configura seu IdP SAML para suportar o fluxo de saída, 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 da `SingleLogoutService` URL 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 `/logout` endpoint do grupo de usuários, o Amazon Cognito redireciona seu usuário do SAML com a seguinte solicitação para o endpoint do 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 `saml2/logout` endpoint com um `LogoutResponse` de seu IdP. Seu IdP deve enviar uma `LogoutResponse` `HTTP POST` solicitação. Em seguida, o Amazon Cognito os redireciona para o destino de redirecionamento a partir da solicitação inicial de desconexão.

Seu provedor de SAML pode enviar um `LogoutResponse` com mais `AuthnStatement` de um. O primeiro `sessionIndex` `AuthnStatement` em uma resposta desse tipo deve corresponder ao da `sessionIndex` 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.

AWS Management Console

Para configurar a saída do SAML

1. Crie um grupo de usuários, um cliente de aplicativo e um SAML IdP.
2. Ao criar ou editar seu provedor de identidade SAML, em Informações do provedor de identidade, marque a caixa com o título Adicionar fluxo de saída.
3. Na guia Experiência de login do seu grupo de usuários, em Login do provedor de identidade federado, escolha seu IdP e localize o certificado de assinatura.
4. Escolha Baixar como .crt.
5. Configure seu provedor de 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 o domínio do grupo `/saml2/logout` de usuários.

API/CLI

Para configurar a saída do SAML

Configure o logout único com o `IDPSignout` parâmetro de uma solicitação de UpdateIdentityProviderAPI CreateIdentityProviderou. Veja a seguir um exemplo `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 Amazon Cognito oferece suporte a solicitações de SAML assinadas e respostas de SAML criptografadas para login e saída. Todas as operações criptográficas durante as operações de SAML do grupo de usuários devem gerar assinaturas e texto cifrado com as chaves geradas pelo user-pool-provided Amazon Cognito. Atualmente, você não pode configurar um grupo de usuários para assinar solicitações ou aceitar afirmações criptografadas com 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 asserções SAML com qualquer certificado de criptografia de grupo de usuários que não tenha expirado. Seus certificados anteriores continuam válidos por toda a duração. Como prática recomendada, atualize o certificado na configuração do seu provedor anualmente.

Tópicos

- Aceitando respostas SAML criptografadas do seu IdP
- Assinatura de solicitações SAML

Aceitando 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 saem. 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, você deve carregar seu certificado em um IdP que suporte respostas SAML criptografadas. A conexão do grupo de usuários com o SAML IdP não funciona antes que o IdP comece a criptografar todas as asserções do SAML com a chave fornecida.

Veja a seguir uma visão geral do fluxo de um login criptografado do SAML.

1. Seu usuário inicia o login e escolhe o SAML IdP.
2. Seu grupo de usuários Autorizar endpoint redireciona seu usuário para o IdP SAML com uma solicitação de login do SAML. Seu grupo de usuários pode, opcionalmente, acompanhar essa solicitação com uma assinatura que permite 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 SAML IdP 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 do grupo de usuários. `/saml2/idpresponse` O IdP deve criptografar a resposta conforme definido pela especificação SAML 2.0. Para obter mais informações, consulte `Element <EncryptedAssertion>` Asserções e protocolos para o OASIS Security Assertion Markup Language (SAML) V2.0.
4. Seu grupo de usuários descriptografa 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, seu 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 de SAML que você configura para oferecer suporte à criptografia.

Qualquer IdP SAML externo em seu grupo de usuários pode suportar criptografia de resposta, e cada IdP recebe seu próprio par de chaves.

AWS Management Console

Para configurar a criptografia de resposta SAML

1. Crie um grupo de usuários, um cliente de aplicativo e um SAML IdP.
2. Ao criar ou editar seu provedor de identidade SAML, em Assinar solicitações e criptografar respostas, marque a caixa com o título Exigir declarações de SAML criptografadas desse provedor.
3. Na guia Experiência de login do seu grupo de usuários, em Login do provedor de identidade federado, selecione seu SAML IdP e escolha Exibir certificado de criptografia.

4. Escolha Baixar como .crt e forneça o arquivo baixado ao seu SAML IdP. Configure seu IdP do 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 `ProviderDetails` de um IdP que oferece suporte à assinatura de solicitações.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

Assinatura de solicitações SAML

A capacidade de provar a integridade das solicitações do SAML 2.0 ao seu IdP é uma vantagem de segurança do login do SAML iniciado pelo Amazon Cognito SP. Cada grupo de usuários com um domínio recebe um certificado de assinatura X.509 do grupo de usuários. 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. Opcionalmente, você pode configurar seu cliente de aplicativo para assinar solicitações de login SAML. Quando você assina suas 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.

AWS Management Console

Para configurar a assinatura da solicitação SAML

1. Crie um grupo de usuários, um cliente de aplicativo e um SAML IdP.
2. Ao criar ou editar seu provedor de identidade SAML, em Assinar solicitações e criptografar respostas, marque a caixa com o título Assinar solicitações SAML para este provedor.

3. Na guia Experiência de login do seu grupo de usuários, em Login do provedor de identidade federado, escolha Exibir certificado de assinatura.
4. Escolha Baixar como .crt e forneça o arquivo baixado ao seu SAML IdP. Configure seu IdP SAML para verificar a assinatura das solicitações SAML recebidas.

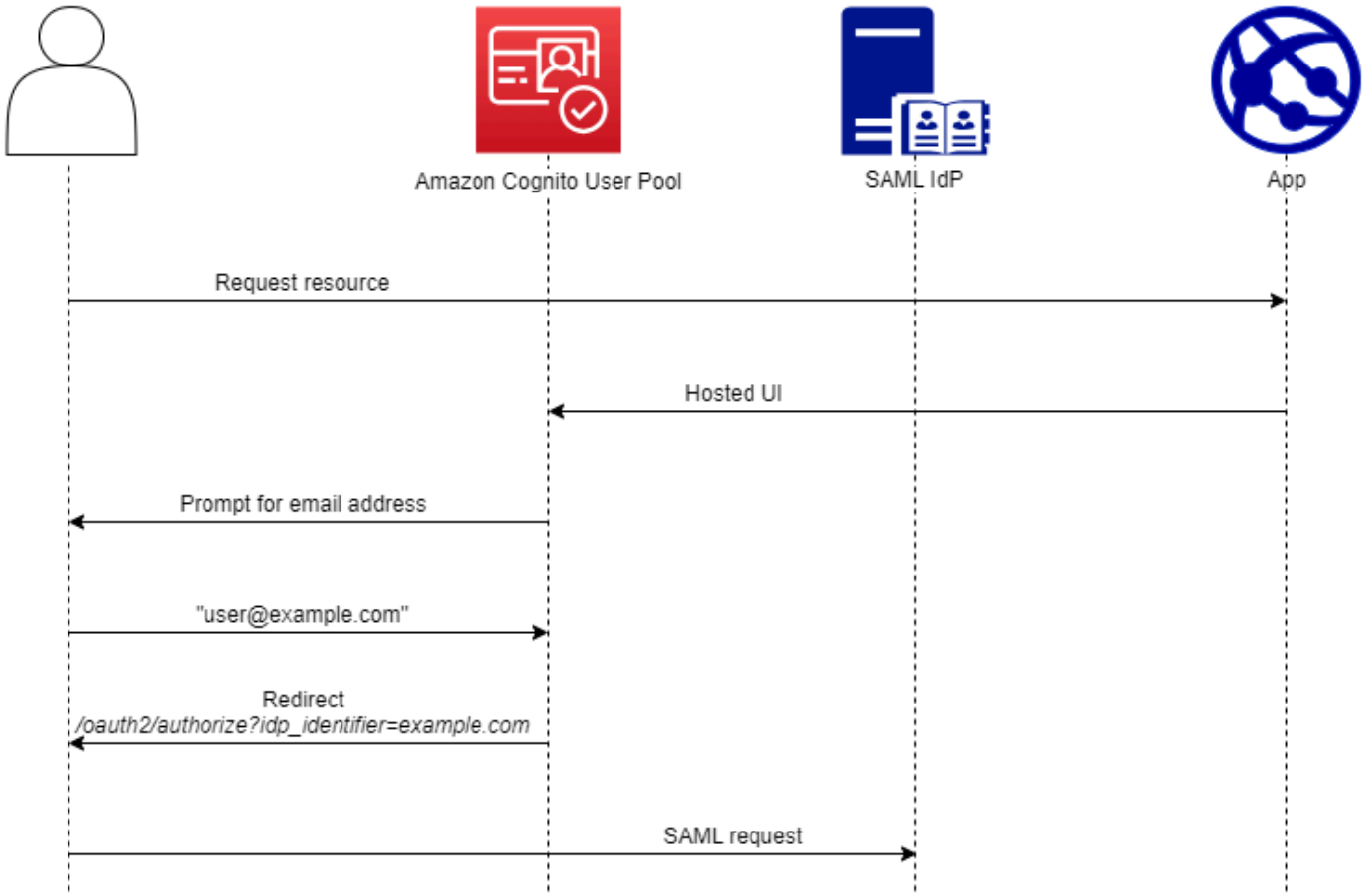
API/CLI

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 `ProviderDetails` de um IdP que oferece suporte à assinatura de solicitações.

```
"ProviderDetails": {
      "MetadataURL" : "https://myidp.example.com/saml/metadata",
      "IDPSignout" : "true",
      "RequestSigningAlgorithm" : "rsa-sha256",
      "EncryptedResponses" : "true",
      "IDPInit" : "true"
}
```

Nomes e identificadores do provedor de identidade 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 seus provedores de SAML. Um identificador é um nome amigável para um IdP em seu grupo de usuários e deve ser exclusivo dentro do grupo de usuários. Se seus identificadores SAML corresponderem aos domínios de e-mail dos seus usuários, a interface de usuário hospedada do Amazon Cognito 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 possuir vários domínios, um único IdP pode ter vários identificadores.

Se você usa ou não identificadores de domínio de e-mail, você pode usar identificadores em um aplicativo multilocatário para redirecionar os usuários para o IdP correto. Quando quiser

ignorar totalmente a interface hospedada, você pode personalizar os links que você apresenta aos usuários para que eles sejam redirecionados Autorizar endpoint diretamente para o IdP. Para cadastrar seus usuários com um identificador e redirecionar para o IdP, inclua o identificador no `idp_identifier=myidp.example.com` formato 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 seu IdP no seguinte formato de URL.

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&
identity_provider=MySAMLIdP&
client_id=1example23456789&
redirect_uri=https://www.example.com
```

Depois que um usuário faz login com seu IdP SAML, seu IdP o redireciona com uma resposta SAML no corpo para seu endpoint. `HTTP POST /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 do aplicativo. Depois que seu usuário tiver concluído a autenticação dessa forma, ele interagirá com páginas da Web somente para seu IdP e seu aplicativo.

Com identificadores de IdP em formato de domínio, a interface de usuário hospedada do Amazon Cognito 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 um aplicativo 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`.

Em seu aplicativo, invoque a interface de usuário hospedada para seu cliente de aplicativo para solicitar que cada usuário insira seu endereço de e-mail. O Amazon Cognito deriva 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 o que contém um parâmetro de solicitação. Autorizar endpoint `idp_identifier` Por exemplo, se um usuário

entrarbob@exampleA.co.uk, a próxima página com a qual ele interage é a página de login do IdP em. https://auth.exampleA.co.uk/sso/saml

Você também pode implementar a mesma lógica de forma independente. No seu aplicativo, 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 de aplicativos personalizados para cada um dos locatários do seu aplicativo, onde 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 na interface hospedada, atribua pelo menos um identificador a cada IdP do SAML que você atribuiu ao seu cliente do aplicativo. Por padrão, a tela de login da IU hospedada exibe um botão para cada um dos IdPs que você atribuiu ao seu cliente de aplicativo. No entanto, se você atribuiu identificadores com sucesso, sua página de login da interface de usuário hospedada será semelhante à imagem a seguir.

A análise de domínio na interface de usuário hospedada exige que você use domínios como seus identificadores de IdP. Se você atribuir um identificador de qualquer tipo a cada SAML IdPs de um cliente de aplicativo, a interface de usuário hospedada 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 sua interface hospedada exiba uma lista de IdPs, não atribua identificadores e use o parâmetro de `identity_provider` solicitação em suas solicitações de autorização.

- Se você atribuir apenas um IdP SAML ao cliente de aplicação, a página de login da interface do usuário hospedada exibirá um botão para fazer login com esse IdP.
- Se você atribuir um identificador a cada IdP SAML ativado para seu cliente de aplicativo, uma solicitação de entrada do usuário para um endereço de e-mail aparecerá na página de login da interface hospedada.
- Se você tiver vários IdPs e não atribuir um identificador a todos eles, a página de login da IU hospedada exibirá um botão para entrar com cada IdP atribuído.
- Se você atribuiu identificadores ao seu IdPs e deseja que sua interface hospedada exiba uma seleção de botões de IdP, adicione um novo IdP que não tenha identificador ao seu cliente de aplicativo ou crie um novo cliente de aplicativo. Você também pode excluir um IdP existente e adicioná-lo novamente sem um identificador. Se você criar um novo IdP, seus usuários do SAML criarão novos perfis de usuário. Essa duplicação de usuários ativos pode ter um impacto no faturamento 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.

Configurando seu provedor de identidade SAML terceirizado

Para configurar soluções de provedor de identidade (IdP) SAML 2.0 de terceiros para trabalhar com federação para grupos de usuários do Amazon Cognito, você deve configurar seu IdP SAML para redirecionar para a seguinte URL do Assertion Consumer Service (ACS):. `https://mydomain.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 na guia App integration (Integração de aplicações) 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 seu ID do grupo de usuários em Visão geral do grupo de usuários no console do Amazon Cognito.

Você também deve configurar seu SAML IdP para fornecer valores para quaisquer atributos que você designou como atributos obrigatórios em seu grupo de usuários. Normalmente, `email` é um atributo obrigatório para grupos de usuários. Nesse caso, o IdP do SAML deve fornecer alguma forma de `email` declaração em sua declaração de SAML, e você deve mapear a declaração para o atributo desse provedor.

As informações de configuração a seguir para soluções de IdP SAML 2.0 de terceiros são um bom lugar para começar a configurar a federação com grupos de usuários do Amazon Cognito. Para obter as informações mais atuais, consulte diretamente a documentação do seu provedor.

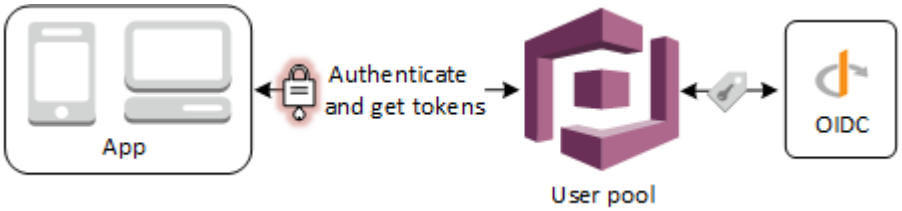
Para assinar solicitações SAML, você deve configurar seu IdP para confiar nas solicitações assinadas pelo certificado de assinatura do grupo de usuários. Para aceitar respostas SAML criptografadas, você deve configurar seu IdP para criptografar todas as respostas SAML para seu grupo de usuários. Seu provedor terá 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 identidade. Seu provedor pode oferecer informações de configuração para Conta da AWS federação com o SAML 2.0; essas informações não são relevantes para a integração com o Amazon Cognito.

Solução	Mais informações
Microsoft Active Directory Federation Services (AD FS)	Explorador de metadados da federação
Okta	Como baixar os metadados do IdP e os certificados de assinatura SAML para uma integração do aplicativo SAML
Auth0	Configurar Auth0 como provedor de identidade SAML
Identidade de ping (PingFederate)	Exportação de metadados SAML de PingFederate
JumpCloud	Notas de configuração do SAML
SecureAuth	Integração de aplicativos SAML

Usando provedores de identidade OIDC com um grupo de usuários

Você pode permitir que seus usuários que já têm contas com provedores de identidade () do OpenID Connect (OIDCIdPs) pulem a etapa de inscrição e acessem seu aplicativo usando uma conta existente. 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.



> **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).

Você pode adicionar um IdP OIDC ao seu grupo de usuários no AWS Management Console, por meio do ou com o método da AWS CLI API do grupo de usuários. CreateIdentityProvider

Tópicos

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.

Etapa 1: registrar com um IdP OIDC

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

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	Instale um provedor de identidade Salesforce	https://login.salesforce.com
Ping Identity	Instale um provedor de identidade Ping Identity	https://*Seu endereço de domínio Ping*:9031/idp/userinfo.openid Por exemplo: `https://pf.company.com:9031/idp/userinfo.openid`
Okta	Instale um provedor de identidade Okta	https://*Seu subdomínio Okta*.oktapreview.com ou `https://`*`Your Okta subdomain`*`.okta.com`
Microsoft Azure Active Directory (Azure AD)	Instale um provedor de identidade Microsoft AD Azure	`https://login.microsoftonline.com/`*`{tenant}`*`/v2.0`
Google	Instale um provedor de identidade Google	https://accounts.google.com **Note** O Amazon Cognito oferece o Google como um IdP de login social

IdP OIDC	Como instalar	URL de descoberta OIDC
		integrado. Recomenda mos que você use o IdP integrado. Consulte Usando provedores de identidade social com um grupo de usuários.

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.us-east-1.amazoncognito.com/oauth2/idpresponse
```

3. Inscreva o URL de retorno de chamada no grupo de usuários do Amazon Cognito. Esse é o URL da página para a qual o Amazon Cognito redireciona o usuário após uma autenticação bem-sucedida.

```
https://www.example.com
```

4. Selecione seus escopos. O escopo openid é obrigatório. O escopo email é necessário para conceder acesso às solicitações email e email_verified.
5. O IdP OIDC fornece um ID e uma chave secreta do cliente. Você vai usá-las ao configurar um IdP OIDC no 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.us-east-1.amazoncognito.com/authorize?
   response_type=code&client_id=<your_client_id>&redirect_uri=https://
   www.example.com&identity_provider=CorpSalesforce
   ```

   c. Habilite OAuth settings (Configurações OAuth) e insira o URL do endpoint `/oauth2/idpresponse` para o domínio do grupo de usuários no Callback URL (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 do OAuth.

   ```
   https://mydomain.us-east-1.amazoncognito.com/oauth2/idpresponse
   ```

9. Selecione seus escopos. Você deve incluir o escopo openid. Para conceder acesso às solicitações email e email_verified, adicione o escopo email. Escopos separados por espaços.
10. Selecione Create (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.

Etapa 2: adicionar um IdP OIDC ao seu grupo de usuários

Nesta seção, você configura o grupo de usuários para processar solicitações de autenticação baseadas em OIDC de um IdP OIDC.

Para adicionar um IdP OIDC (console do Amazon Cognito)

Adicionar um IdP OIDC

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. Escolha a guia Sign-in experience (Experiência de acesso). Localize Federated sign-in (Acesso federado) e selecione Add an identity provider (Adicionar um provedor de identidade).
5. Escolha um IdP 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 receberá uma solicitação de consentimento para o fornecimento desses atributos à aplicação.
10. Selecione um Attribute request method (Método de solicitação de atributos) para fornecer ao Amazon Cognito o método HTTP (GET ou POST) que ele deve usar 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 Auto fill through issuer URL (Preenchimento automático por meio do URL emissor) ou Manual input (Entrada manual). Use o endpoint Auto fill through issuer URL (Preenchimento automático por meio do URL do emissor) quando o Amazon Cognito puder recuperar os URLs dos endpoints `authorization`, `token`, `userInfo` e `jwks_uri`.
12. Insira o URL emissor ou os URLs dos endpoints `authorization`, `token`, `userInfo` e `jwks_uri` do seu IdP.

> **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.

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 escolha o atributo do 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 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.
15. Selecione Create (Criar).
16. Na guia App client integration (Integração de cliente da aplicação), escolha um dos App clients (Clientes da aplicação) na lista e escolha Edit hosted UI settings (Editar configurações da interface do usuário hospedada). Adicione o novo IdP OIDC ao cliente de aplicação em Identity providers (Provedores de identidade).
17. Escolha Salvar alterações.

Para adicionar um IdP OIDC (AWS CLI)

- Veja as descrições dos parâmetros do método CreateIdentityProviderda API.

```
aws cognito-idp create-identity-provider
--user-pool-id string
--provider-name string
--provider-type OIDC
--provider-details map
```

```
--attribute-mapping string
--idp-identifiers (list)
--cli-input-json string
--generate-cli-skeleton string
```

Use este mapa de detalhes do provedor:

```
{
  "client_id": "string",
  "client_secret": "string",
  "authorize_scopes": "string",
  "attributes_request_method": "string",
  "oidc_issuer": "string",

  "authorize_url": "string",
  "token_url": "string",
  "attributes_url": "string",
  "jwks_uri": "string"
}
```

Etapa 3: testar a configuração de IdP OIDC

Você pode criar o URL de autorização usando os elementos das duas seções anteriores, e usá-lo para testar sua configuração de IdP OIDC.

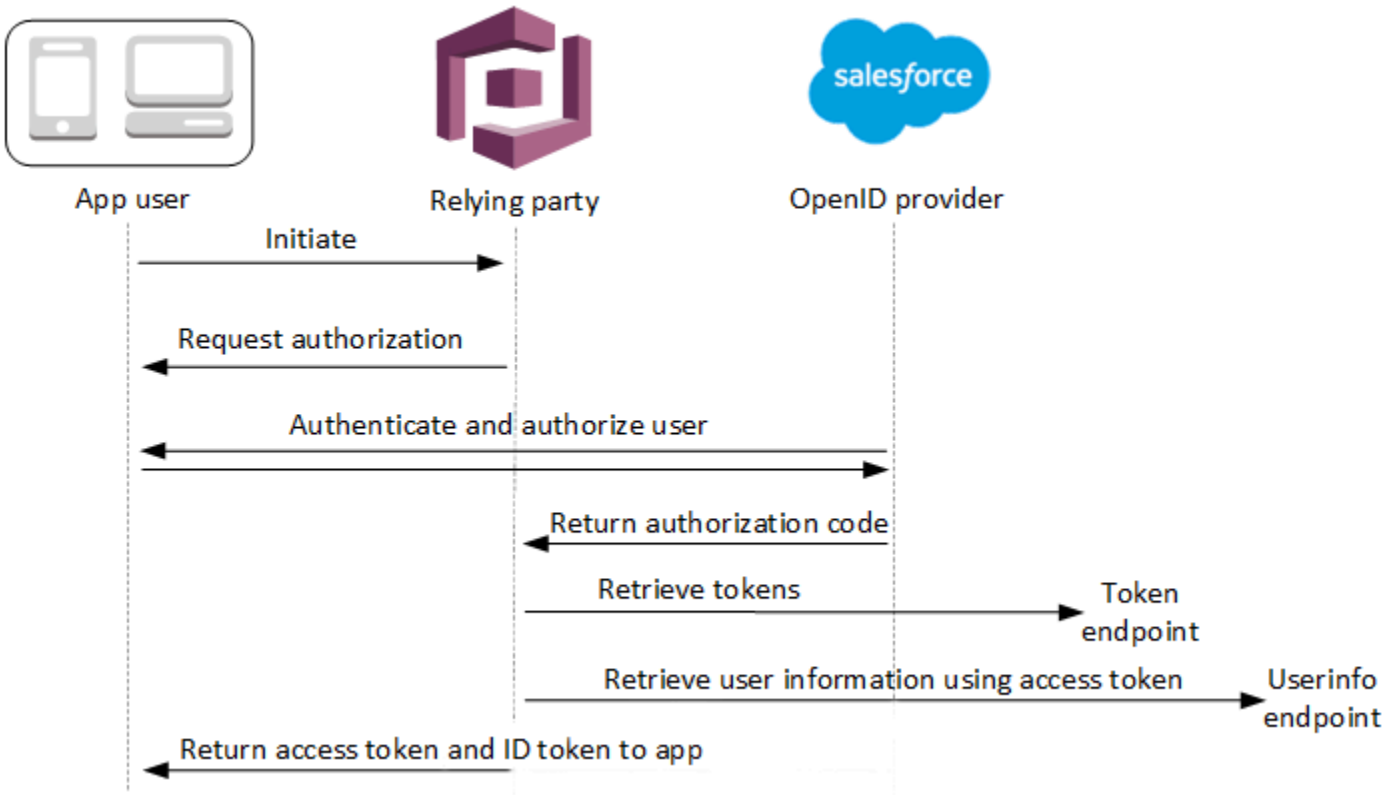
```
https://mydomain.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&client_id=1example23456789&redirect_uri=https://www.example.com
```

Você pode encontrar o domínio na página do console Domain name (Nome do domínio) do grupo de usuários. O client_id está na página General settings (Configurações gerais). 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.

Fluxo de autenticação do IdP do grupo de usuários do OIDC

Quando o usuário acessa sua aplicação usando um IdP OIDC, ele passa pelo seguinte fluxo de autenticação.

1. O usuário acessa a página de login integrada do Amazon Cognito e vê a opção de fazer login por meio de um IdP OIDC, como o Salesforce.
2. O usuário é redirecionado ao endpoint `authorization` do IdP OIDC.
3. Depois que o usuário é autenticado, o IdP OIDC é redirecionado para o Amazon Cognito com um código de autorização.
4. O Amazon Cognito troca o código de autorização com o IdP OIDC por um token de acesso.
5. O Amazon Cognito cria ou atualiza a conta do usuário no grupo de usuários.
6. O Amazon Cognito emite tokens do portador da aplicação, o que pode incluir tokens de identidade, acesso e atualização.



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

O OIDC é uma camada de identidade sobre o OAuth 2.0, que especifica tokens de identidade formatados em JSON (JWT) que são emitidos pelos aplicativos clientes do OIDC (partes confiáveis). IdPs 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 o gerenciamento de identidades, 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 de OIDC de terceiros, a interface de usuário hospedada do Amazon Cognito 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.
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.

Como especificar mapeamentos de atributos do provedor de identidade para seu grupo de usuários

Você pode usar a AWS Management Console, ou a API AWS CLI ou, para especificar mapeamentos de atributos para o provedor de identidade (IdP) do seu grupo de usuários.

Coisas a saber sobre mapeamentos

Antes de começar a configurar o mapeamento de atributos do usuário, revise os detalhes importantes a seguir.

- 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 deriva o `username` atributo no perfil de um usuário federado de declarações específicas aprovadas pelo seu IdP federado, conforme mostrado na tabela a seguir. O Amazon Cognito acrescenta esse valor de atributo ao nome do 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 with Amazon	`user_id`
Fazer login com a Apple	`sub`
Provedores SAML	`NameID`
Provedores OpenID Connect (OIDC)	`sub`

- 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 atualiza cada atributo mapeado, mesmo se o valor atual corresponder às informações mais recentes. 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 Mutable (Mutável) correspondente ao atributo adicionado ao selecionar Add custom attributes (Adicionar atributos personalizados) na guia Sign-up experience (Experiência de inscrição). 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 nivela todos os valores em uma única string delimitada por vírgula e codifica de forma URL os valores que contêm caracteres não alfanuméricos (excluindo os caracteres '', '', 'e' '). `.` `-` `*` `_` Você deve decodificar e analisar os valores individuais antes de usá-los em sua aplicação.

Como especificar mapeamentos de atributos do provedor de identidade para o grupo de usuários (AWS Management Console)

Você pode usar o AWS Management Console 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. Escolha a guia Sign-in experience (Experiência de login) e localize Federated sign-in (Acesso federado).
4. Escolha Add an identity provider (Adicionar um provedor de identidade) ou escolha 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 Usando 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 da coluna ***<provider>*** attribute (atributo do <provedor>). 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. Escolha a guia Sign-in experience (Experiência de login) e localize Federated sign-in (Acesso federado).
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 Usando 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 CLIAWS

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 cotações ("), elas deverão ser precedidas por (\").

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
```

- AWS API: CreateIdentityProvider

Para especificar mapeamentos de atributos para um IdP existente

- AWS CLI: `aws cognito-idp update-identity-provider`

  Exemplo: `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`
- AWS API: UpdateIdentityProvider

Para obter informações sobre o mapeamento de atributos para determinado IdP

- AWS CLI: `aws cognito-idp describe-identity-provider`

  Exemplo: `aws cognito-idp describe-identity-provider --user-pool-id <user_pool_id> --provider-name <provider_name>`
- AWS API: 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 que tenha vários usuários de IdP pode ter uma experiência consistente em sua aplicação. AdminLinkProviderForUserinstrui o Amazon Cognito a reconhecer a ID exclusiva 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 seu usuário federado a qualquer momento antes do primeiro login, em uma solicitação de `AdminLinkProviderForUser` API, em uma tarefa de pré-configuração planejada ou em uma. 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 `identities` atributo 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 ele seja usado 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 do IdPs ADFS1, ADFS2 e 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.
- 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 AWS Management Console.
- 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 OIDC e também confere o endpoint `userInfo` dos provedores do OAuth 2.0 e de OIDC. 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 nos AWS SDKs.

Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda

O Amazon Cognito trabalha com funções do AWS Lambda para modificar o comportamento de autenticação do 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. As funções podem modificar o comportamento-padrão do fluxo de autenticação, fazer solicitações de API para modificar o grupo de usuários ou outros recursos da AWS 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 o grupo de usuários e outro armazenamento de identidades, personalizar mensagens e modificar tokens JSON da web (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 executar operações de runtime no ambiente da AWS, APIs externas, bancos de dados ou armazenamentos de identidades. 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.

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	Operação	Descrição
Fluxo de autenticação personalizado	Definir o desafio de autentica ção	Determina o próximo desafio em um fluxo de autenticação personalizado
	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 “Acionado r do Lambda de pré-auten ticação”	Validação personalizada para aceitar ou negar a solicitação de login
	the section called “Acionado r do Lambda de pós-auten ticação”	Registra eventos para análise personalizada
	the section called “Acionador do Lambda antes da geração do token”	Aumenta ou suprime solicitaç ões de token

Fluxo de grupo de usuários	Operação	Descrição
Cadastrar-se	the section called “Acionador do Lambda de pré-cadastro”	Executa uma validação personalizada que aceita ou nega a solicitação de cadastro
	the section called “Acionador do Lambda de pós-confirmação”	Adiciona mensagens de boas-vindas personalizadas ou registro de eventos para análise personalizada
	the section called “Migrar o acionador do Lambda do usuário”	Migra um usuário de um diretório de usuário existente para grupos de usuários
Mensagens	the section called “Acionador do Lambda de mensagem personalizada”	Realiza personalização avançada e localização de mensagens
Criação de token	the section called “Acionador do Lambda antes da geração do token”	Adiciona ou remove atributos em tokens de ID
Provedores de terceiros de e-mail e SMS	the section called “Acionadores do Lambda remetente personalizado”	Usa um provedor de terceiros para enviar mensagens de e-mail e SMS

Tópicos

- Considerações importantes
- Como 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
- 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
- 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

Considerações importantes

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 Definição de 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 atributos de segurança avançados e atualizar a configuração do gatilho 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 a chamada puder ser repetida, o Amazon Cognito tentará novamente. Após três tentativas malsucedidas, a função encerra a sessão. Não é possível alterar esse valor de tempo limite de cinco segundos. Para obter mais informações, consulte Modelo de programação do Lambda no Guia do desenvolvedor do AWS Lambda.

  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 Lidar com erros e novas tentativas automáticas no 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, é possível associar uma versão da função a um alias e definir o acionador `LambdaArn` para o ARN do alias em uma solicitação de API CreateUserPool ou UpdateUserPool. Essa opção não está disponível no AWS Management Console. 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.

  A interface de usuário hospedada do Amazon Cognito 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()` para registrar qualquer informação confidencial ou de depuração no CloudWatch Logs. 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 do Lambda a outra Conta da AWS como um acionador para seu grupo de usuários. Você deve adicionar acionadores entre contas com as operações da API CreateUserPool e UpdateUserPool ou seus equivalentes no AWS CloudFormation e na AWS CLI. Você não pode adicionar funções entre contas no AWS Management Console.
- 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. É possível adicionar permissões do console do Lambda ou usar a operação de API AddPermission do Lambda.

  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`.

```
{
    "Version": "2012-10-17",
    "Id": "default",
    "Statement": [
        {
            "Sid": "lambda-allow-cognito",
            "Effect": "Allow",
            "Principal": {
                "Service": "cognito-idp.amazonaws.com"
            },
            "Action": "lambda:InvokeFunction",
            "Resource": "<your Lambda function ARN>",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "<your account number>"
                },
                "ArnLike": {
                    "AWS:SourceArn": "<your user pool ARN>"
                }
            }
        }
    ]
}
```

Como 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. Escolha a guia User pool properties (Propriedades do grupo de usuários) e localize Lambda triggers (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 Assign Lambda function (Atribuir função do Lambda) e selecione uma função na mesma Região da AWS do seu grupo de usuários.

> **Note**
>
> Se suas credenciais do AWS Identity and Access Management (IAM) tiverem permissão para atualizar a função do 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 “Considerações importantes”.

8. Selecione Save changes.
9. Você pode usar o CloudWatch no console do Lambda para registrar sua função do Lambda. Para obter mais informações, consulte Acessar o CloudWatch Logs 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. Esse evento mostra os parâmetros comuns do acionador do Lambda:

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

A Região da AWS como uma instância da `AWSRegion`.

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 awsSdkVersion e 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"`.

Retorno

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.

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 por meio da API de grupos de usuários do Amazon Cognito, da interface do usuário hospedada 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 Usar a API de grupos de usuários e endpoints de grupo de usuários do Amazon Cognito. 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 na interface do usuário hospedada
- 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`
	Geração de pré-token	`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	Pós-confirmação	`PostConfirmation_ConfirmSignUp`
InitiateAuth AdminInitiateAuth	Pre authentication	`PreAuthentication_Authentication`

Operação de API	Gatilho do Lambda	Fonte de gatilhos
	Definir o desafio de autenticação	`DefineAuthChallenge_Authentication`
	Criar o desafio de autenticação	`CreateAuthChallenge_Authentication`

	Geração de pré-token	`TokenGeneration_Authentication`
		`TokenGeneration_AuthenticateDevice`
		`TokenGeneration_RefreshTokens`

	Migração de usuário	`UserMigration_Authentication`
	Mensagem personalizada	`CustomMessage_Authentication`
	Remetente de e-mail personalizado	`CustomEmailSender_AccountTakeOverNotification`
	Remetente de SMS personalizado	`CustomSMSSender_Authentication`
ForgotPassword	Migração de usuário	`UserMigration_ForgotPassword`
	Mensagem personalizada	`CustomMessage_ForgotPassword`
	Remetente de e-mail personalizado	`CustomEmailSender_ForgotPassword`

Operação de API	Gatilho do Lambda	Fonte de gatilhos
	Remetente de SMS personalizado	`CustomSMSSender_ForgotPassword`
ConfirmForgotPassword	Pós-confirmação	`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`

Acionadores do Lambda para usuários locais do Amazon Cognito na interface do usuário hospedada

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 a interface do usuário hospedada.

Fontes locais de gatilho de usuário na interface do usuário hospedada

URI de interface hospedada	Gatilho do Lambda	Fonte de gatilhos
`/signup`	Pré-cadastro	`PreSignUp_SignUp`

URI de interface hospedada	Gatilho do Lambda	Fonte de gatilhos
	Mensagem personalizada	`CustomMessage_SignUp`
	Remetente de e-mail personalizado	`CustomEmailSender_SignUp`
	Remetente de SMS personalizado	`CustomSMSSender_SignUp`
`/confirmuser`	Pós-confirmação	`PostConfirmation_ConfirmSignUp`
`/login`	Pre authentication	`PreAuthentication_Authentication`
	Definir o desafio de autenticação	`DefineAuthChallenge_Authentication`
	Criar o desafio de autenticação	`CreateAuthChallenge_Authentication`

	Geração de pré-token	`TokenGeneration_Authentication`
		`TokenGeneration_AuthenticateDevice`
		`TokenGeneration_RefreshTokens`

	Migração de usuário	`UserMigration_Authentication`
	Mensagem personalizada	`CustomMessage_Authentication`
	Remetente de e-mail personalizado	`CustomEmailSender_AccountTakeOverNotification`

URI de interface hospedada	Gatilho do Lambda	Fonte de gatilhos
	Remetente de SMS personalizado	`CustomSMSSender_Authentication`
`/forgotpassword`	Migração de usuário	`UserMigration_ForgotPassword`
	Mensagem personalizada	`CustomMessage_ForgotPassword`
	Remetente de e-mail personalizado	`CustomEmailSender_ForgotPassword`
	Remetente de SMS personalizado	`CustomSMSSender_ForgotPassword`
`/confirmforgotpassword`	Pós-confirmação	`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 a interface de usuário hospedada do Amazon Cognito 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`
	Pós-confirmação	`PostConfirmation_ConfirmSignUp`
	Geração de pré-token	`TokenGeneration_HostedAuth`
Logins subsequentes	Pre authentication	`PreAuthentication_Authentication`
	Post authentication	`PostAuthentication_Authentication`
	Geração de pré-token	`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 função do LambdaSource	Evento
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.
Pós-confirmação	`PostConfirmation_ConfirmSignUp`	Confirmação pós-cadastro.
Pós-confirmação	`PostConfirmation_ConfirmForgotPassword`	Confirmação após esquecimento de senha.
Pre authentication	`PreAuthentication_Authentication`	Pré-autenticação.
Post authentication	`PostAuthentication_Authentication`	Pós-autenticação.

Acionadores de desafio de autenticação personalizado

Trigger	Valor função do LambdaSource	Evento
Definir o desafio de autenticação	`DefineAuthChallenge_Authentication`	Definir o desafio de autenticação.

Trigger	Valor função do LambdaSource	Evento
Criar o desafio de autenticação	`CreateAuthChallenge_Authentication`	Criar desafio de autenticação.
Verificar desafio de autenticação	`VerifyAuthChallengeResponse_Authentication`	Verificar a resposta do desafio de autenticação.

Acionadores de geração de pré-token

Trigger	Valor função do LambdaSource	Evento
Geração de pré-token	`TokenGeneration_HostedAuth`	O Amazon Cognito autentica o usuário na página de login da interface do usuário hospedada.
Geração de pré-token	`TokenGeneration_Authentication`	Fluxos completos de autenticação do usuário.
Geração de pré-token	`TokenGeneration_NewPasswordChallenge`	O administrador cria o usuário. O Amazon Cognito invoca isso quando o usuário precisa alterar uma senha temporária.
Geração de pré-token	`TokenGeneration_AuthenticateDevice`	Final da autenticação do dispositivo de um usuário.
Geração de pré-token	`TokenGeneration_RefreshTokens`	O usuário tenta atualizar a identidade e acessar tokens.

Acionadores de migração do usuário

Trigger	Valor função do LambdaSource	Evento
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 função do LambdaSource	Evento
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

Trigger	Valor função do LambdaSource	Evento
		endereço de e-mail ou número de telefone e o Amazon 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.

Acionador do Lambda de pré-cadastro

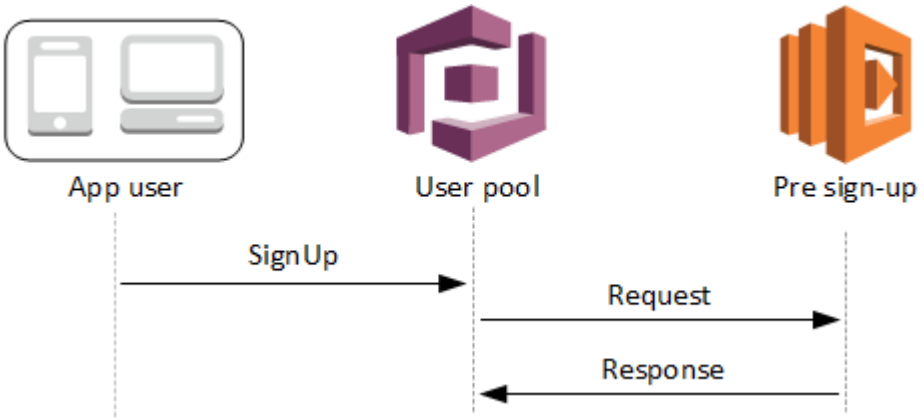
Um pouco antes de o Amazon Cognito inscrever um novo usuário, ele ativa a função de pré-cadastro do AWS Lambda. Como parte do processo de cadastro, você pode usar essa função para executar a validação personalizada e, com base nos resultados de sua validação, aceitar ou negar a solicitação de registro.

Tópicos

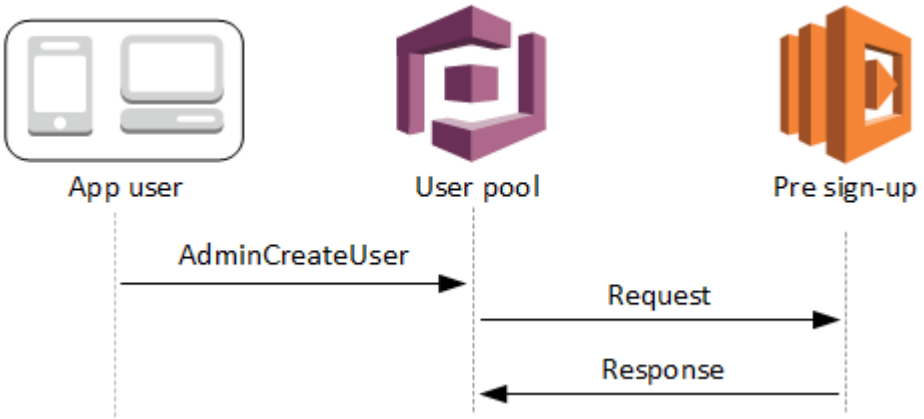
- Fluxos do Lambda de pré-cadastro
- Parâmetros do acionador do Lambda de pré-cadastro
- Tutoriais de cadastro
- Exemplo de pré-cadastro: confirmação automática de usuários em um domínio registrado
- Exemplo de pré-cadastro: confirmação e verificação automáticas de todos os usuários
- Exemplo de pré-cadastro: negar cadastro se o nome de usuário tiver menos de cinco caracteres

Fluxos do Lambda de pré-cadastro

Fluxo de cadastro do cliente



Fluxo de cadastro do servidor



A solicitação inclui dados de validação do cliente. Esses dados vêm dos `ValidationData` valores passados para o grupo de usuários SignUp e os métodos AdminCreateUser da API.

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 ter 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 ter mais informações, consulte Personalização dos atributos de login.

Tutoriais de cadastro

A função Lambda de pré-cadastro é acionada antes que o usuário se cadastre. Veja esses tutoriais de inscrição no Amazon Cognito para JavaScript Android e iOS.

Plataforma	Tutorial
JavaScript SDK de identidade	Inscrever usuários com JavaScript
SDK de identidade para Android	Inscrever usuários com Android
SDK de identidade para iOS	Inscrever usuários com iOS

Exemplo de pré-cadastro: confirmação automática de usuários em um domínio registrado

Você pode usar o acionador do Lambda de pré-cadastro para adicionar lógica personalizada para validar novos usuários que se cadastram em seu grupo de usuários. Este é um exemplo de JavaScript programa que mostra como inscrever um novo usuário. Ele invoca um acionador do Lambda de pré-cadastro como parte da autenticação.

JavaScript

```
var attributeList = [];
var dataEmail = {
  Name: "email",
  Value: "...", // your email here
};
var dataPhoneNumber = {
```

```
  Name: "phone_number",
  Value: "...", // your phone number here with +country code and no delimiters in
 front
};

var dataEmailDomain = {
  Name: "custom:domain",
  Value: "example.com",
};
var attributeEmail = new AmazonCognitoIdentity.CognitoUserAttribute(dataEmail);
var attributePhoneNumber = new AmazonCognitoIdentity.CognitoUserAttribute(
  dataPhoneNumber
);
var attributeEmailDomain = new AmazonCognitoIdentity.CognitoUserAttribute(
  dataEmailDomain
);

attributeList.push(attributeEmail);
attributeList.push(attributePhoneNumber);
attributeList.push(attributeEmailDomain);

var cognitoUser;
userPool.signUp(
  "username",
  "password",
  attributeList,
  null,
  function (err, result) {
    if (err) {
      alert(err);
      return;
    }
    cognitoUser = result.user;
    console.log("user name is " + cognitoUser.getUsername());
  }
);
```

Esse é um exemplo de acionador do Lambda chamado logo antes do cadastro com o acionador do Lambda de pré-cadastro do grupo de usuários. 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

```
exports.handler = (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

```
const handler = async (event) => {
  // 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 event;
};

export { handler };
```

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

```
exports.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.

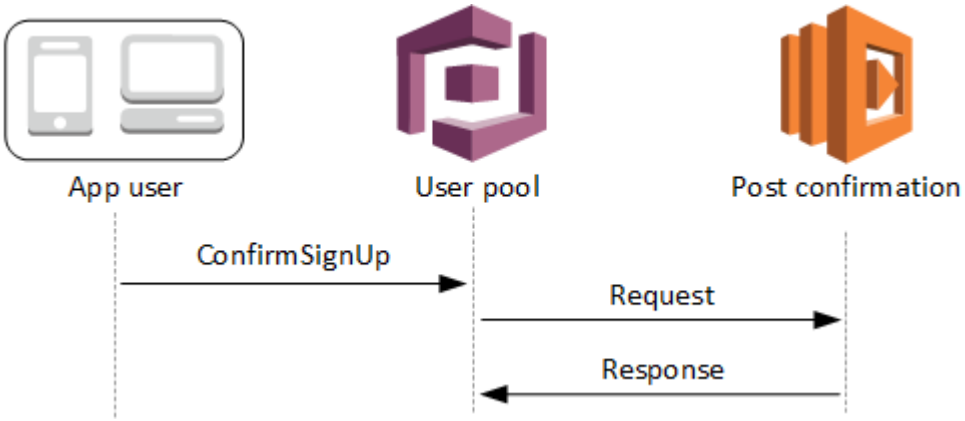
A solicitação contém os atributos atuais do usuário confirmado.

Tópicos

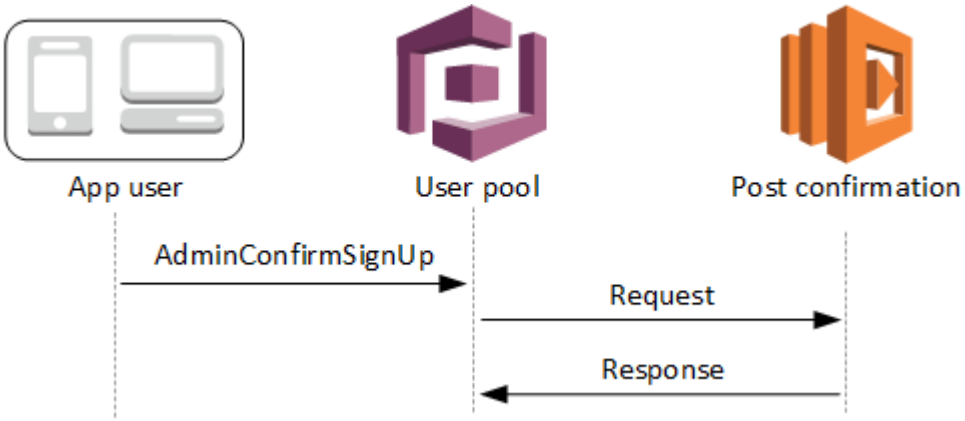
- Fluxos do Lambda de pós-confirmação
- Parâmetros do acionador do Lambda de pós-confirmação
- Tutoriais de confirmação de usuário
- Exemplo de pós-confirmação

Fluxos do Lambda de pós-confirmação

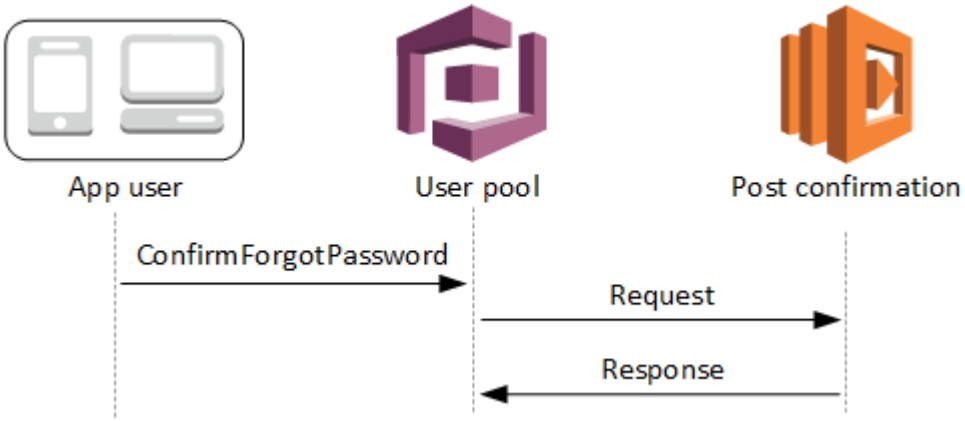
Fluxo de confirmação de inscrição do cliente



Fluxo de confirmação de cadastro do servidor



Fluxo de confirmação de senha esquecida



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. É possível passar esses dados para a função Lambda usando o parâmetro ClientMetadata nas seguintes ações de API: AdminConfirmSignUp, ConfirmForgotPassword, ConfirmSignUp e SignUp.

Parâmetros de resposta de pós-confirmação

Nenhuma informação de retorno adicional é esperada na resposta.

Tutoriais de confirmação de usuário

A função Lambda de pós-confirmação é acionada logo depois que o Amazon Cognito confirma um novo usuário. Consulte esses tutoriais de confirmação do usuário para JavaScript, Android e iOS.

Plataforma	Tutorial
SDK de identidade para JavaScript	Confirmar usuários com JavaScript
SDK de identidade para Android	Confirmar usuários com Android
SDK de identidade para iOS	Confirmar usuários com iOS

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": {}
}
```

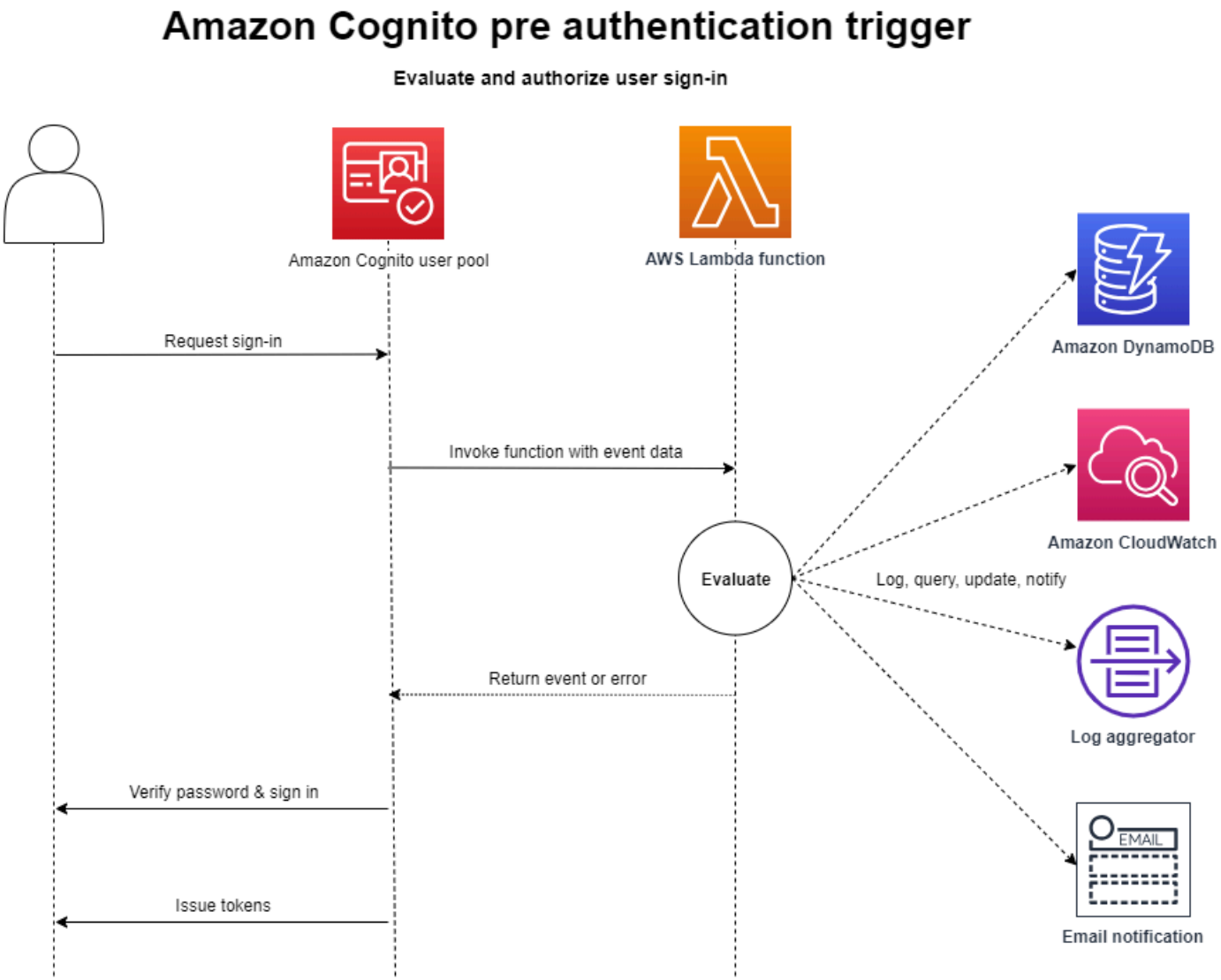
Acionador do Lambda de pré-autenticação

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 ou já tem uma sessão existente no grupo de usuários. Se a configuração `PreventUserExistenceErrors` de um cliente de aplicação do grupo de usuários estiver definida como `ENABLED`, o acionador do Lambda será ativado.

Tópicos

Visão geral do fluxo de autenticação



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 Fluxo de autenticação de grupo de usuários.

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 transmitir esses dados para a função do Lambda, use o parâmetro ClientMetadata nas ações de API InitiateAuth e AdminInitiateAuth.

Parâmetros de resposta de pré-autenticação

O Amazon Cognito não espera nenhuma outra informação de retorno 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": {}
}
```

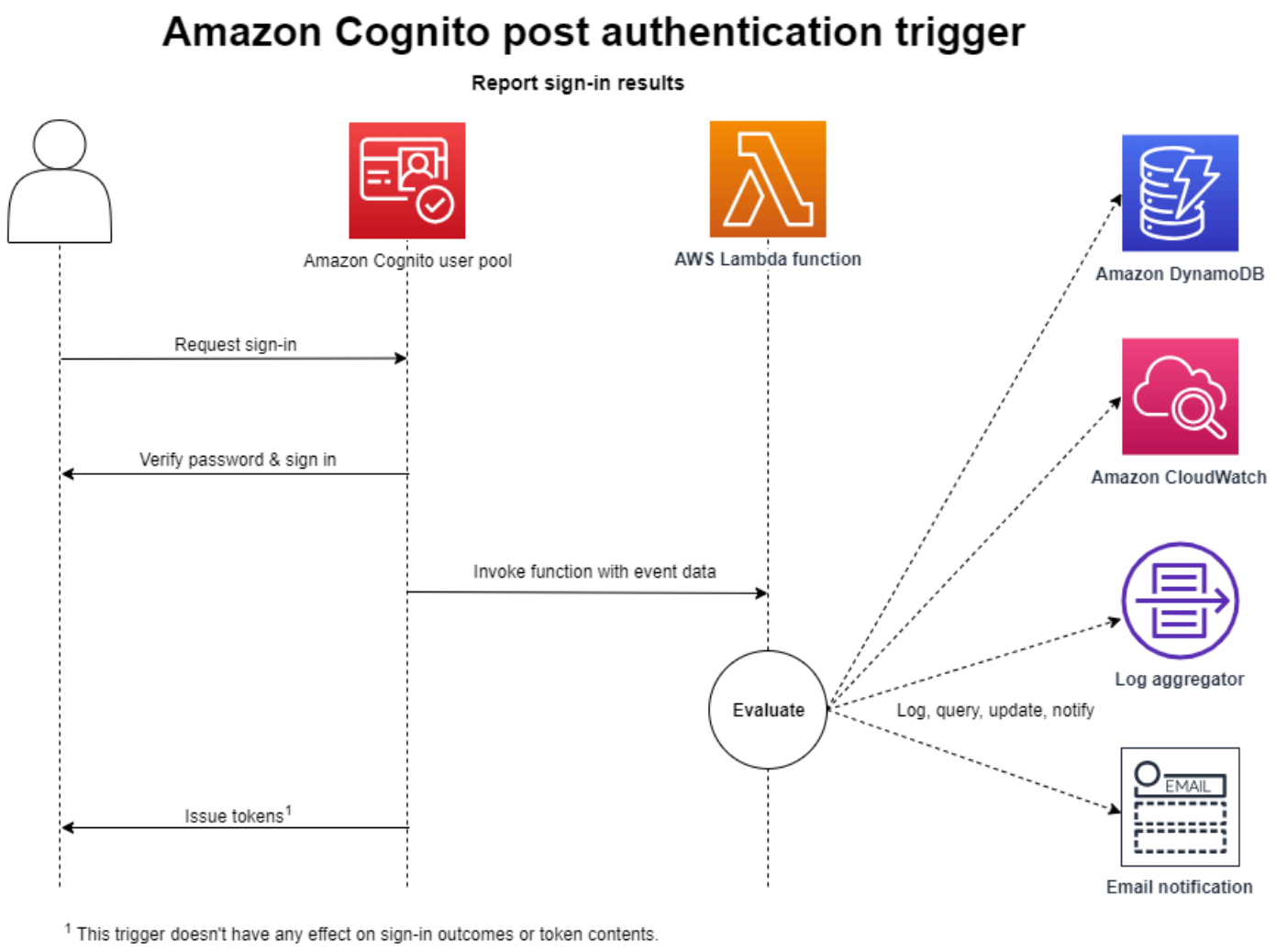
Acionador do Lambda de pós-autenticação

Como o Amazon Cognito invoca esse acionador depois de fazer login de um usuário, você pode adicionar lógica personalizada assim que o Amazon Cognito autenticar o usuário.

Tópicos

- Visão geral do fluxo de autenticação
- Parâmetros do acionador do Lambda de pós-autenticação
- Tutoriais de autenticação
- Exemplo de pós-autenticação

Visão geral do fluxo de autenticação



Para obter mais informações, consulte Fluxo de autenticação de grupo de usuários.

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 transmitir esses dados para sua função do Lambda, é possível usar o parâmetro ClientMetadata nas ações de API AdminRespondToAuthChallenge e RespondToAuthChallenge. O Amazon Cognito não inclui dados do parâmetro ClientMetadata nas operações AdminInitiateAuth e InitiateAuth da API na solicitação que ele transmite 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.

Tutoriais de autenticação

Imediatamente depois que o Amazon Cognito faz login de um usuário, ele ativa a função do Lambda de pós-autenticação. Consulte esses tutoriais de login para JavaScript, Android e iOS.

Plataforma	Tutorial
SDK de identidade para JavaScript	Fazer login de usuários com JavaScript
SDK de identidade para Android	Fazer login de usuários com Android
SDK de identidade para iOS	Fazer login de usuários com iOS

Exemplo de pós-autenticação

Esse exemplo de função Lambda de pós-autenticação envia dados de um login bem-sucedido para o CloudWatch Logs.

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
```

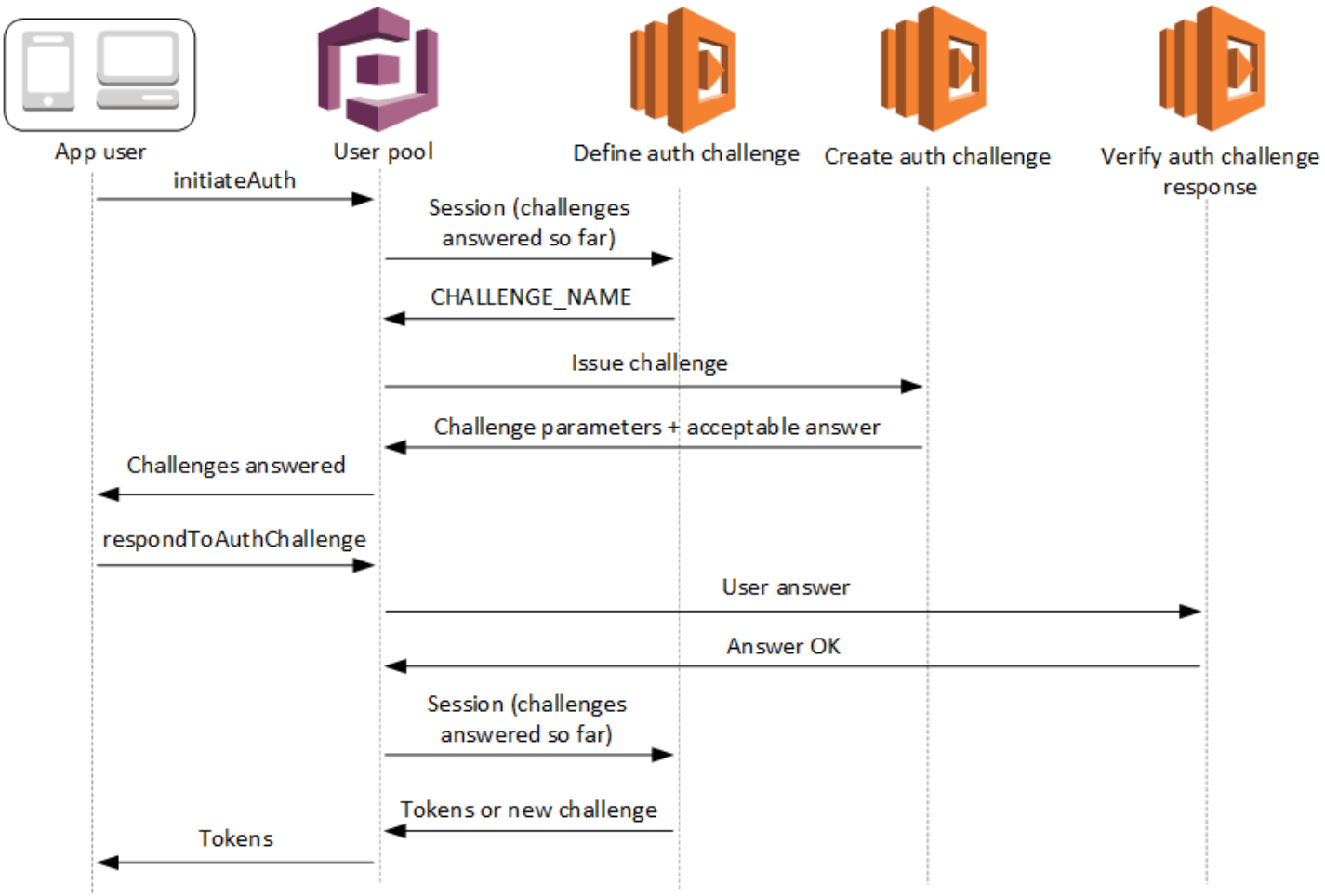
```
def lambda_handler(event, context):

    # 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": {}
}
```

Acionadores do Lambda de desafio personalizado de autenticação



Esses acionadores do Lambda emitem e verificam seus próprios desafios como parte do fluxo de autenticação personalizada de um grupo de usuários.

Definir o desafio de autenticação

O Amazon Cognito invoca esse acionador para iniciar o fluxo de autenticação personalizada.

Criar o desafio de autenticação

O Amazon Cognito invoca esse acionador depois de Define Auth Challenge (Definir desafio de autenticação) para criar um desafio personalizado.

Verificar a resposta do desafio de autenticação

O Amazon Cognito invoca esse acionador para verificar se a resposta do usuário final para um desafio personalizado é válida ou não.

Você pode incorporar novos tipos de desafio com esses tipos de acionadores do Lambda. Por exemplo, esses tipos de desafio podem incluir CAPTCHAs ou perguntas de desafio dinâmicas.

Você pode generalizar a autenticação em duas etapas comuns com os métodos de API InitiateAuth e RespondToAuthChallenge do grupo de usuários.

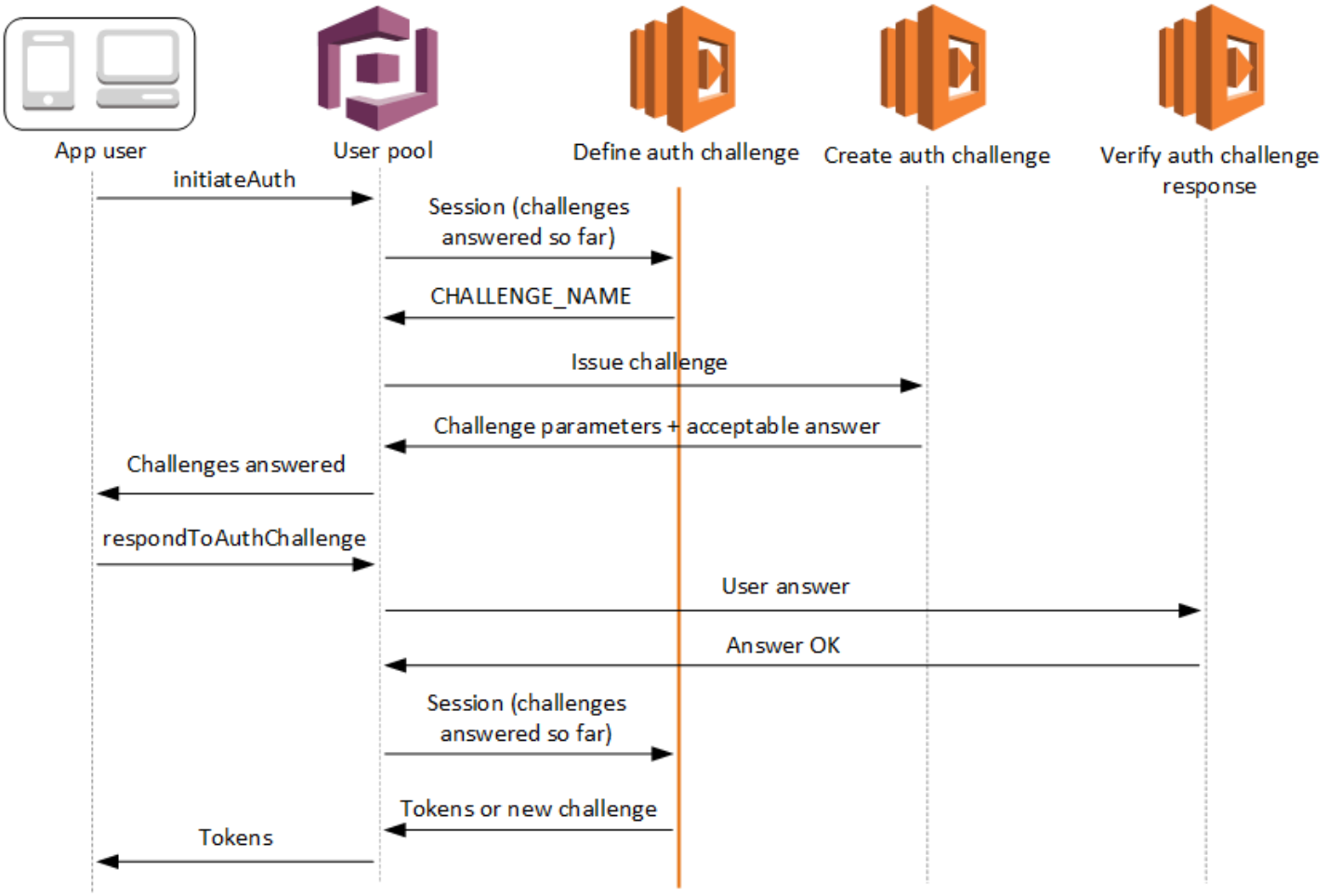
Nesse fluxo, um usuário faz a autenticação respondendo desafios sucessivos até que ela falhe ou ele receba os tokens. Essas duas chamadas de API podem ser repetidas para incluir desafios diferentes.

> **Note**
>
> A interface de usuário hospedada do Amazon Cognito não é compatível com a autenticação personalizada com gatilhos do Lambda de desafio de autenticação personalizada.

Tópicos

- 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

Acionador do Lambda para definir desafio de autenticação



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.

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.

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.

session

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, 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`. Em sua função, inclua o tratamento de eventos de entrada de desafios `SMS_MFA`. Você não precisa invocar o desafio `SMS_MFA` usando sua função de desafio de autenticação definida.

> **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 transmitir esses dados para sua função do Lambda, você pode usar o parâmetro `ClientMetadata` nas operações de API AdminRespondToAuthChallenge e RespondToAuthChallenge. A solicitação que invoca a função de definição de desafio de autenticação não inclui dados transmitidos no parâmetro ClientMetadata nas operações AdminInitiateAuth e InitiateAuth da 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

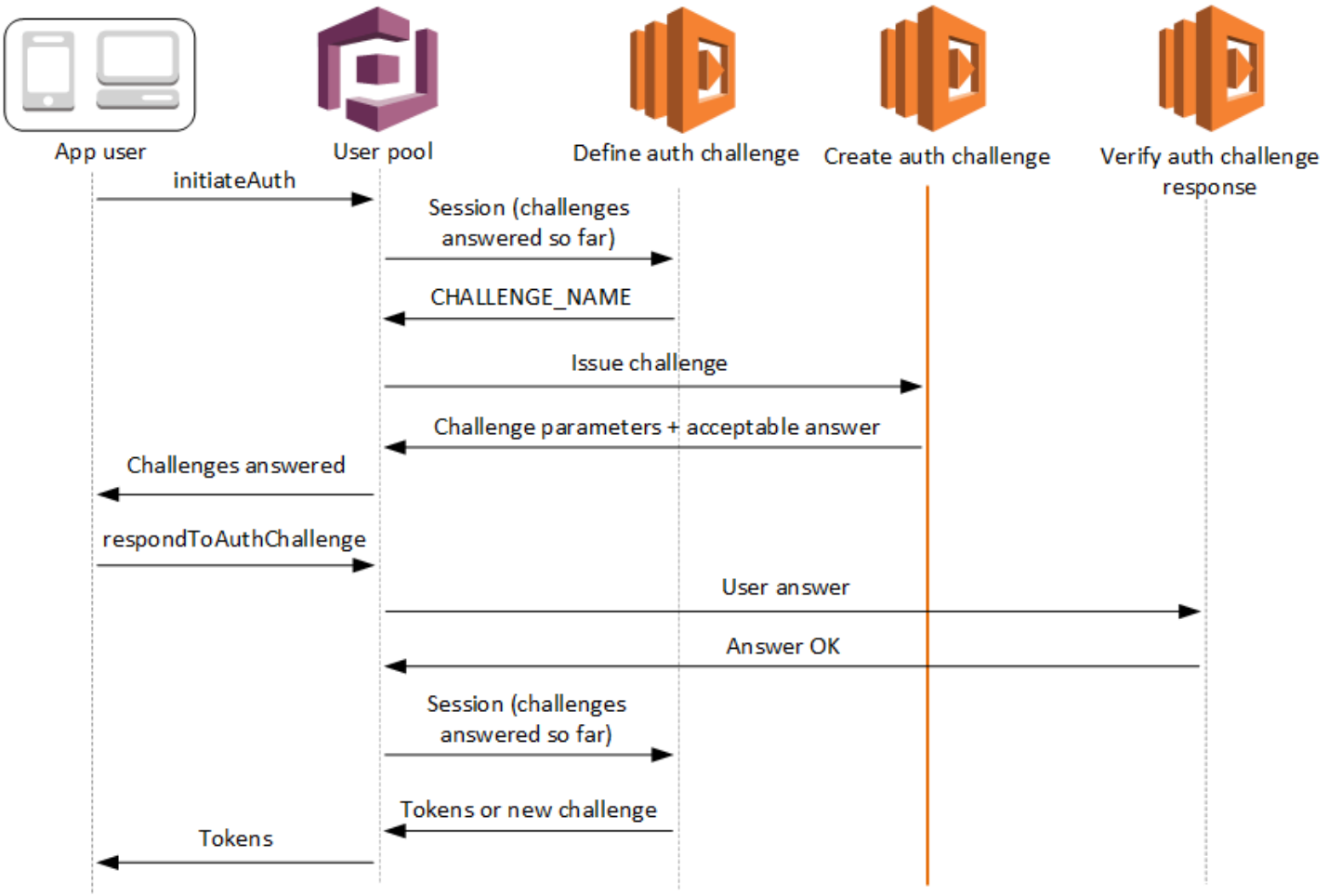
Este exemplo definirá uma série de desafios de autenticação e emitirá tokens somente se o usuário concluir todos os desafios com êxito.

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



Criar o 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.

session

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"` 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. É possível usar o parâmetro ClientMetadata nas ações de API AdminRespondToAuthChallenge e RespondToAuthChallenge para transmitir esses dados à sua função do Lambda. A solicitação que invoca a função de criação de desafio de autenticação não inclui dados transmitidos no parâmetro ClientMetadata nas operações AdminInitiateAuth e InitiateAuth 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.

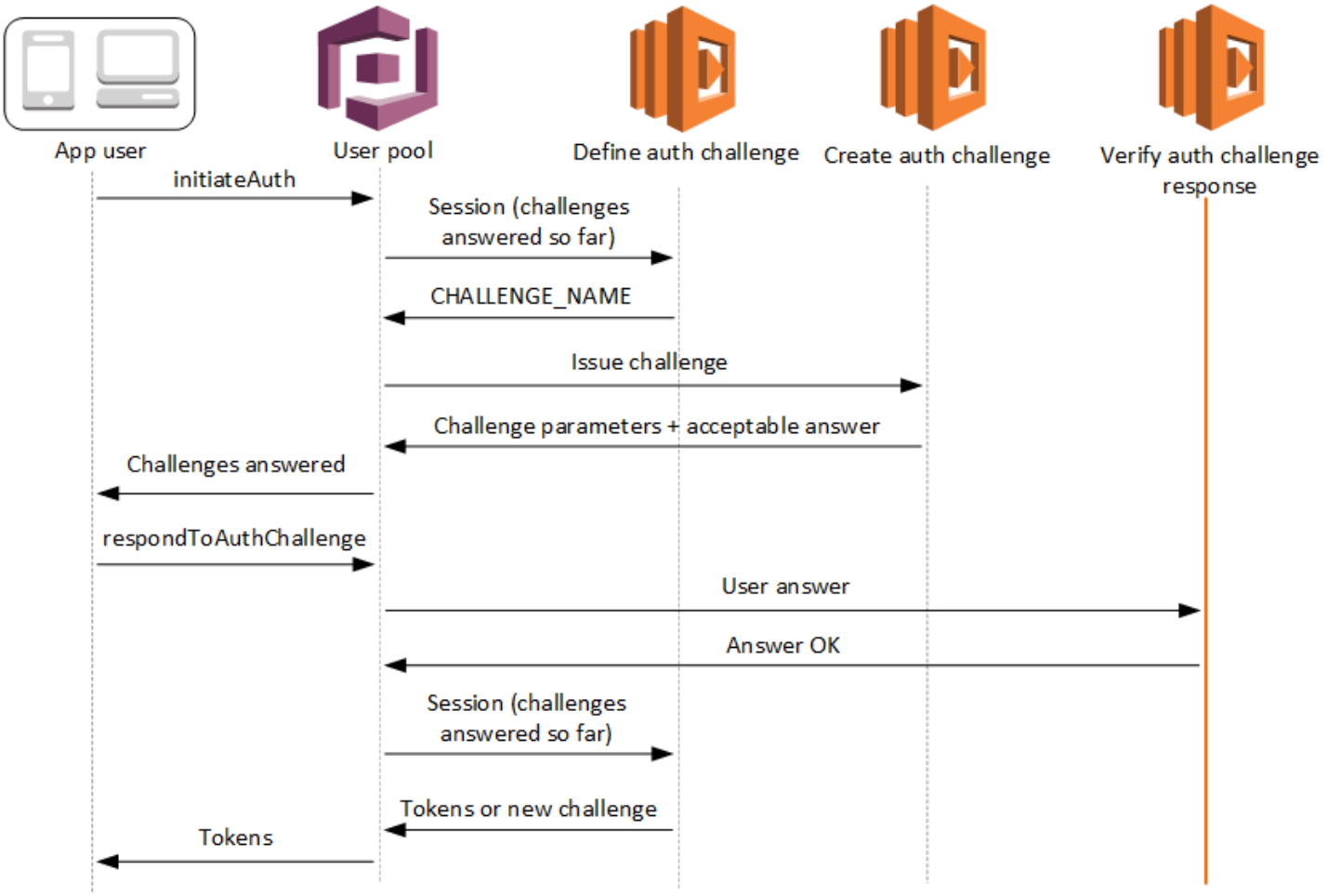
Exemplo de criar desafio de autenticação

Um CAPTCHA é criado como desafio para o 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.

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



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
- Exemplo de resposta de verificar desafio de autenticação

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 transmitir esses dados para sua função do Lambda, use o parâmetro ClientMetadata nas operações de API AdminRespondToAuthChallenge e RespondToAuthChallenge. O Amazon Cognito não inclui dados do parâmetro ClientMetadata nas operações AdminInitiateAuth e InitiateAuth da API na solicitação que ele transmite 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

Nesse exemplo, a função Lambda verifica se a resposta do usuário a um desafio corresponde à resposta esperada. 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 recursos avançados de segurança ativos, é possível gerar a versão 2 ou `V2_0` do evento de acionamento com a personalização do token de acesso.

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` 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 mais recente do gatilho e enviar dados aos dois tokens na mesma resposta.

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 gatilho pré-geração de token para sua função Lambda. Um `V1_0` gatilho fornece os parâmetros para modificação dos tokens de ID. Um `V2_0` acionador fornece parâmetros para o seguinte.

1. As funções de um `V1_0` gatilho.
2. A capacidade de personalizar os tokens de acesso.
3. A capacidade de transmitir tipos de dados complexos para valores de declaração de ID e token de acesso:
   - String
   - Número
   - Booleano
   - Matriz de cadeias de caracteres, números, booleanos ou uma combinação de qualquer um desses
   - JSON

> **Note**
>
> No token de ID, você pode preencher objetos complexos com os valores das declarações, exceto para `phone_number_verifiedemail_verified`,`updated_at`, e. `address`

Os grupos de usuários entregam `V1_0` eventos por padrão. Para configurar seu grupo de usuários para enviar um `V2_0` evento, escolha uma versão do evento Trigger dos recursos básicos + personalização do token de acesso ao configurar seu gatilho no console do Amazon Cognito. Você também pode definir o valor de `LambdaVersion` nos LambdaConfigparâmetros em uma solicitação de CreateUserPool API UpdateUserPool ou de uma solicitação. Custos adicionais se aplicam à personalização do token de acesso com `V2_0` eventos. Para mais informações, consulte Preços do Amazon Cognito.

Declarações e escopos excluídos

O Amazon Cognito limita as declarações e os escopos que você pode adicionar, modificar ou suprimir em tokens de acesso e identidade. Se a função do Lambda tentar definir um valor para qualquer uma dessas declarações, o Amazon Cognito emitirá um token com o valor da declaração original, se houver um na declaração.

Declarações compartilhadas

- `acr`
- `amr`
- `at_hash`
- `auth_time`
- `azp`
- `exp`
- `iat`
- `iss`
- `jti`
- `nbf`
- `nonce`
- `origin_jti`
- `sub`
- `token_use`

Declarações de token de ID

- `identities`

- `aud`
- `cognito:username`

Declarações de token de acesso

- `username`
- `client_id`
- `scope`

> **Note**
>
> É possível alterar os escopos em um token de acesso com os valores de resposta `scopesToAdd` e `scopesToSuppress`, mas não modificar a declaração `scope` diretamente. Não é possível adicionar escopos que comecem com `aws.cognito`, incluindo o escopo reservado `aws.cognito.signin.user.admin` dos grupos de usuários.

- `device_key`
- `event_id`
- `version`

Não é possível adicionar nem substituir declarações com os prefixos a seguir, mas é possível suprimi-las ou impedir que elas apareçam no token.

- `dev:`
- `cognito:`

É possível adicionar uma declaração `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 o gatilho 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 usar o token de 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 o gatilho 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 pelo token do Amazon Cognito e APIs de terceiros. Embora você possa gerar tokens de acesso para autorização machine-to-machine (M2M) com o Amazon Cognito com a concessão de credenciais do cliente, as solicitações M2M não invocam a função de gatilho de pré-geração de token e não podem emitir tokens de acesso personalizados. Para obter mais informações sobre tokens de acesso, consulte Como usar 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 do OAuth 2.0 na declaração `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.

AWS Management Console

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. Se ainda não o fez, ative os recursos avançados de segurança na guia Integração de aplicações.
4. Escolha a guia User pool properties (Propriedades do grupo de usuários) e localize Lambda triggers (Acionadores do Lambda).
5. Adicione ou edite um Acionador de geração de pré-token.
6. Selecione uma função do Lambda em Atribuir função do Lambda.
7. Escolha uma Versão do evento do acionador de Recursos básicos + personalização do token de acesso. 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 ter mais informações, consulte Atualizar a configuração do grupo de usuários.

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 personalização do token de acesso. 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": "V2_0"
},
```

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 da segunda versão do evento de pré-geração de tokens: adicionar declarações 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	Evento
`TokenGeneration_HostedAuth`	Chamado durante a autenticação na página de login da IU hospedada 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_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 1

O token da versão 1 pode definir a associação ao grupo, funções do IAM e novas reivindicações em tokens de ID.

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

Version 2

O evento de solicitação da versão 2 adiciona campos que personalizam o token de acesso. Ele também adiciona suporte para tipos de `claimsToOverride` dados complexos no objeto de resposta. Sua função Lambda pode retornar os seguintes tipos de dados no valor de: `claimsToOverride`

- String
- Número
- Booleano
- Matriz de cadeias de caracteres, 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

Nome	Descrição	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

Nome	Descrição	Versão mínima do evento de gatilho
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
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	Os escopos do OAuth 2.0 do usuário. 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

Nome	Descrição	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`.	2
idTokenGeneration	As declarações que você deseja substituir, adicionar ou suprimir no token de ID do usuário. Esses valores de personalização do token pai para ID aparecem somente nos eventos da versão 2, mas os elementos filhos aparecem nos eventos da versão 1.	2
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 aparecem somente nos eventos da versão 2.	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` . Nos eventos da versão 2, 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.	1

Nome	Descrição	Versão mínima do evento de gatilho
	Nos eventos da versão 2, esse elemento aparece em `accessTokenGeneration` e `idTokenGeneration`.	
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.	1
scopesToAdd	Uma lista de escopos do OAuth 2.0 que você deseja adicionar à declaração `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 do OAuth 2.0 que você deseja remover da declaração `scope` no token de acesso do usuário.	2

* Objetos de resposta aos eventos da versão 1 podem retornar sequências de caracteres. Objetos de resposta aos eventos da versão 2 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 ter 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 da segunda versão do evento de pré-geração de tokens: adicionar declarações com objetos complexos

Este exemplo faz as seguintes modificações nos tokens de um usuário.

1. Adiciona declarações dos tipos número, string, booleano e JSON ao token de ID. Essa é a única alteração que os eventos de gatilho da versão dois disponibilizam para o token de ID.
2. Adiciona declarações dos tipos número, string, booleano e JSON ao token de acesso.
3. Adiciona três escopos ao token de acesso.
4. Suprime as `sub` reivindicações `email` e nos tokens de ID e acesso.
5. Suprime o `aws.cognito.signin.user.admin` escopo 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","sub"]
            },
            "accessTokenGeneration": {
                "claimsToAddOrOverride": claims,
                "claimsToSuppress": ["email","sub"],
                "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 para grupos de usuários com um acionador Lambda de migração de usuário.

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 função do LambdaSource	Evento
`UserMigration_Authentication`	Migração de usuários no login.
`UserMigration_ForgotPassword`	Migração de usuários durante o fluxo de esquecimento de 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. É possível passar esses dados para a função do Lambda usando o parâmetro ClientMetadata nas ações de API InitiateAuth e AdminInitiateAuth.

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 a função do Lambda, você pode usar o parâmetro ClientMetadata nas ações de API AdminRespondToAuthChallenge e ForgotPassword.

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. Depois, se a senha não atender à política, defina finalUserStatus 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 o parâmetro estiver definido como `TRUE` e o número de telefone ou o endereço de e-mail no parâmetro UserAttributes já existir como alias com um usuário diferente, a chamada de 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

```
const validUsers = {
  belladonna: { password: "Test123", emailAddress: "bella@example.com" },
};

// Replace this mock with a call to a real authentication service.
const authenticateUser = (username, password) => {
  if (validUsers[username] && validUsers[username].password === password) {
    return validUsers[username];
  } else {
    return null;
  }
```

```
};

const lookupUser = (username) => {
  const user = validUsers[username];

  if (user) {
    return { emailAddress: user.emailAddress };
  } else {
    return null;
  }
};

const handler = async (event) => {
  if (event.triggerSource == "UserMigration_Authentication") {
    // Authenticate the user with your existing user directory service
    const 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";
    }
  } else if (event.triggerSource == "UserMigration_ForgotPassword") {
    // Look up the user in your existing user directory service
    const 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";
    }
  }

  return event;
};

export { handler };
```

Acionador do Lambda de mensagem personalizada

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. É possível editar mensagens personalizadas estáticas na guia Message Customizations (Personalizações de mensagem) do console do Amazon Cognito.

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**
>
> 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, `request.usernameParameter` e `request.codeParameter`.

Tópicos

Fontes do acionador do Lambda de mensagem personalizada

Valor de triggerSource	Evento
`CustomMessage_SignUp`	Custom message – Para enviar o código de confirmação após cadastro.

Valor de triggerSource	Evento
`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.
`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,
  "triggerSource": "CustomMessage_SignUp/CustomMessage_ResendCode/
CustomMessage_ForgotPassword/CustomMessage_VerifyUserAttribute",
  "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": "####"
  },
  "response": {
      "smsMessage": "<custom message to be sent in the message with code parameter>"
      "emailMessage": "<custom message to be sent in the message with code
 parameter>"
      "emailSubject": "<custom email subject>"
  }
}
```

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 Lambda tem um `triggerSource` valor e um nome de usuário `CustomMessage_AdminCreateUser` e uma senha temporária. A função é preenchida a `${event.request.codeParameter}` partir da senha temporária na solicitação e `${event.request.usernameParameter}` 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` `event.response.emailMessage` e.

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 grupos de usuários do Amazon Cognito fornecem os acionadores do Lambda `CustomEmailSender` e `CustomSMSSender` para ativar notificações por e-mail e SMS de terceiros. 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 deve enviar notificações como códigos de confirmação, códigos de verificação ou 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 cliente AWS KMS e o AWS Encryption SDK. O AWS Encryption SDK é uma biblioteca de criptografia do lado do cliente que ajuda você a criptografar e descriptografar dados genéricos.

> **Note**
>
> 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.

CustomEmailSender

O Amazon Cognito invoca esse acionador para enviar notificações por e-mail aos usuários.

CustomSMSSender

O Amazon Cognito invoca esse acionador para enviar notificações SMS aos usuários.

Recursos

Os recursos a seguir podem ajudar você a usar os acionadores `CustomEmailSender` e `CustomSMSSender`.

AWS KMS

AWS KMS é um serviço gerenciado para criar e controlar chaves AWS KMS. Essas chaves criptografam seus dados. Para obter mais informações, consulte O que é o AWS Key Management Service?.

Chave do 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 do KMS simétrica, você deve chamar o AWS KMS. O Amazon Cognito usa chaves simétricas. A mesma chave criptografa e descriptografa. Para obter mais informações, consulte Chaves simétricas do KMS.

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 acionador de remetente personalizado, a função do AWS Lambda pode enviar notificações por e-mail aos 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.

> **Note**
>
> No momento, não é possível atribuir acionadores de remetente personalizados no console do Amazon Cognito. Você pode atribuir um acionador com o parâmetro `LambdaConfig` em uma solicitação de API `CreateUserPool` ou `UpdateUserPool`.

Para configurar esse acionador, execute as seguintes etapas:

1. Crie uma chave de criptografia simétrica no AWS Key Management Service (AWS KMS). O Amazon Cognito gera segredos, que são senhas temporárias, códigos de verificação e códigos de autorização, e usa essa chave do KMS para criptografá-los. Depois, você pode usar a operação Descriptografar da API na função do Lambda para descriptografar os segredos e enviá-los ao usuário em texto simples. O AWS Encryption SDK é uma ferramenta útil para operações do AWS KMS em sua função.
2. Crie uma função do Lambda que você deseja atribuir como acionador de remetente personalizado. Conceda permissões `kms:Decrypt` para a chave do KMS ao perfil da função do Lambda.
3. Conceda à entidade principal `cognito-idp.amazonaws.com` do serviço do Amazon Cognito acesso para invocar a função do Lambda.
4. Escreva um código da função do Lambda que direcione suas mensagens para métodos de entrega personalizados ou provedores de terceiros. Para entregar o código de verificação ou confirmação do usuário, o Base64 decodifica e descriptografa o valor do parâmetro `code` na solicitação. Essa operação produz um código ou senha em texto simples que você deve incluir na mensagem.
5. Atualize o grupo de usuários para que ele use um acionador do Lambda de remetente personalizado. A entidade principal do IAM que atualiza ou cria um grupo de usuários com um acionador de remetente personalizado deve ter permissão para criar uma concessão para a chave do KMS. O trecho `LambdaConfig` a seguir atribui funções personalizadas de remetente de SMS e e-mail.

```
"LambdaConfig": {
   "KMSKeyID": "arn:aws:kms:us-
east-1:123456789012:key/a6c4f8e2-0c45-47db-925f-87854bc9e357",
   "CustomEmailSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
      "LambdaVersion": "V1_0"
```

```
  },
  "CustomSMSSender": {
     "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
     "LambdaVersion": "V1_0"
  }
```

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 transmitir esses dados para sua função do Lambda, é possível usar o parâmetro ClientMetadata nas ações de API AdminRespondToAuthChallenge e RespondToAuthChallenge. O Amazon Cognito não inclui dados do parâmetro ClientMetadata nas operações AdminInitiateAuth e InitiateAuth da API na solicitação que ele transmite 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 e-mail

O Amazon Cognito não espera nenhuma outra informação de retorno na resposta do remetente personalizado de e-mail. Sua função pode usar operações de API para consultar e modificar seus recursos ou registrar metadados de eventos em um sistema externo.

Ativar o acionador do Lambda de remetente personalizado de e-mail

Para configurar um acionador de remetente personalizado de e-mail que usa lógica personalizada para enviar mensagens de e-mail ao grupo de usuários, ative-o da maneira a seguir. O procedimento a seguir atribui um gatilho de e-mail personalizado, um gatilho de SMS personalizado ou ambos ao grupo de usuários. Depois de adicionar o gatilho de remetente de e-mail personalizado, o Amazon Cognito sempre envia atributos do usuário, incluindo o endereço de e-mail e o código único para a função do Lambda, quando, de outra forma, enviaria uma mensagem de e-mail com o Amazon Simple Email Service.

> **Important**
>
> 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.

1. Crie uma chave de criptografia no AWS KMS. Essa chave criptografa senhas temporárias e códigos de autorização gerados pelo Amazon Cognito. Depois, é possível descriptografar esses segredos na função do Lambda de remetente personalizado para enviá-los ao usuário em texto não criptografado.
2. Conceda acesso à entidade principal `cognito-idp.amazonaws.com` do serviço Amazon Cognito para criptografar códigos com a chave do KMS.

   Aplique a seguinte política baseada em recursos à chave do KMS.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": {
            "Service": "cognito-idp.amazonaws.com"
        },
        "Action": "kms:CreateGrant",
        "Resource": "arn:aws:kms:us-
west-2:111222333444:key/1example-2222-3333-4444-999example",
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "111222333444"
            },
            "ArnLike": {
                "aws:SourceArn": "arn:aws:cognito-idp:us-
west-2:111222333444:userpool/us-east-1_EXAMPLE"
            }
        }
    }]
}
```

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.

   - Atribua um perfil do IAM à função do Lambda que tenha, no mínimo, as permissões `kms:Decrypt` para a chave do KMS.
4. Conceda à entidade principal `cognito-idp.amazonaws.com` do serviço do Amazon Cognito acesso para invocar a função do Lambda.

O comando AWS CLI a seguir concede permissão ao Amazon Cognito para invocar a função do Lambda:

```
        aws lambda add-permission --function-name lambda_arn --statement-id
 "CognitoLambdaInvokeAccess" --action lambda:InvokeFunction --principal cognito-
idp.amazonaws.com
```

5. Componha o código da função do Lambda para enviar as mensagens. O Amazon Cognito usa o AWS Encryption SDK para criptografar segredos antes de enviá-los à função do Lambda de remetente personalizado. Em sua função, decifre o segredo 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.
6. Adicione o AWS Encryption SDK à função do Lambda. Para ter mais informações, consulte Linguagens de programação do AWS Encryption SDK. Para atualizar o pacote do Lambda, conclua as etapas a seguir.

   a. Exporte a função do Lambda como um arquivo .zip no AWS Management Console.

   b. Abra a função e adicione o AWS Encryption SDK. Para ter mais informações e links de download, consulte Linguagens de programação do AWS Encryption SDK no Guia do desenvolvedor do AWS Encryption SDK.

   c. 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.
7. Atualize o grupo de usuários para adicionar acionadores do Lambda de remetente personalizado. Inclua um parâmetro `CustomSMSSender` ou `CustomEmailSender` em uma solicitação de API do `UpdateUserPool`. A operação de API `UpdateUserPool` 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 a seguir, inclua entradas para todas as funções do Lambda que você deseja adicionar ou manter no grupo de usuários. Para obter mais informações, consulte Atualizar a configuração do grupo de usuários.

```
   #Send this parameter in an 'aws cognito-idp update-user-pool' CLI command,
including any existing
```

```
    #user pool configurations.

      --lambda-config "PreSignUp=lambda-arn, \
                       CustomSMSSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                       CustomEmailSender={LambdaVersion=V1_0,LambdaArn=lambda-arn},
\
                       KMSKeyID=key-id"
```

Para remover um acionador do Lambda de remetente personalizado com um AWS CLI da `update-user-pool`, omita o parâmetro `CustomSMSSender` ou `CustomEmailSender` e inclua todos os outros acionadores que você deseja usar com o 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.

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_ALIAS`

O alias 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.

```
const AWS = require('aws-sdk');
const b64 = require('base64-js');
const encryptionSdk = require('@aws-crypto/client-node');
//Configure the encryption SDK client with the KMS key from the environment variables.
const { encrypt, decrypt } =
 encryptionSdk.buildClient(encryptionSdk.CommitmentPolicy.REQUIRE_ENCRYPT_ALLOW_DECRYPT);
const generatorKeyId = process.env.KEY_ALIAS;
```

```
const keyIds = [ process.env.KEY_ARN ];
const keyring = new encryptionSdk.KmsKeyringNode({ generatorKeyId, keyIds })
exports.handler = async (event) => {
 //Decrypt the secret code using encryption SDK.
 let plainTextCode;
 if(event.request.code){
  const { plaintext, messageHeader } = await decrypt(keyring,
 b64.toByteArray(event.request.code));
  plainTextCode = plaintext
 }
 //PlainTextCode now contains the decrypted secret.
 if(event.triggerSource == 'CustomEmailSender_SignUp'){
  //Send an email message to your user via a custom provider.
  //Include the temporary password in the message.
 }
 else if(event.triggerSource == 'CustomEmailSender_ResendCode'){
 }
 else if(event.triggerSource == 'CustomEmailSender_ForgotPassword'){
 }
 else if(event.triggerSource == 'CustomEmailSender_UpdateUserAttribute'){
 }
 else if(event.triggerSource == 'CustomEmailSender_VerifyUserAttribute'){
 }
 else if(event.triggerSource == 'CustomEmailSender_AdminCreateUser'){
 }
 else if(event.triggerSource == 'CustomEmailSender_AccountTakeOverNotification'){
 }
 return;
};
```

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`	Evento
`CustomEmailSender_SignUp`	Um usuário se cadastra e o Amazon Cognito envia uma mensagem de boas-vindas.
`CustomEmailSender_ForgotPassword`	Um usuário solicita um código para redefinir a senha.

`TriggerSource value`	Evento
`CustomEmailSender_ResendCode`	Um usuário solicita um código de substituição para redefinir a senha.
`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.

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.

> **Note**
>
> No momento, não é possível atribuir acionadores de remetente personalizados no console do Amazon Cognito. Você pode atribuir um acionador com o parâmetro `LambdaConfig` em uma solicitação de API `CreateUserPool` ou `UpdateUserPool`.

Para configurar esse acionador, execute as seguintes etapas:

1. Crie uma chave de criptografia simétrica em AWS Key Management Service (AWS KMS). O Amazon Cognito gera segredos, que são senhas temporárias, códigos de verificação e códigos de autorização, e usa essa chave do KMS para criptografá-los. Depois, você pode usar a operação Descriptografar da API na função do Lambda para descriptografar os segredos e enviá-los ao usuário em texto simples. AWS Encryption SDKÉ uma ferramenta útil para AWS KMS operações em sua função.
2. Crie uma função do Lambda que você deseja atribuir como acionador de remetente personalizado. Conceda permissões `kms:Decrypt` para a chave do KMS ao perfil da função do Lambda.
3. Conceda à entidade principal `cognito-idp.amazonaws.com` do serviço do Amazon Cognito acesso para invocar a função do Lambda.
4. Escreva um código da função do Lambda que direcione suas mensagens para métodos de entrega personalizados ou provedores de terceiros. Para entregar o código de verificação ou confirmação do usuário, o Base64 decodifica e descriptografa o valor do parâmetro `code` na solicitação. Essa operação produz um código ou senha em texto simples que você deve incluir na mensagem.
5. Atualize o grupo de usuários para que ele use um acionador do Lambda de remetente personalizado. A entidade principal do IAM que atualiza ou cria um grupo de usuários com um acionador de remetente personalizado deve ter permissão para criar uma concessão para a chave do KMS. O trecho `LambdaConfig` a seguir atribui funções personalizadas de remetente de SMS e e-mail.

```
"LambdaConfig": {
   "KMSKeyID": "arn:aws:kms:us-
east-1:123456789012:key/a6c4f8e2-0c45-47db-925f-87854bc9e357",
   "CustomEmailSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
      "LambdaVersion": "V1_0"
   },
   "CustomSMSSender": {
      "LambdaArn": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",
      "LambdaVersion": "V1_0"
   }
```

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.

Ativar o acionador do Lambda de remetente personalizado de SMS

É possível configurar um acionador de remetente personalizado de e-mail que usa lógica personalizada para enviar mensagens de SMS ao grupo de usuários. 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 com o Amazon Simple Notification Service.

> **Important**
>
> 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.

1. Crie uma chave de criptografia no AWS KMS. Essa chave criptografa senhas temporárias e códigos de autorização gerados pelo Amazon Cognito. Depois, é possível descriptografar esses segredos na função do Lambda de remetente personalizado para enviá-los ao usuário em texto não criptografado.
2. Conceda acesso à entidade principal `cognito-idp.amazonaws.com` do serviço Amazon Cognito para criptografar códigos com a chave do KMS.

   Aplique a seguinte política baseada em recursos à chave do KMS.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": {
            "Service": "cognito-idp.amazonaws.com"
        },
        "Action": "kms:CreateGrant",
        "Resource": "arn:aws:kms:us-
west-2:111222333444:key/1example-2222-3333-4444-999example",
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "111222333444"
            },
            "ArnLike": {
                "aws:SourceArn": "arn:aws:cognito-idp:us-
west-2:111222333444:userpool/us-east-1_EXAMPLE"
            }
        }
    }]
}
```

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.

   - Atribua um perfil do IAM à função do Lambda que tenha, no mínimo, as permissões `kms:Decrypt` para a chave do KMS.

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. Componha o código da função do Lambda para enviar as mensagens. O Amazon Cognito usa para AWS Encryption SDK criptografar segredos antes que o Amazon Cognito envie os segredos para a função Lambda personalizada do remetente. Em sua função, decifre o segredo 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.
6. 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.

   a. Exporte a função do Lambda como um arquivo .zip no AWS Management Console.

   b. 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 .

   c. 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 .
7. Atualize o grupo de usuários para adicionar acionadores do Lambda de remetente personalizado. Inclua um parâmetro `CustomSMSSender` ou `CustomEmailSender` em uma solicitação de API do `UpdateUserPool`. A operação de API `UpdateUserPool` 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 a seguir, inclua entradas para todas as funções do Lambda que você deseja adicionar ou manter no grupo de usuários. Para ter mais informações, consulte Atualizar a configuração do grupo de usuários.

```
   #Send this parameter in an 'aws cognito-idp update-user-pool' CLI command,
including any existing
   #user pool configurations.

     --lambda-config "PreSignUp=lambda-arn, \
                      CustomSMSSender={LambdaVersion=V1_0,LambdaArn=lambda-arn}, \
                      CustomEmailSender={LambdaVersion=V1_0,LambdaArn=lambda-arn},
\
                      KMSKeyID=key-id"
```

Para remover um gatilho Lambda personalizado do remetente com `update-user-pool` AWS CLI um, 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.

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_ALIAS`

O alias 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.

```
const AWS = require('aws-sdk');
const b64 = require('base64-js');
const encryptionSdk = require('@aws-crypto/client-node');
//Configure the encryption SDK client with the KMS key from the environment variables.

const { encrypt, decrypt } =
 encryptionSdk.buildClient(encryptionSdk.CommitmentPolicy.REQUIRE_ENCRYPT_ALLOW_DECRYPT);
const generatorKeyId = process.env.KEY_ALIAS;
const keyIds = [ process.env.KEY_ARN ];
const keyring = new encryptionSdk.KmsKeyringNode({ generatorKeyId, keyIds })
exports.handler = async (event) => {
 //Decrypt the secret code using encryption SDK.
 let plainTextCode;
 if(event.request.code){
  const { plaintext, messageHeader } = await decrypt(keyring,
 b64.toByteArray(event.request.code));
  plainTextCode = plaintext
```

```
 }
 //PlainTextCode now contains the decrypted secret.
 if(event.triggerSource == 'CustomSMSSender_SignUp'){
  //Send an SMS message to your user via a custom provider.
  //Include the temporary password in the message.
 }
 else if(event.triggerSource == 'CustomSMSSender_ResendCode'){
 }
 else if(event.triggerSource == 'CustomSMSSender_ForgotPassword'){
 }
 else if(event.triggerSource == 'CustomSMSSender_UpdateUserAttribute'){
 }
 else if(event.triggerSource == 'CustomSMSSender_VerifyUserAttribute'){
 }
 else if(event.triggerSource == 'CustomSMSSender_AdminCreateUser'){
 }
 else if(event.triggerSource == 'CustomSMSSender_AccountTakeOverNotification'){
 }
 return;
};
```

Tópicos

- Avaliar os recursos de mensagem SMS com uma função de remetente personalizado de SMS
- Fontes de acionador do Lambda remetente personalizado de SMS

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.

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`	Evento
`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_VerifyUserA ttribute`	Um usuário cria um endereço de e-mail ou um atributo de número de telefone e o Amazon

TriggerSource value	Evento
	Cognito envia um código para verificar o atributo.
`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 configurado com autenticação multifator (MFA) de SMS faz login.
`CustomSMSSender_AdminCreateUser`	Você cria um usuário em seu grupo de usuários e o Amazon Cognito envia uma senha temporária.

Como usar análise do Amazon Pinpoint com grupos de usuários do Amazon Cognito

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 à análise do Amazon Pinpoint em grupos de usuários do Amazon Cognito, você pode rastrear inscrições, logins, falhas de autenticação do grupo de usuários, DAUs (usuários ativos diários) e MAUs (usuários ativos mensais) no console do Amazon Pinpoint. 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 Share user attribute data with Amazon Pinpoint (Compartilhar dados de atributos do usuário com o Amazon Pinpoint) na guia Analytics (Análise) no console do Amazon Cognito, o Amazon Pinpoint cria 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. A entidade principal do IAM que adiciona sua configuração de análise deve ter permissões CreateServiceLinkedRole. O perfil vinculado ao serviço é denominado 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 mapeamentos de Região da AWS entre o Amazon Cognito e o Amazon Pinpoint que atendem a uma das condições a seguir.

- 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)

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
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 todas as Regiões da AWS exceto aquelas na tabela anterior, o Amazon Cognito só pode usar um projeto do Amazon Pinpoint na mesma região do 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 (AWS Management Console)

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 da AWS.
2. Selecione User Pools (Grupos de usuários) e escolha um grupo de usuários existente na lista.
3. Escolha a guia App integration (Integração da aplicação).
4. Em App clients and analytics (Clientes e análise da aplicação), escolha um App client name (Nome do cliente da aplicação) existente na lista.
5. Em Pinpoint analytics (Análise do Pinpoint), selecione Enable (Habilitar).
6. Escolha uma Pinpoint Region (Região do Pinpoint).
7. 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.

8. 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.

9. Save changes (Salvar alterações).

Como especificar configurações de análise do Amazon Pinpoint (AWS CLI e API da AWS)

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`
- API da AWS: 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`
- API da AWS: 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 uma aplicação com uma interface de usuário (UI) personalizada. Você não pode transmitir metadados de análise para usuários federados ou nativos que fazem login por meio da interface do usuário hospedada. 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. Para ver um exemplo de JavaScript, consulte "Por que minhas análises do grupo de usuários do Amazon Cognito não estão aparecendo no meu painel do Amazon Pinpoint?" no Centro de Conhecimentos da AWS.

Como configurar a análise do grupo de usuários

Usando a análise do Amazon Pinpoint, você pode rastrear cadastros, logins, falhas de autenticação do grupo de usuários, usuários ativos diariamente (DAUs) e usuários ativos mensalmente (MAUs) do Amazon Cognito. Você também pode configurar atributos de usuário específicos para sua aplicação usando o AWS Mobile SDK for Android ou o AWS Mobile SDK for iOS. Eles poderão ser usados para segmentar seus usuários no Amazon Pinpoint e enviar a eles notificações por push direcionadas.

Na guia Integração de aplicações em Clientes e análises de aplicativos, você pode navegar até um cliente de aplicação existente ou criar um. Na configuração do seu cliente de aplicativo, você pode especificar um projeto do Amazon Pinpoint que deseja usar com sua aplicação. Para obter mais informações, consulte Usar análise do Amazon Pinpoint com grupos de usuários do Amazon Cognito.

> **Note**
>
> O Amazon Pinpoint está disponível em várias regiões da AWS na 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 compatível com o 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 obter 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.

Para adicionar análises e campanhas

1. Selecione Add analytics and campaigns.
2. Selecione um Cognito app client na lista.
3. Para mapear sua aplicação do Amazon Cognito para um Projeto do Amazon Pinpoint, escolha o projeto do Amazon Pinpoint na lista.

> **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.

4. Escolha Share user attribute data with Amazon Pinpoint (Compartilhar dados do atributo do usuário com o Amazon Pinpoint) se você quiser que o Amazon Cognito envie endereços de e-mail e números de telefone para o Amazon Pinpoint para criar endpoints adicionais para os usuários.

> **Note**
>
> Um endpoint identifica exclusivamente um dispositivo de usuário ao qual você pode enviar notificações por push com o Amazon Pinpoint. Para obter mais informações sobre endpoints, consulte Adicionar endpoints no Guia do desenvolvedor do Amazon Pinpoint.

5. Insira uma função do IAM que você já criou ou selecione Create new role (Criar nova função) para criar uma nova função no console do IAM.
6. Save changes (Salvar alterações).
7. Para especificar mapeamentos de aplicações adicionais, escolha Add app mapping (Adicionar mapeamento de aplicações).
8. Save changes (Salvar alterações).

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.

Você pode importar usuários para um grupo de usuários empregando um acionador de migração de usuários do Lambda. Essa abordagem permite uma migração sem falhas de usuários do diretório de usuário existente para grupos de usuários quando eles fazem login no seu grupo de usuários pela primeira vez.

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 de aplicação, com ou sem uma interface de usuário hospedada, cria um perfil de usuário para qualquer pessoa na Internet que conheça seu ID de cliente de aplicação que pode ser descoberto publicamente e solicite se cadastrar. 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.

AWS Management Console

A guia Experiência de cadastro do grupo de usuários e a etapa Configurar experiência de cadastro do assistente de criação de grupos de usuários contêm algumas das configurações de cadastro e criação administrativa de usuários em seu 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 no assistente de criação de grupos 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. Na guia Cadastro por autoatendimento, selecione Habilitar autorregistro, se desejar que os usuários gerem uma nova conta com a API `SignUp` não autenticada. Se você desativar o autorregistro, só poderá criar usuários como administrador no console do Amazon Cognito ou com solicitações da API AdminCreateUser. Em um grupo de usuários em que o autorregistro está inativo, as solicitações da API SignUp retornam `NotAuthorizedException`, e a interface do usuário hospedada não exibe um link para Cadastrar-se.

Para grupos de usuários nos quais você planeja criar usuários como administrador, é possível configurar a duração de suas senhas temporárias na guia Experiência de login 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 na guia Mensagens, 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 de aplicações com segredos de clientes na guia Integração de aplicações do console do Amazon Cognito.

Amazon Cognito user pools API

É possível definir os parâmetros para a criação de usuários em um grupo de usuários de forma programática em uma solicitação da API CreateUserPool ou UpdateUserPool.

O elemento AdminCreateUserConfig define os 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 da API CreateUserPool ou UpdateUserPool 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

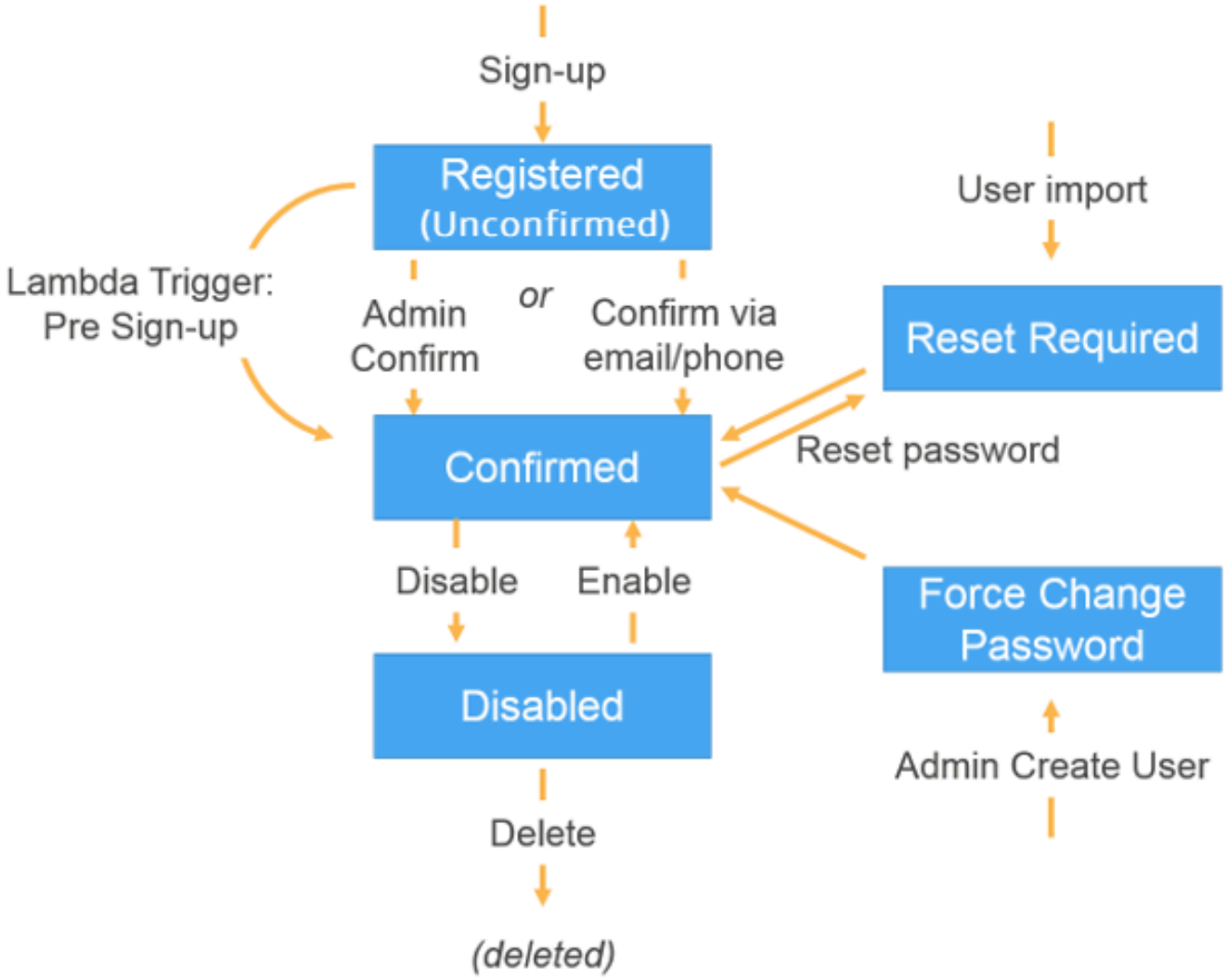
Contas de usuários são adicionadas ao grupo de usuários de uma das seguintes formas:

- 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 ter 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 ter 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.

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.

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 o. AWS CLI 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.

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 AWS Management Console 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. Escolha a guia Sign-up experience (Experiência de cadastro) e localize Attribute verification and user account confirmation (Verificação de atributo e confirmação da conta do usuário). Escolha Edit (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 ter 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 ter 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 cadastra em sua aplicação inserindo um nome de usuário, número de telefone e/ou endereço de 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 cadastra em sua aplicação inserindo um nome de usuário, número de telefone e/ou endereço de 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 AWS Management Console 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 a guia Usuários, selecione o usuário que você deseja confirmar e, no menu Ações, selecione 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`.

As seguintes APIs de grupos de usuários do Amazon Cognito aceitam um valor de hash de segredo do cliente em um parâmetro `SecretHash`.

- ConfirmForgotPassword
- ConfirmSignUp
- ForgotPassword
- ResendConfirmationCode
- SignUp

Além disso, as APIs a seguir aceitam um valor de hash do segredo do cliente em um parâmetro `SECRET_HASH`, 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 pseudocode, + indica concatenação, `HMAC_SHA256` representa uma função que produz um valor de HMAC usando HmacSHA256, e `Base64` representa uma função que produz a versão codificada em Base64 do hash de saída.

```
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

Quando um usuário atualiza o endereço de e-mail ou número de telefone na aplicação, o Amazon Cognito envia imediatamente uma mensagem com um código de confirmação ao usuário se você tiver configurado seu grupo de usuários para confirmar automaticamente esse atributo. Depois, o usuário precisa fornecer o código da mensagem de confirmação para a aplicação. Em seguida, seu aplicativo envia o código em uma solicitação de VerifyUserAttributeAPI para concluir a verificação do novo valor do atributo.

Se o grupo de usuários não exigir que os usuários confirmem um endereço de e-mail ou um número de telefone atualizado, o Amazon Cognito alterará imediatamente o valor de um atributo `email` ou `phone_number` atualizado e o marcará como não confirmado. Seu usuário não pode fazer login com um e-mail ou número de telefone não confirmado. Ele precisa concluir a confirmação do valor atualizado para poder usar esse atributo como um alias de login.

Se o grupo de usuários exigir que os usuários confirmem um endereço de e-mail ou um número de telefone atualizado, o Amazon Cognito deixará o atributo verificado e definido como seu valor original até que o usuário confirme o novo valor do atributo. Se o atributo for um alias para login, seu usuário poderá entrar com o valor do atributo original até que a confirmação altere o atributo para o novo valor. Para obter mais informações sobre como configurar seu grupo de usuários para exigir que os usuários confirmem atributos atualizados, consulte Configurar a confirmação de e-mail ou telefone.

Você pode usar um acionador de mensagem personalizada do Lambda para personalizar a mensagem de confirmação. Para ter mais informações, consulte Acionador do Lambda de mensagem personalizada. Sempre que o endereço de e-mail ou o número de telefone do usuário não estiverem verificados, sua aplicação deverá informar ao usuário que ele precisa verificar o atributo e fornecer um botão ou um link para que o usuário verifique o novo e-mail ou número de telefone.

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 AWS Management Console 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 ter 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 na guia Mensagens. 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 na guia Mensagens do console do Amazon Cognito. Para ter 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 ter 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 ter 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 mensagens e como elas funcionam? em Perguntas frequentes do Amazon SNS.

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 ter mais informações, consulte Escolha o Região da AWS para mensagens SMS do Amazon SNS.

Região do Amazon Cognito	Região alternativa herdada do Amazon SNS
Leste dos EUA (Ohio)	Leste dos EUA (N. 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. Na guia Sign-up experience (Experiência de cadastro), escolha Edit (Editar) em Attribute verification and user account confirmation (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). Na guia Mensagens do console Amazon Cognito em SMS, selecione Editar para definir uma função.

Como configurar mensagens de verificação de SMS e de e-mail, e mensagens de convite de usuário

O Amazon Cognito permite que você personalize mensagens de verificação por SMS e e-mail, bem como mensagens de convite de usuário, para aprimorar a segurança e a experiência do usuário do seu aplicativo. Com o Amazon Cognito, você pode escolher entre verificações de links com base em código ou com um clique para atender às necessidades do seu aplicativo. Este tópico discute como você pode personalizar a autenticação multifatorial (MFA) e as comunicações de verificação no console do Amazon Cognito.

Na guia Mensagens em Modelos de mensagens, você pode personalizar:

- Sua mensagem de texto SMS de autenticação multifator (MFA)
- Mensagens de verificação de SMS e e-mail
- O tipo de verificação para e-mail: código ou link
- 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 de SMS e e-mail só serão exibidos se você tiver optado por exigir a verificação de número de telefone e e-mail na guia Verifications. 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 campos em suas mensagens usando espaços reservados que serão substituídos pelos valores correspondentes.

Espaços reservados de modelo

Descrição	Token
Código de verificação	`{####}`

Descrição	Token
Senha temporária	`{####}`
Nome do usuário	`{username}`

> **Note**
>
> Não é possível usar o espaço reservado `{username}` em mensagens de e-mail de verificação. Você pode usar o `{username}` espaço reservado nas mensagens de e-mail de convite que você gera com a AdminCreateUseroperação. Essas mensagens de e-mail de convite usam dois espaços reservados: o nome de usuário, como `{username}`, e a senha temporária, como `{####}`.

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. Os recursos de segurança avançada do Amazon Cognito podem 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.

Espaços reservados de modelo de segurança avançada

Descrição	Token
Endereço IP	`{ip-address}`

Descrição	Token
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}`

Como personalizar mensagem SMS

> **Note**
>
> Na nova experiência de console do Amazon Cognito, você pode personalizar mensagens SMS.

Você pode personalizar a mensagem SMS para autenticação multifator (MFA) na guia Sistema de mensagens abaixo do título Modelos de mensagens.

> **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 impõe uma extensão máxima para mensagens SMS, incluindo o código de autenticação, de 140 caracteres UTF-8.

Como personalizar mensagens SMS de verificação

Personalize a mensagem SMS para verificações de número de telefone editando o modelo abaixo do título Do you want to customize your SMS verification messages? (Deseja personalizar suas mensagens de verificação por SMS?).

> **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 da mensagem para mensagens de verificação de endereço de e-mail, edite o modelo de Mensagem de verificação na guia Mensagens de seu grupo de usuários. Você pode selecionar um Tipo de verificação de Código ou Link ao editar seu 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 na guia 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. Selecione a guia Messaging (Sistema de mensagens). 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 API da AWS. 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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "stmnt1234567891234",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "email.cognito-idp.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "<your SES identity ARN>",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<your account number>"
                },
                "ArnLike": {
                    "aws:SourceArn": "<your user pool ARN>"
                }
            }
        }
    ]
}
```

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

Depois de criar seu grupo de usuários, você pode criar usuários com o uso do AWS Management Console, da AWS Command Line Interface ou da API do 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.

Desenvolvedores e administradores podem executar as seguintes tarefas:

- Crie um perfil de novo usuário usando o AWS Management Console ou chamando a API `AdminCreateUser`.
- Defina os valores dos atributos do usuário.
- Crie atributos personalizados.
- Defina o valor dos atributos personalizados imutáveis nas solicitações da API `AdminCreateUser`. Esse recurso não está disponível no console do Amazon Cognito.
- Especifique a senha temporária ou permita que o Amazon Cognito gere uma automaticamente.
- Especifique se endereços de e-mail e números de telefone fornecidos são marcadas como verificados para novos usuários.
- Especifique mensagens de convite personalizadas de SMS e e-mail para novos usuários por meio do AWS Management Console ou de um trigger Lambda de mensagem personalizada. Para obter mais informações, consulte Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda.
- 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`.

> **Note**
>
> Esta ação não pode ser executada no momento com o uso do AWS Management Console.

- Suprimir o envio da mensagem de convite quando o usuário for criado.
- Especifique um limite de tempo de expiração para a conta de usuário (até 90 dias).
- Permita que os usuários se cadastrem ou exija que novos usuários sejam adicionados apenas pelo administrador.

Fluxo de autenticação para usuários criados por administradores ou desenvolvedores

O fluxo de autenticação para esses usuários inclui a etapa extra para enviar a nova senha e fornecer qualquer valor ausente para atributos necessários. As etapas são descritas a seguir; as etapas 5, 6 e 7 são específicas para esses usuários.

1. O usuário começa a fazer login pela primeira vez enviando o nome de usuário e a senha.
2. O SDK chama `InitiateAuth(Username, USER_SRP_AUTH)`.
3. O Amazon Cognito retorna o desafio `PASSWORD_VERIFIER` com bloqueio Salt & Secret.
4. O SDK executa os cálculos de Secure Remote Password (SRP – Senha remota protegida) e chama `RespondToAuthChallenge(Username, <SRP variables>, PASSWORD_VERIFIER)`.
5. O Amazon Cognito retorna o desafio `NEW_PASSWORD_REQUIRED`. O corpo desse desafio inclui os atributos atuais do usuário e quaisquer atributos necessários em seu grupo de usuários que no momento não têm um valor no perfil do usuário. Para obter mais informações, consulte RespondToAuthChallenge.
6. O usuário é alertado e insere uma nova senha e qualquer valor ausente para atributos necessários.
7. O SDK chama `RespondToAuthChallenge(Username, <New password>, <User attributes>)`.
8. Se o usuário exigir um segundo fator para MFA, o Amazon Cognito retornará o desafio SMS_MFA e o código será enviado.
9. Quando o usuário tiver alterado com êxito sua senha e, opcionalmente, fornecido valores atribuídos ou concluído a MFA, ele será conectado, e os tokens serão emitidos.

Quando o usuário tiver cumprido todos os desafios, o serviço do Amazon Cognito marcará o usuário como confirmado, emitirá uma ID e atualizará os tokens para ele. Para obter mais informações, consulte Como usar tokens com grupos de usuários.

Como criar um novo usuário no AWS Management Console

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

É possível definir configurações para complexidade mínima de senha e se os usuários podem se cadastrar usando APIs públicas 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. Selecione a guia Sign-in experience (Experiência de acesso) e localize Password policy (Política de senha). Selecione a opção 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 Save changes (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. Selecione a guia Sign-up experience (Experiência de cadastro) e localize Self-service sign-up (Cadastro por autoatendimento). Selecione Edit (Editar).
4. Escolha se deseja Enable self-registration (Habilitar a autoinscrição). Normalmente, a autoinscrição é usada com clientes de aplicações públicas que precisam inscrever novos usuários em seu grupo de usuários sem distribuir um segredo de cliente ou credenciais da API do 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 Save changes (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. Selecione a guia Messaging (Sistema de mensagens) e localize os Message templates (Modelos de mensagens). Selecione Verification messages (Mensagens de verificação), Invitation messages (Mensagens de convite) ou MFA messages (Mensagens de MFA) e escolha Edit (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 Save changes (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 Save changes (Salvar alterações).

    c. Mensagens de MFA

    i. Insira um modelo personalizado de SMS message (Mensagem SMS) para mensagens de SMS (SMS).

    ii. Escolha Save changes (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 AWS CLI ou a API, 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. Escolha a guia Users (Usuários) e, em seguida, Create a user (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.

   > **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 na guia Messaging (Sistema de mensagens) 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. Qualquer senha temporária deve aderir à política de senha do grupo de usuários.
8. Escolha Create (Criar).
9. Escolha a guia Users (Usuários) e, em seguida, a entrada User name (Nome de usuário) para o usuário. 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.

Você pode criar e gerenciar grupos em um grupo de usuários a partir do AWS Management Console, das APIs 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 Preços 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 ARNs de função diferentes, a requisição `cognito:preferred_role` 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 mais informações Controlar o acesso a APIs HTTP 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 do serviço 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.

Como criar um novo grupo no AWS Management Console

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. Escolha a guia Groups (Grupos) e então escolha Create a group (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 ter 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 ter 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

Depois de criar seu grupo de usuários, você pode visualizar e gerenciar usuários usando o AWS Management Console, bem como a AWS Command Line Interface ou a API do Amazon Cognito. Este tópico descreve como você pode visualizar e pesquisar usuários usando o AWS Management Console.

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 credenciais da AWS.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Selecione a guia Users (Usuários) e, em seguida, 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 credenciais da AWS.

2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Selecione a guia Users (Usuários) e, em seguida, 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.

Como pesquisar atributos de usuários

Se você já tiver criado um grupo de usuários, poderá pesquisar no painel Users (Usuários) no AWS Management Console. Você também pode usar a API ListUsers do Amazon Cognito, que aceita um parâmetro Filter (Filtro).

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 no 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 mais informações sobre filtragem no lado do servidor e no lado do cliente, consulte Filtrar resultados da AWS CLI no Guia do usuário da AWS Command Line Interface.

Como pesquisar usuários usando o AWS Management Console

Se você já tiver criado um grupo de usuários, poderá pesquisar no painel Users (Usuários) no AWS Management Console.

Pesquisas do AWS Management Console são sempre pesquisas de prefixo ("começa com").

Para pesquisar um usuário no console do Amazon Cognito

1. Acesse o console do Amazon Cognito. Podem ser solicitadas suas credenciais da AWS.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Escolha a guia Users (Usuários) e insira o nome de usuário do 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 da sua aplicação, use a API ListUsers do Amazon Cognito. 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"
}
```

Como recuperar contas de usuário

O parâmetro `AccountRecoverySetting` permite personalizar qual método um usuário pode usar para recuperar a senha ao chamar a API `ForgotPassword`. `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 Cognito envia uma verificação para apenas um dos métodos especificados.

Por exemplo, `admin_only` é um valor usado quando o administrador não deseja que o próprio usuário recupere sua conta e, em vez disso, exige que ele entre em contato com o administrador para redefinir sua conta. Você não pode usar `admin_only` com nenhum outro mecanismo de recuperação de conta.

Se você não especificar `AccountRecoverySetting`, o Amazon Cognito usará o mecanismo legado para determinar o método de recuperação de senha. Nesse caso, o Cognito usa primeiro um telefone verificado. Se o telefone verificado não for encontrado para o usuário, o Cognito fará fallback e usará o e-mail verificado em seguida.

Para obter mais informações sobre `AccountRecoverySetting`, consulte CreateUserPool e UpdateUserPool na Referência de API do provedor de identidade do Amazon Cognito.

Comportamento para esquecimento da senha

Em uma determinada hora, permitimos entre 5 e 20 tentativas para um usuário solicitar ou inserir um código de redefinição de senha como parte das ações forgot-password e confirm-forgot-password. O valor exato depende dos parâmetros de risco associados às solicitações. Observe que esse comportamento está sujeito a alterações.

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.

Tópicos

Como importar usuários para grupos de usuários com um acionador Lambda de migração de usuário

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 da interface do usuário hospedada do Amazon Cognito. Para obter mais informações sobre como facilitar o login com as APIs do Amazon 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 sua aplicação tiver uma interface do usuário de login personalizada que você criou com um AWS SDK, ela precisará usar InitiateAuth ou AdminInitiateAuth com o fluxo `USER_PASSWORD_AUTH` 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 logs, as senhas aparecerão no CloudWatch Logs.

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. Na aplicação, solicite ao usuário o código de confirmação e uma nova senha e envie essas informações para o Amazon Cognito em uma solicitação de API ConfirmForgotPassword. Também é possível usar as páginas integradas para o fluxo de esquecimento da senha na interface do usuário hospedada do Amazon Cognito.

Como importar usuários para grupos de usuários com base em um arquivo CSV

Você pode importar usuários para um grupo de usuários do Amazon Cognito. As informações do usuário são importadas de um arquivo .csv com formatação especial. 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. Portanto, os usuários estarão em um estado RESET_REQUIRED quando importados usando esse método.

É possível definir as senhas dos usuários com uma solicitação de API AdminSetUserPassword que define o parâmetro `Permanent` como `true`.

> **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.

As etapas básicas são:

1. Crie uma função do Amazon CloudWatch Logs no console do 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 o CloudWatch para verificar o log de eventos.
7. Solicite que os usuários importados redefinam suas senhas.

Tópicos

- Criar o perfil do IAM do CloudWatch Logs
- Criar o arquivo CSV de importação do usuário
- Como criar e executar o trabalho de importação do grupo de usuários do Amazon Cognito
- Como visualizar os resultados de importação de grupo de usuários no console do CloudWatch
- Solicitação de redefinição de senha aos usuários importados

Criar o perfil do IAM do CloudWatch Logs

Se você estiver usando a CLI ou a API do Amazon Cognito, precisará criar uma função do IAM do CloudWatch. O procedimento a seguir descreve como criar um perfil do IAM que o Amazon Cognito possa usar para gravar os resultados do trabalho de importação no CloudWatch Logs.

> **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.

Como criar o perfil do IAM do CloudWatch Logs para importação do grupo de usuários (AWS CLI, API)

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Criar um perfil do IAM para um AWS service (Serviço da AWS). Para obter instruções detalhadas, consulte Creating a role for an AWS service (Serviço da AWS) (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. Substitua *`REGION`* pela Região da AWS do seu grupo de usuários, por exemplo `us-east-1`. Substitua *`ACCOUNT`* pelo ID da sua Conta da AWS, por exemplo `111122223333`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
```

```
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:REGION:ACCOUNT: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. Escolha 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:

```
{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "cognito-idp.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }
```

7. Escolha Update policy.
8. Anote o Nome de região da Amazon (ARN) da função. 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. Podem ser solicitadas suas credenciais da AWS.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Escolha a guia Users.
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, execute o seguinte comando da CLI, em que *USER_POOL_ID* é 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 Atributos de grupo de usuários.

- 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
- cognito:mfa_enabled
- 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.

- Se estiverem presentes, os valores de birthdate (data de nascimento) devem estar no formato *mm/dd/aaaa*. 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 é obrigatório. Se você tiver definido a autenticação multifator (MFA) para ser necessária no grupo de usuários, esse campo deverá ser `true` para todos os usuários. Se você tiver definido a MFA para ser desativada, este campo deverá ser `false` para todos os usuários. Se você tiver definido a MFA como opcional, esse campo poderá ser `true` ou `false`, mas não poderá ficar vazio.
- 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 a 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. Podem ser solicitadas suas credenciais da AWS.
2. Escolha User Pools (Grupos de usuários).
3. Escolha um grupo de usuários existente da lista.
4. Escolha a guia Users.
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 Criar o perfil do IAM do CloudWatch Logs.
8. 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.
9. Se você criou o trabalho, mas não o iniciou, poderá iniciá-lo mais tarde. Na guia Users (Usuários), em Import users (Importar usuários), escolha o trabalho de importação e selecione Start (Iniciar). Você também pode enviar uma solicitação da API StartUserImportJob de um AWS SDK.
10. Monitore o andamento do trabalho de importação de usuários na guia Users (Usuários) em Import users (Importar usuários). Se o trabalho não for bem-sucedido, você poderá selecionar o valor do Status. Para obter mais detalhes, selecione View the CloudWatch logs for more details

(Visualizar os logs do CloudWatch para obter mais detalhes) e analisar os problemas no console do CloudWatch Logs.

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

Após criar o arquivo CSV, crie um trabalho de importação de usuário executando o seguinte comando da CLI, no qual *JOB_NAME* é o nome que você está escolhendo para o trabalho, *USER_POOL_ID* é o ID do grupo de usuários no qual os novos usuários serão adicionados e *ROLE_ARN* é o ARN do perfil que você recebeu em Criar o perfil do IAM do CloudWatch Logs:

```
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* retornado 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 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
    }
}
```

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.

Como descrever um trabalho de importação de usuário

Para obter uma descrição do trabalho de importação de usuário, use o comando a seguir, em que *USER_POOL_ID* é o ID do grupo de usuários e *JOB_ID* é o job ID retornado quando você criou o trabalho de importação de usuário.

```
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
    }
}
```

Na saída de exemplo anterior, o *PRE_SIGNED_URL* é o URL no qual você fez upload do arquivo CSV. O *ROLE_ARN* é o ARN de função do CloudWatch Logs que você recebeu quando criou 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 string *PAGINATION_TOKEN* após o segundo trabalho indica que há outros resultados para esse comando de listagem. 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
    }
}
```

Como visualizar os resultados de importação de grupo de usuários no console do CloudWatch

Você pode visualizar os resultados do trabalho de importação no console do Amazon CloudWatch.

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 AWS Management Console e abra o console do CloudWatch 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 log 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

O status dos usuários importados com êxito é 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 do Amazon CloudWatch Logs não pode ser presumida, 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

Na primeira vez que cada usuário importado fizer login e inserir uma senha, será solicitado que ele insira uma nova senha. O procedimento a seguir descreve a experiência do usuário em uma aplicação personalizada com usuários locais após a importação de um arquivo CSV. Se os usuários fizerem login com a interface de usuário hospedada, o Amazon Cognito solicitará que eles definam uma nova senha ao fazerem login pela primeira vez.

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.

Atributos de grupo de usuários

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 AWS Management Console. 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.

> **Note**
>
> Alguns documentos e padrões se referem a atributos como membros.

Tópicos

- Nome de usuário e nome de usuário preferencial
- Personalização dos atributos de login
- Atributos personalizados
- Permissões e escopos do atributo

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:

- `address`
- `birthdate`
- `email`
- `family_name`
- `gender`
- `given_name`
- `locale`
- `middle_name`
- `name`
- `nickname`
- `phone_number`
- `picture`
- `preferred_username`
- `profile`
- `sub`
- `updated_at`
- `website`
- `zoneinfo`

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.

> **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 de dez caracteres válida no formato AAAA-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 na guia Usuários do AWS Management Console 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 ter 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 na guia Usuários do AWS Management Console 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. AWS Management Console 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. Escolha a guia Sign-up experience (Experiência de cadastro).
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 ter 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 ter 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 número de telefone antes de poderem fazer login com ele	Sim	Não
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)

Você pode ativar aliases se quiser permitir que os usuários optem por inserir o nome de usuário ou outros valores de atributo ao fazerem login. Por padrão, os usuários fazem login com nome de usuário e senha. O nome de usuário 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.

Como configurar um grupo de usuários para que seja possível fazer login com um nome de usuário preferido

1. Acesse o Amazon Cognito no AWS Management Console. Se o console solicitar, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. No canto superior direito da página, escolha Create a user pool (Criar um grupo de usuários) para iniciar o assistente de criação de grupo de usuários.
4. Em Configure sign-in experience (Configurar experiência de acesso), escolha a identidade Provider types (Tipos de provedores) que você deseja associar ao grupo de usuários.
5. Em Cognito user pool sign-in options (Opções de acesso do grupo de usuários do Cognito), escolha qualquer combinação de User name (Nome de usuário), Email (E-mail) e Phone number (Número de telefone).
6. Em Requisitos de nome do usuário, selecione Permitir que os usuários façam login com um nome de usuário preferido para permitir a definição de um nome de usuário alternativo para login.
7. Selecione Next (Próximo) e, em seguida, conclua todas as etapas no assistente.

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.

> **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.

Para configurar um grupo de usuários para cadastro e login com um endereço de e-mail ou número de telefone

1. Acesse o Amazon Cognito no AWS Management Console. Se o console solicitar, insira suas AWS credenciais.
2. Escolha User Pools (Grupos de usuários).
3. No canto superior direito da página, escolha Create a user pool (Criar um grupo de usuários) para iniciar o assistente de criação de grupo de usuários.
4. Em Cognito user pool sign-in options (Opções de login do grupo de usuários do Cognito), escolha qualquer combinação de Email (E-mail) e Phone number (Número de telefone) que represente os atributos de alias que o usuário pode usar para fazer login.
5. Selecione Next (Próximo) e conclua todas as etapas restantes no assistente.

> **Note**
>
> Não é necessário marcar o endereço de e-mail ou número de telefone como atributos obrigatórios para o grupo de usuários.

Para implementar a opção 2 no aplicativo

1. Chame a API `CreateUserPool` para criar o grupo de usuários. Defina o parâmetro `UserNameAttributes` como `phone_number`, `email` ou `phone_number | email`.
2. Chame a API `SignUp` e envie um endereço de e-mail ou número de telefone para o parâmetro `username` da API. Esta API faz o seguinte:

   - Se a string `username` estiver no formato válido de endereço de e-mail, 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, 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.
   - A API `SignUp` gera um UUID persistente para o usuário e o utiliza internamente como o atributo nome de usuário imutável. Este UUID possui o mesmo valor reivindicado pelo `sub` no token de identidade do usuário.
   - Se a string `username` contiver um endereço de e-mail ou número de telefone já em uso, a API `SignUp` retornará uma exceção.

É possível usar um endereço de e-mail ou número de telefone como um alias em vez do nome de usuário em todas as APIs, exceto na API `ListUsers`. Quando você chama `ListUsers`, é possível pesquisar pelo atributo `email` ou `phone_number`. Se você pesquisar por `username`, será necessário fornecer o nome de usuário real, não o alias.

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.

Todo atributo personalizado tem as seguintes características:

- Você pode defini-lo como uma string ou um número. O Amazon Cognito grava valores de atributo personalizados no token de ID somente como strings.
- 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 ter 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 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. AWS Management Console 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. Selecione a guia Sign-up experience (Experiência de cadastro) e na seção Custom attributes (Atributos personalizados), escolha Add custom attributes (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, ao criar um novo cliente de aplicativo, você concede permissões de leitura e gravação ao seu aplicativo 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 de atributos ao criar um cliente de aplicativo. Conceda ao cliente do aplicativo acesso ao conjunto mínimo de atributos de usuário necessários para a operação do seu aplicativo.

> **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 (AWS Management Console)

1. Acesse o Amazon Cognito no. AWS Management Console 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. Selecione a guia App integration (Integração de aplicação) e na seção App clients (Clientes da aplicação), escolha um cliente da aplicação na lista.
5. Na seção Attribute read and write permissions (Permissões de gravação e leitura de atributos), escolha Edit (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 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 a aplicação tentar definir um valor para um atributo que não esteja autorizado a gravar, 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.
- `AdminCreateUser`: você fornece um valor para um atributo imutável.

Para receber informações sobre os escopos que você pode atribuir aos clientes da aplicação, consulte Escopos, M2M e autorização de API com servidores de recursos.

Você pode alterar os escopos e as permissões de atributo após ter criado o grupo de usuários.

Como adicionar requisitos de senha do grupo de usuários

Senhas fortes e complexas são a melhor prática de segurança para seu grupo de usuários. Especialmente em aplicativos abertos à Internet, senhas fracas podem expor as credenciais de seus 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 recursos avançados de segurança e ACLs AWS WAF da web, 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 registro externo da idade da senha, seu aplicativo ou um acionador do Lambda pode pesquisar a idade da senha de um usuário e exigir uma redefinição após um determinado período.

Você pode configurar seu grupo de usuários para exigir uma complexidade mínima de senha que esteja em conformidade com seus padrões de segurança. As senhas complexas têm um comprimento mínimo de pelo menos oito caracteres. Eles também incluem uma mistura de caracteres maiúsculos, numéricos e especiais.

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 a guia Experiência de login.
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 pelo menos um dos seguintes caracteres nas senhas. 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 o tamanho máximo da senha.

   - 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.

   - Para redefinir o acesso de uma conta de usuário expirada, siga um destes procedimentos.
     - Exclua o perfil de usuário e crie outro.
     - Defina uma nova senha permanente em uma solicitação de AdminSetUserPasswordAPI.
     - Gere um novo código de confirmação em uma solicitação de AdminResetUserPasswordAPI.

Configurações de e-mail para grupos de usuários do Amazon Cognito

Determinados eventos no aplicativo do cliente do grupo de usuários 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 incorporada ao serviço 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	E-mail, SMS	Código	Não	N/D
Convite	AdminCreateUser	E-mail, SMS	Código	Sim	Mensagem de convite
Autorregistro	SignUp	E-mail, SMS	código, link	Sim	Mensagem de verificação
Verificação de endereço de e-mail ou número de telefone	UpdateUserAttributes	E-mail, SMS	Código	Sim	Mensagem de verificação
Autenticação multifator (MFA)	AdminInitiateAuth, InitiateAuth	SMS	Código	Sim[1]	Mensagem de MFA

[1] Para mensagens SMS.

O Amazon SES cobra por mensagens de e-mail. Para obter mais informações, consulte Definição de preço do Amazon SES.

Configuração de e-mail padrão

O Amazon Cognito pode usar sua configuração de e-mail padrão para lidar com 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 pelo 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 não entregue for entregue posteriormente, você não poderá controlar sua remoção da lista de supressão enquanto seu 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 `FROM` e-mail personalizado, o Amazon Cognito usa uma identidade `no-reply@verificationemail.com` verificada na mesma região.

Compatível com versões anteriores

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 (Norte da Virgínia)

- Oeste dos EUA (Oregon)
- Europa (Irlanda)

Esse recurso oferece suporte à continuidade dos recursos do grupo de usuários que você pode ter criado para atender aos requisitos do Amazon Cognito quando o serviço foi lançado. 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 `FROM` e-mail personalizado, 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 ter 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 `FROM` e-mail personalizado, a política de autorização de envio do Amazon SES para sua identidade verificada em cada região alternativa deve confiar no responsável pelo serviço principal do Amazon Cognito da região de origem. Para ter mais informações, consulte Para conceder permissões para usar a configuração de e-mail padrão. Com a configuração de e-mail do Amazon SES `DEVELOPER` nessas regiões, você deve usar uma identidade verificada na primeira região listada e configurá-la para confiar no principal 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 ser confiável`cognito-idp.me-central-1.amazonaws.com`. Na configuração de e-mail padrão sem um endereço de `FROM` e-mail personalizado, 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
>
> - Seu EmailSendingAccount é`COGNITO_DEFAULT`.
> - Você quer usar um `FROM` endereço personalizado.
> - Seu grupo de usuários envia e-mails em uma região alternativa.
> - Seu grupo de usuários tem uma segunda região [1]alternativa especificada na tabela de regiões suportadas pelo 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 suportadas pelo Amazon SES
Leste dos EUA (Norte da Virgínia)	Compatível com versões anteriores	Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Leste dos EUA (Ohio)	Compatível com versões anteriores	Leste dos EUA (Ohio), Leste dos EUA (Norte da Virgínia)

Região do grupo de usuários	Opção de região	Regiões suportadas pelo Amazon SES
		, 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)	Compatível com versões anteriores	Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Canadá (Central)	Compatível com versões anteriores	Canadá (Central), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Ásia-Pacífico (Tóquio)	Compatível com versões anteriores	Ásia-Pacífico (Tóquio), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Ásia-Pacífico (Seul)	Compatível com versões anteriores	Ásia-Pacífico (Seul), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Ásia-Pacífico (Mumbai)	Compatível com versões anteriores	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 (Cingapura) [1]

Região do grupo de usuários	Opção de região	Regiões suportadas pelo Amazon SES
Ásia-Pacífico (Singapura)	Compatível com versões anteriores	Ásia-Pacífico (Singapura), Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Ásia-Pacífico (Sydney)	Compatível com versões anteriores	Ásia-Pacífico (Sydney), Leste dos EUA (Norte da Virgínia) , Oeste dos EUA (Oregon), Europa (Irlanda)
Asia Pacific (Osaka)	Somente na região	Asia Pacific (Osaka)
Ásia-Pacífico (Jacarta)	Somente na região	Ásia-Pacífico (Jacarta)
Ásia-Pacífico (Melbourne)	Região alternativa	Ásia-Pacífico (Sydney), Ásia-Pacífico (Cingapura) [1]
Europa (Irlanda)	Compatível com versões anteriores	Leste dos EUA (Norte da Virgínia), Oeste dos EUA (Oregon), Europa (Irlanda)
Europa (Londres)	Compatível com versões anteriores	Europa (Londres), Leste dos EUA (Norte da Virgínia) , Oeste dos EUA (Oregon), Europa (Irlanda)
Europa (Paris)	Somente na região	Europa (Paris)
Europa (Frankfurt)	Compatível com versões anteriores	Europa (Frankfurt), Leste dos EUA (Norte da Virgínia) , Oeste dos EUA (Oregon), Europa (Irlanda)
Europa (Zurique)	Região alternativa	Europa (Frankfurt), Europa (Londres) [1]

Região do grupo de usuários	Opção de região	Regiões suportadas pelo Amazon SES
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 (Barém)	Somente na região	Oriente Médio (Barém)
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 `FROM` endereço personalizado, configure `EmailConfiguration` com um `SourceArn` parâmetro 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

Omita essa etapa se você 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 que você usa para concedê-las dependem se você está 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 definido como`EmailSendingAccount`. `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 padrãono`no-reply@verificationemail.com`.
- Um endereço FROM personalizado de seus endereços de e-mail ou domínios 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 o diretor do serviço Amazon Cognito envie mensagens de e-mail. Consulte a tabela Regiões para obter mais informações. 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 de serviço global no exemplo a seguir.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "stmnt1234567891234",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "email.cognito-idp.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "<your SES identity ARN>",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<your account number>"
                },
                "ArnLike": {
                    "aws:SourceArn": "<your user pool ARN>"
                }
            }
        }
    ]
}
```

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 a Europa (Londres). Você deve atualizar a política de autorização de envio nas duas regiões.

Sua política de autorização de envio em cada uma das regiões alternativas deve permitir que o responsável pelo serviço Amazon Cognito na região de opt-in do grupo de usuários envie mensagens de e-mail. Consulte a tabela Regiões para obter mais informações. Se sua região estiver marcada como Região alternativa, configure suas políticas de autorização de envio com o diretor de serviço regional, como no exemplo a seguir. Substitua o exemplo de identificador de região *`me-central-1`* pelo ID de região necessário, conforme necessário.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "cognito-idp.me-central-1.amazonaws.com"
                ]
            },
            "Action": [
                "SES:SendEmail",
                "SES:SendRawEmail"
            ],
            "Resource": "<your SES identity ARN>",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "<your account number>"
                },
                "ArnLike": {
                    "aws:SourceArn": "<your user pool ARN>"
                }
            }
        }
    ]
```

```
}
```

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 os endereços de e-mail FROM e REPLY-TO. Por exemplo, se o domínio verificado for exemplo.com, você poderá definir no-reply@exemplo.com como os endereços de e-mail FROM e REPLY-TO.

Omita esse procedimento se quiser usar a configuração e o endereço de e-mail 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. Escolha a guia Messaging (Sistema de mensagens), localize Email configuration (Configuração de e-mail), escolha Edit (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. Se você estiver enviando uma mensagem de texto por meio do Amazon Cognito ou do Amazon SNS pela primeira vez, o Amazon SNS o colocará em um ambiente de área restrita para testes. No ambiente de área restrita para testes, você pode testar suas aplicações para mensagens de texto SMS. Na área restrita para testes, as mensagens só podem ser enviadas para números de telefone verificados.

O Amazon SNS cobra por mensagens de texto SMS. Para obter mais informações, consulte Definição de preço do Amazon SNS.

> **Note**
>
> 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 as

regulamentações relacionadas a mensagens SMS em países onde seus usuários possam morar e manter a configuração de suas mensagens SMS atualizada. Para ter mais informações, consulte Mensagens de texto móveis (SMS) no Guia do desenvolvedor do Amazon Simple Notification Service.
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.

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	E-mail, SMS	Código	Não	N/D
Convite	AdminCreateUser	E-mail, SMS	Código	Sim	Mensagem de convite
Autorregistro	SignUp	E-mail, SMS	código, link	Sim	Mensagem de verificação
Verificação de endereço de e-mail ou número de telefone	UpdateUserAttributes	E-mail, SMS	Código	Sim	Mensagem de verificação

Atividade	Operação de API	Opções de entrega	Opções de formato	Personalizável	Modelo de mensagem
Autenticação multifator (MFA)	AdminInitiateAuth, InitiateAuth	SMS, aplicativo autenticador	Código	Sim[1]	Mensagem de MFA

[1] Para mensagens SMS.

Configurar mensagens SMS pela primeira vez nos grupos de usuários do Amazon Cognito

O Amazon Cognito usa o Amazon SNS para enviar mensagens SMS para 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. Na primeira vez que você configura o Amazon SNS para enviar mensagens de texto SMS em uma determinada região Região da AWS, o Amazon SNS coloca Conta da AWS você na sandbox de SMS dessa região. O Amazon SNS usa a sandbox para evitar fraudes e abusos e para atender aos requisitos de conformidade. Quando você Conta da AWS está na sandbox, o Amazon SNS impõe algumas restrições. Por exemplo, você pode enviar mensagens de texto para até dez números de telefone verificados com o Amazon SNS. Enquanto você Conta da AWS permanecer na sandbox, não use a configuração do Amazon SNS para aplicativos que estão 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.

Para enviar mensagens de texto SMS para usuários do grupo de usuários

1. Prepare um perfil do IAM que o Amazon Cognito possa usar para enviar mensagens SMS com o Amazon SNS
2. Escolha o Região da AWS para mensagens SMS do Amazon SNS
3. Obter uma identidade de origem para enviar mensagens SMS a números de telefone dos EUA
4. Confirmar se você está na sandbox SMS
5. Retirar sua conta da sandbox do Amazon SNS
6. Verificar os números de telefone do Amazon Cognito no Amazon SNS
7. Concluir a configuração do grupo de usuários no Amazon Cognito

Prepare um perfil do IAM que o Amazon Cognito possa usar para enviar mensagens SMS com o Amazon SNS

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 IAM role selection (Seleção de perfis do IAM) na guia Messaging (Sistema de mensagens) de seu 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 faz isso em nome do grupo de usuários na condição `aws:SourceArn` e da Conta da AWS na condição `aws:SourceAccount`.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Principal": {
            "Service": "cognito-idp.amazonaws.com"
        },
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "<your account number>"
            },
            "ArnLike": {
                "aws:SourceArn": "<your user pool ARN>"
            }
        }
    }]
}
```

Você pode especificar um ARN do grupo de usuários exato ou um ARN curinga no valor da condição `aws:SourceArn`. Pesquise os ARNs dos seus grupos de usuários no AWS Management Console ou com uma solicitação de DescribeUserPoolAPI.

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 do Amazon SNS

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.

> **Note**
>
> No AWS Management Console, você só pode alterar a região dos recursos de SMS depois de mudar para a nova experiência de console do Amazon Cognito.

Escolha uma região para recursos de SMS na etapa Configurar a entrega de mensagens do novo assistente de grupo de usuários. Você também pode escolher Edit (Editar) em SMS na guia Messaging (Sistema de mensagens) 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 define a cota de gastos para todas as novas contas como USD 1,00 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 Solicitar aumentos de sua cota de gastos mensais de SMS para o Amazon SNS no Guia do desenvolvedor do Amazon Simple Service Notification.

Você pode enviar mensagens SMS a qualquer Amazon Cognito Region (Região do Amazon Cognito) descrita na tabela a seguir com recursos do Amazon SNS na Amazon SNS Region (Região do Amazon SNS) correspondente.

Região do Amazon Cognito	Região do Amazon SNS
Leste dos EUA (Ohio)	Leste dos EUA (Ohio), Leste dos EUA (Norte da Virgínia)
Canadá (Central)	Canadá (Central), Leste dos EUA (Norte da Virgínia)
Europa (Frankfurt)	Europa (Frankfurt), Europa (Irlanda)
Europa (Londres)	Europa (Londres), Europa (Irlanda)
Ásia-Pacífico (Seul)	Ásia-Pacífico (Tóquio)
Leste dos EUA (Norte da Virgínia)	Leste dos EUA (N. da Virgínia)
Oeste dos EUA (N. da Califórnia)	Oeste dos EUA (N. da Califórnia)
Oeste dos EUA (Oregon)	Oeste dos EUA (Oregon)
Ásia-Pacífico (Mumbai)	Ásia-Pacífico (Mumbai), Ásia-Pacífico (Singapura)
Ásia-Pacífico (Hyderabad)	Ásia-Pacífico (Hyderabad)
Ásia-Pacífico (Singapura)	Ásia-Pacífico (Singapura)
Ásia-Pacífico (Sydney)	Ásia-Pacífico (Sydney)
Ásia-Pacífico (Tóquio)	Ásia-Pacífico (Tóquio)
Ásia-Pacífico (Jacarta)	Ásia-Pacífico (Jacarta)
Asia Pacific (Osaka)	Asia Pacific (Osaka)
Ásia-Pacífico (Melbourne)	Ásia-Pacífico (Melbourne)

Região do Amazon Cognito	Região do Amazon SNS
Europa (Irlanda)	Europa (Irlanda)
Europa (Paris)	Europa (Paris)
Europa (Estocolmo)	Europa (Estocolmo)
Europa (Milão)	Europa (Milão)
Europa (Espanha)	Europa (Espanha)
Oriente Médio (Barém)	Middle East (Bahrain)
South America (São Paulo)	América do Sul (São Paulo)
Israel (Tel Aviv)	Israel (Tel Aviv)
África (Cidade do Cabo)	África (Cidade do Cabo)
Oriente Médio (Emirados Árabes Unidos)	Oriente Médio (Emirados Árabes Unidos)
Europa (Zurique)	Europa (Zurique)

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.

Desde 1.º de junho de 2021, as operadoras americanas 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 Requesting a number (Solicitar um número) no Guia do usuário do Amazon Pinpoint.

Se você operar da seguinte forma Regiões da AWS, deverá abrir um AWS Support tíquete para obter uma identidade de origem. Para obter instruções, consulte Solicitar suporte para mensagens SMS no Guia do desenvolvedor do Amazon Simple Notification Service.

- Leste dos EUA (Ohio)
- Europa (Estocolmo)
- Europa (Paris)
- Europa (Milão)
- Oriente Médio (Bahrein)
- South America (São Paulo)
- Oeste dos EUA (N. da Califórnia)

Quando você tem mais de uma identidade de origem na mesma Região da AWS, o Amazon SNS 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 Perguntas frequentes do Amazon SNS.

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. Selecione a guia Messaging (Sistema de mensagens).
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:

   ```
   You are currently in the SMS Sandbox and cannot send SMS messages to unverified numbers.
   ```

   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. Escolha o link Amazon SNS na mensagem. 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).
   ```

Retirar sua conta da sandbox do Amazon SNS

Se você estiver testando seu aplicativo e só precisar enviar mensagens SMS para números de telefone que seus administradores possam verificar, ignore esta etapa.

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 recursos do Amazon SNS que você deseja que o Amazon Cognito use, você pode verificar os números de telefone dos EUA enquanto permanece na sandbox do Conta da AWS SMS. Quando seu ambiente Amazon SNS está em produção, você não precisa verificar números de telefone de usuários no Amazon SNS para enviar mensagens SMS aos usuários.

Para obter instruções detalhadas, consulte Sair da sandbox de SMS no Guia do desenvolvedor do Amazon Simple Notification Service.

Verificar os números de telefone do Amazon Cognito no Amazon SNS

Se você tiver removido sua conta da área restrita para testes de SMS, ignore esta etapa.

Quando você estiver na área restrita para testes de SMS, poderá enviar mensagens para qualquer número de telefone verificado com o Amazon SNS.

Para verificar um número de telefone, faça o seguinte:

1. Adicione um número de telefone de destino da área restrita para testes à seção Text messaging (SMS) [Mensagens de texto (SMS)] do console do Amazon SNS.
2. Receba uma mensagem SMS com um código no número de telefone fornecido.
3. Digite o código de verificação na mensagem SMS no console do Amazon SNS.

Para obter instruções detalhadas, consulte Adicionar e verificar números de telefone na sandbox SMS no Guia do desenvolvedor do Amazon Simple Notification Service.

> **Note**
>
> O Amazon SNS limita o número de telefones de destino que você pode verificar enquanto estiver na área restrita para testes de SMS. Consulte Sandbox de SMS no Guia do desenvolvedor do Amazon Simple Notification Service.

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.

Como usar tokens com grupos de usuários

Autentique usuários e conceda acesso a recursos com tokens. 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 reivindicações como o `scope` que o usuário autenticado pode usar para acessar APIs de terceiro, operações de API de autoatendimento de usuários do Amazon Cognito e o Endpoint do 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 Base64. Você pode decodificar qualquer ID ou token de acesso do Amazon Cognito de Base64 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. É possível usar os tokens para conceder aos usuários acesso aos recursos e APIs de downstream, 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 declarações incluem escopos do OAuth 2.0, associação a grupos de usuários, atributos de 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 Preços do Amazon Cognito.

Tópicos

Como usar o token de 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 um token 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 seu usuário faz login com a UI hospedada ou um provedor de identidades federadas (IdP), o Amazon Cognito define cookies de sessão válidos por 1 hora. Se você usar a interface do usuário ou federação hospedada e especificar uma duração mínima inferior a 1 hora para seus tokens de acesso e ID, seus 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. É possível ver os IDs de chave de assinatura do grupo de usuários no endpoint `jwks_uri`.

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 utilizam um algoritmo criptográfico RS256, 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

Este é um exemplo de carga útil de um token de ID. Ela contém alegações sobre o usuário autenticado. Para obter mais informações sobre as declarações padrão do OpenID Connect (OIDC), consulte a lista de declarações padrão do OIDC. Você pode adicionar reivindicações de seu próprio design com umAcionador 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 identidade 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 declaração `nonce` vem de um parâmetro com o mesmo nome que você pode adicionar às solicitações feitas ao seu endpoint `authorize` do 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 ter 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 são sempre prefixados com`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 um token Web JSON

Como usar 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 os escopos do OAuth 2.0 em um token de acesso, derivados dos escopos personalizados que você adiciona ao grupo de usuários, é possível autorizar o 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.

O token de acesso do usuário é a permissão para solicitar mais informações sobre os atributos do usuário no Endpoint do UserInfo. O token de acesso do usuário também é permissão para ler e gravar atributos do usuário. O nível de acesso aos atributos que seu token de acesso concede depende das permissões atribuídas ao cliente da aplicação e dos escopos concedidos no token.

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 ter mais informações, consulte Como verificar um token 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 uma hora se você usar a IU hospedada. A UI hospedada do Amazon Cognito usa cookies válidos por uma hora. Se você inserir um mínimo de menos de uma hora, não obterá um tempo de validade menor.

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. É possível ver os IDs de chave de assinatura do grupo de usuários no endpoint `jwks_uri`.

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 utilizam um algoritmo criptográfico RS256, 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 reivindicações de seu próprio design com umAcionador 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",
   "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.<Region>.amazonaws.com/<your user pool ID>`

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.

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`.

`token_use`

A finalidade do token. Em um token de acesso, seu valor é `access`.

`scope`

Uma lista de escopos do Oauth 2.0 que definem o acesso que o token oferece. 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 do usuário no grupo de usuários.

Assinatura do token de acesso

A assinatura do token de acesso é calculada com base no cabeçalho e na carga útil do token JWT. Quando o token de ID for usado fora de uma aplicação em suas APIs da Web, sempre será necessário verificar essa assinatura antes de aceitá-lo. Para ter mais informações, consulte Como verificar um token Web JSON.

Como usar o token 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.

O Mobile SDK for iOS, Mobile SDK for Android, Amplify para iOS, Android e Flutter atualizarão automaticamente seu tokens de ID e de acesso se um token de atualização válido (não expirado) estiver presente. Os tokens de ID e de acesso têm uma validade mínima restante de dois minutos. Se o token de atualização tiver expirado, o usuário da aplicação precisará se reautenticar fazendo login novamente no grupo de usuários. Se o mínimo para o token de acesso e o token de ID estiver definido para 5 minutos, e você estiver usando o SDK, o token de atualização será usado continuamente para recuperar novos tokens de acesso e ID. Você verá o comportamento esperado em no mínimo 7 minutos ao invés de 5 minutos.

A conta do usuário em si nunca expira, desde que o usuário tenha feito login pelo menos uma vez antes do limite de tempo `UnusedAccountValidityDays` para novas contas.

Obter novos tokens de acesso e identidade com um token de atualização

Use a API ou a UI hospedada para iniciar a autenticação de tokens de atualização.

Para usar o token de atualização para obter novos tokens de ID e acesso com a API de grupos de usuários, use as operações da InitiateAuthAPI AdminInitiateAuthou. 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.

> **Note**
>
> Para usar a API de grupos de usuários do Amazon Cognito a fim de atualizar tokens para um usuário de interface de usuário hospedado, gere uma solicitação `InitiateAuth`.

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.

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 Como revogar tokens.

> **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.

Os usuários podem sair de todos os dispositivos em que estão conectados quando você revoga todos os tokens dos usuários usando as operações de API `GlobalSignOut` e `AdminUserGlobalSignOut`. 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 interface de usuário hospedados não expiram automaticamente, o usuário pode se autenticar novamente com um cookie de sessão, sem nenhuma solicitação adicional de credenciais. Depois de desconectar os usuários de interface de usuário hospedados, redirecione-os para o Endpoint de logout, em que 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, talvez os usuários queiram desautorizar alguns dispositivos nos quais fizeram login, atualizando continuamente a sessão. Para desconectar o usuário em um único dispositivo, 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. Operações de API autenticadas e não autenticadas de grupos de usuários do Amazon Cognito

Como revogar tokens

Você pode revogar um token de atualização para um usuário usando a AWS API. 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.

> **Note**
>
> Tokens JWT são independentes com uma assinatura e um tempo de validade que foi atribuído 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.

Antes de poder revogar um token para um cliente de grupo de usuários existente, você deve habilitar a revogação de token. Quando você cria um novo cliente do grupo de usuários, a revogação de token é habilitada 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`.

Quando você cria um novo cliente de grupo de usuários usando a AWS Management Console, a ou a AWS API AWS CLI, a revogação de token é ativada por padrão.

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.

Para criar ou modificar um cliente de aplicativo com a revogação de token ativada, inclua o seguinte parâmetro na sua solicitação CreateUserPoolClientou na sua solicitação de UpdateUserPoolClientAPI.

```
"EnableTokenRevocation": true
```

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.

> **Note**
>
> A solicitação para revogar um token de atualização deve incluir ID do cliente que foi usado para obter o token.

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 um token Web JSON

Essas etapas descrevem como verificar um JSON web token (JWT) do grupo de usuários.

Tópicos

Pré-requisitos

Sua biblioteca, o SDK ou o framework de software podem processar as tarefas nesta seção. AWS Os SDKs fornecem 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 SDKs AWS
- 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 seu usuário passa para 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.
- Que as reivindicações de token de acesso contêm os escopos corretos do OAuth 2.0.
- 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 à aplicação, considere como o Amazon Cognito monta os 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 os JWTs do Amazon Cognito enquanto você prepara a aplicação.

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`.

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`.

Assinatura

A assinatura não é decodificável em base64, como o cabeçalho e a carga útil. É um identificador RSA256 derivado de uma chave de assinatura e parâmetros que você pode observar no URI do JWKS.

O cabeçalho e a carga útil são JSON codificados em base64. É possível identificá-los pelos caracteres de abertura `eyJ` que são decodificados para o caractere inicial `{`. Se o usuário apresentar um JWT codificado em base64 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.

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 JWTs do grupo 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 utilizam um algoritmo criptográfico RS256, 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 como Base64urlUInt.

Modulus (**`n`**) RSA

O parâmetro `n` contém o valor de modulus da chave pública RSA. Ele é representado como um valor codificado como 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.

3. Use uma biblioteca de JWT para comparar a assinatura do emissor com a assinatura no token. A assinatura do emissor é derivada da chave pública (o módulo RSA `"n"`) do `kid` em jwks.json que corresponde ao token `kid`. Talvez seja necessário converter a JWK em formato PEM primeiro. Esse exemplo a seguir adota o JWT e a JWK, e usa a biblioteca Node.js, jsonwebtoken, para verificar a assinatura do JWT:

   Node.js

```
var jwt = require('jsonwebtoken');
var jwkToPem = require('jwk-to-pem');
var pem = jwkToPem(jwk);
jwt.verify(token, pem, { algorithms: ['RS256'] }, function(err, decodedToken) {
});
```

Verificar as declarações

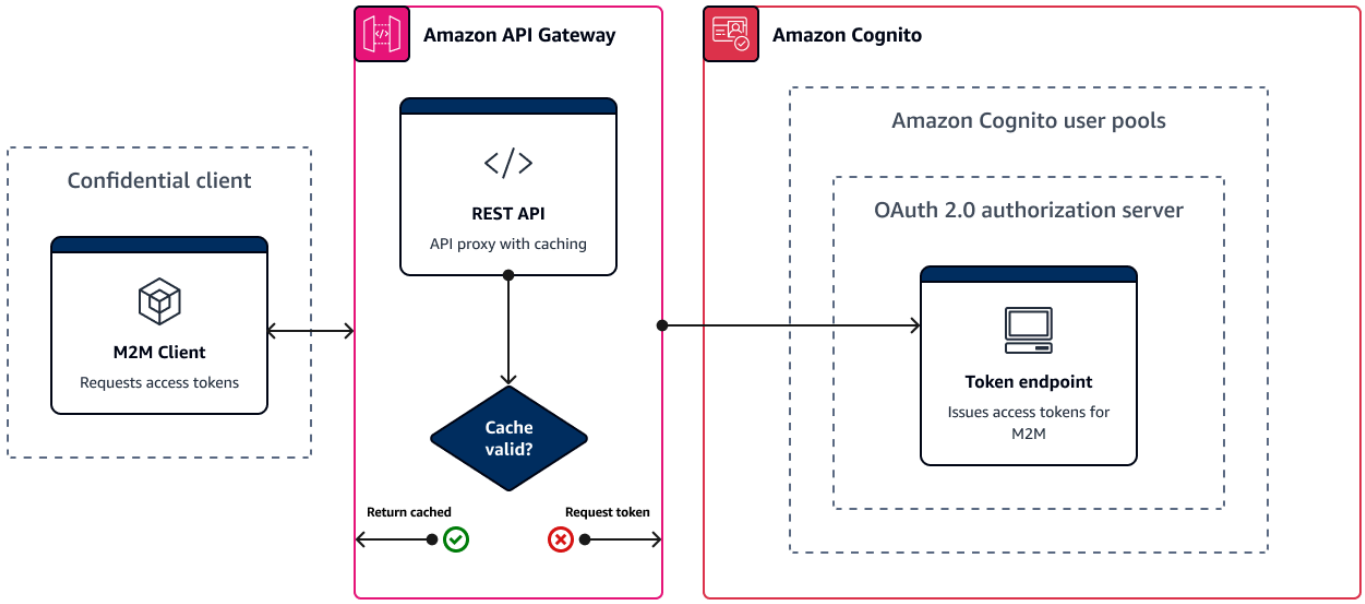
Para verificar alegações JWT

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 do 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.

Armazenar tokens em cache



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 sua UI hospedada.
- 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 até que eles expirem 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 OAuth 2.0 do Amazon Cognito incluem o endpoint de token, que fornece as credenciais de cliente e as solicitações de código de autorização da UI hospedada. 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, a aplicação pode reduzir a escala horizontalmente em resposta a eventos maiores do que a cota de taxa de solicitação padrão dos endpoints OAuth do Amazon 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. As APIs do API Gateway são criptografadas em trânsito e, opcionalmente, em repouso. Um cache do API Gateway é ideal para a concessão de credenciais do cliente OAuth 2.0, um tipo de concessão frequentemente de alto volume que produz tokens de acesso para machine-to-machine autorizar 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 API a fim armazenar um token de acesso separado para cada combinação de escopos do OAuth e cliente da aplicação que você deseja solicitar na aplicação. 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 escopos do OAuth que você solicita no parâmetro de URL `scope` e do cabeçalho `Authorization` na 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 o ElastiCache for Redis. 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.

b. Defina o parâmetro `scope` e o cabeçalho `Authorization` como sua chave de armazenamento em cache.

   i. Adicione uma string de consulta aos URL query string parameters (parâmetros de string de consulta de URL) e selecione Caching (Armazenamento em cache) para a string `scope`.

   ii. Adicione um cabeçalho aos HTTP request headers (Cabeçalhos de solicitação HTTP) e selecione Caching (Armazenamento em cache) para o cabeçalho `Authorization`.

4. Em Stages (Estágios), configure o armazenamento em cache.

   a. Selecione o estágio que deseja modificar.

   b. Em Settings (Configurações), selecione Enable API cache (Habilitar cache da API).

   c. Selecione uma Cache capacity (Capacidade de cache).

   d. Escolha um cache time-to-live (TTL) de pelo menos 3600 segundos.

   e. Desmarque a caixa de seleção Exigir autorização.

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 uma autenticação bem-sucedida do grupo de usuários

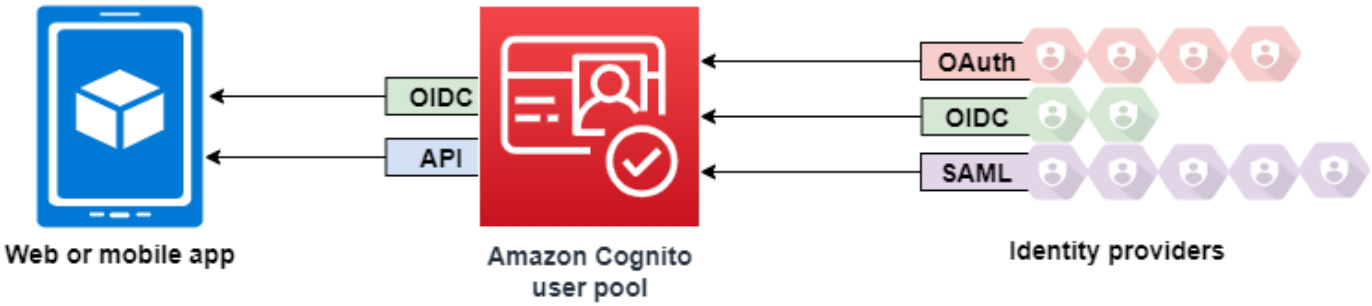
Os usuários do seu aplicativo podem fazer login diretamente por meio de um grupo de usuários ou podem se federar por meio de um provedor de identidade (IdP) terceirizado. O grupo de usuários gerencia a sobrecarga de lidar com os tokens que são retornados do login social por meio do Facebook, Google, Amazon e Apple, e do OpenID Connect (OIDC) e SAML. IdPs Para ter mais informações, consulte Como usar tokens com grupos de usuários.

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.
- Forneça um comprovante de autenticação temporário e revogável.
- Preencha dados de identidade em um perfil de usuário no seu aplicativo.

- Autorize alterações no perfil do usuário conectado no diretório do grupo de usuários.
- Autorize solicitações de informações do usuário com um token de acesso.
- Autorize solicitações de dados que estão por trás de APIs externas protegidas por acesso com tokens de acesso.
- Autorize o acesso aos ativos do aplicativo que estão armazenados no cliente ou no servidor com as Permissões Verificadas da Amazon.

Para obter mais informações, consulte Fluxo de autenticação de grupo de usuários e Como usar tokens com grupos de usuários.



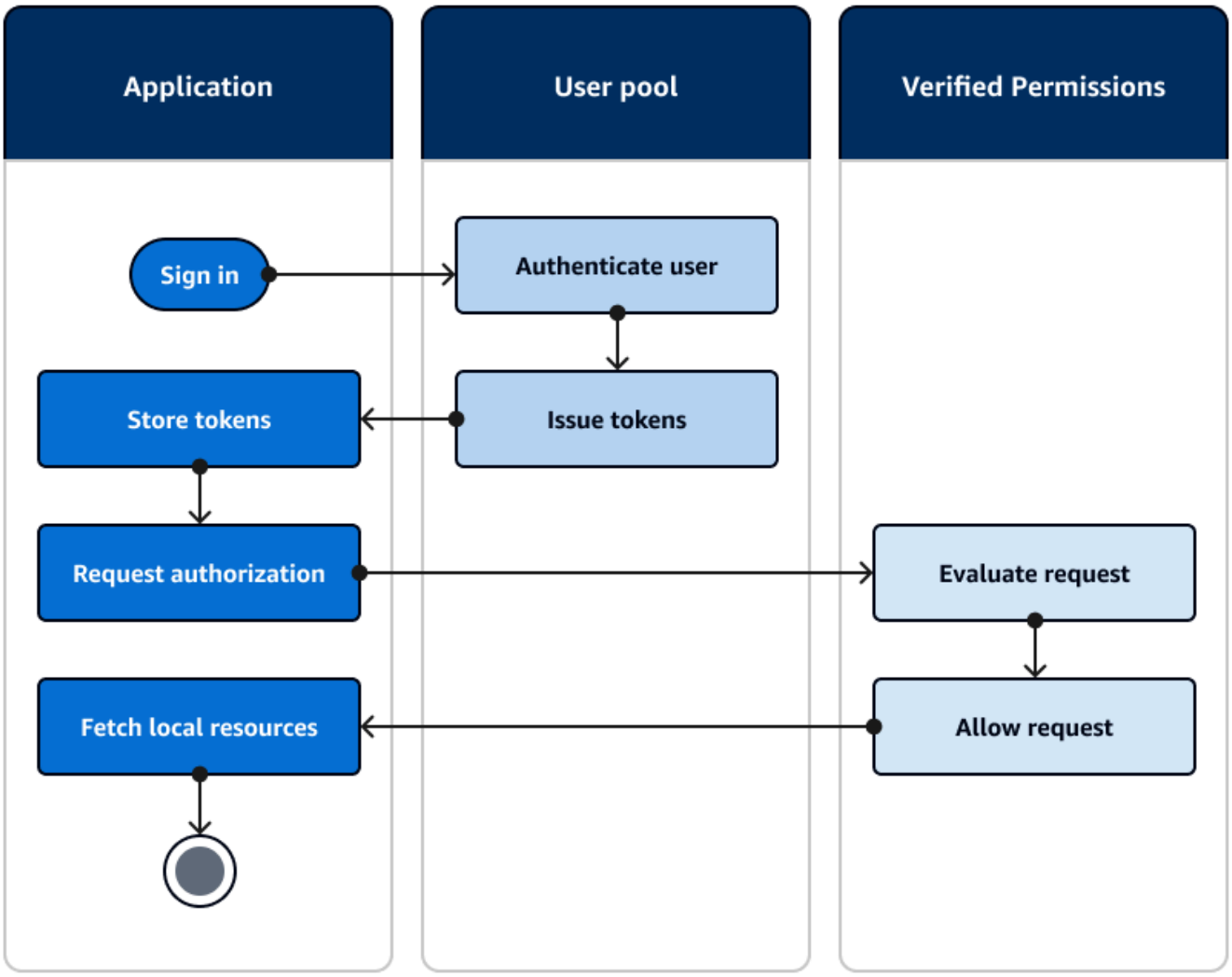
Tópicos

- Autorizando o acesso aos recursos do cliente ou do servidor com as permissões verificadas da Amazon
- Acessando recursos com o API Gateway após o login
- Acessando Serviços da AWS usando um pool de identidades após o login

Autorizando o acesso aos recursos do cliente ou do servidor com as permissões verificadas da Amazon

Seu aplicativo pode passar os tokens de um usuário conectado para as Permissões Verificadas da Amazon. O Verified Permissions é um serviço de autorização e gerenciamento de permissões escalável e refinado para aplicativos personalizados que você criou. Um grupo de usuários do Amazon Cognito pode ser uma fonte de identidade para um armazenamento de políticas de permissões verificadas. As permissões verificadas tomam decisões de autorização para ações e recursos solicitados`premium_badge.png`, como `GetPhoto` for, do principal e seus atributos nos tokens do grupo de usuários.

O diagrama a seguir mostra como seu aplicativo pode passar o token de um usuário para Permissões verificadas em uma solicitação de autorização.



Comece a usar as permissões verificadas da Amazon

Depois de integrar seu grupo de usuários com Permissões verificadas, você obtém uma fonte central de autorização granular para todos os seus aplicativos do Amazon Cognito. Isso elimina a necessidade de uma lógica de segurança refinada que, de outra forma, você teria que codificar e replicar entre todos os seus aplicativos. Para obter mais informações sobre autorização com permissões verificadas, consulte[Autorização com o Amazon Verified Permissions](#).

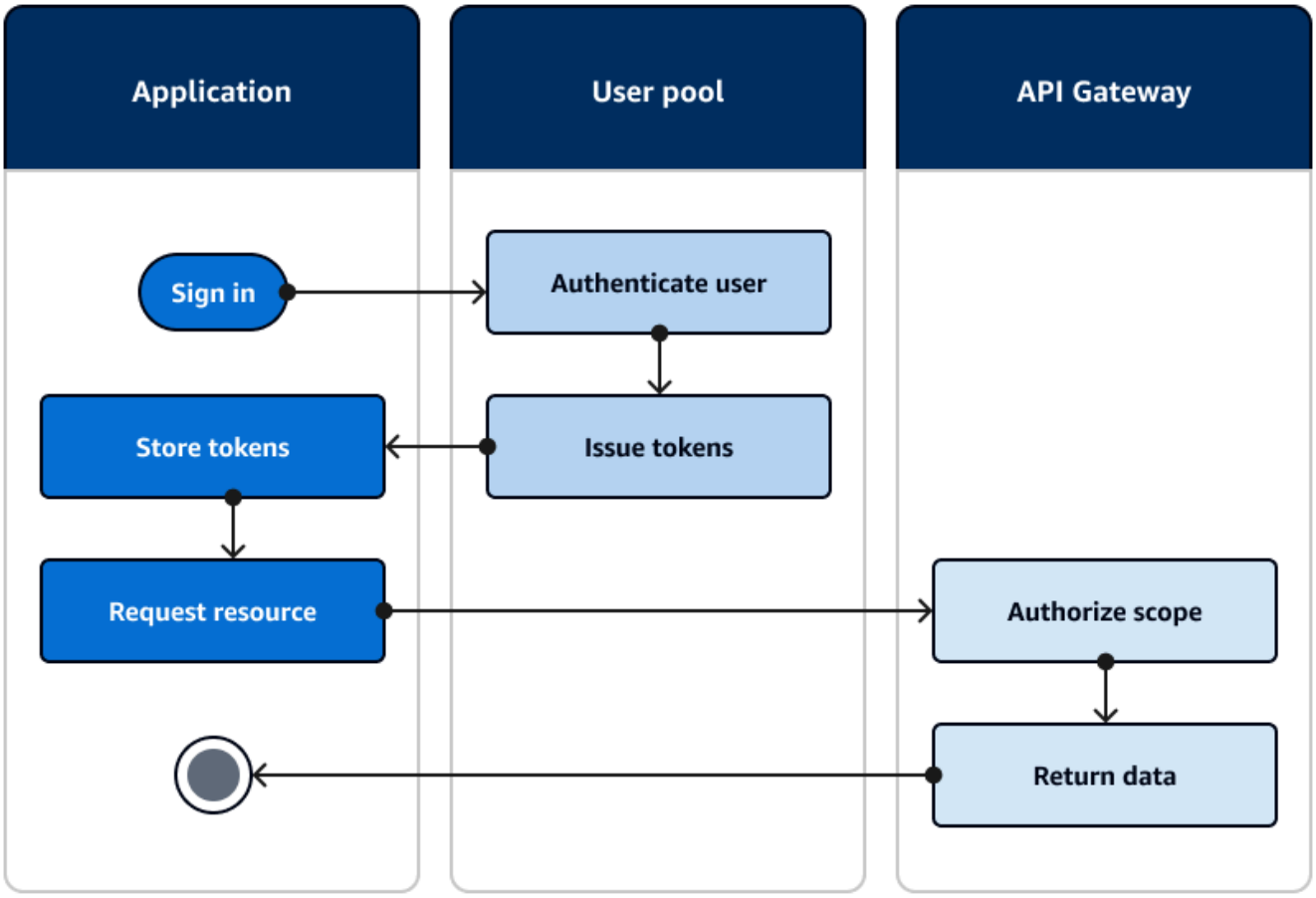
As solicitações de autorização de permissões verificadas exigem AWS credenciais. Você pode implementar algumas das técnicas a seguir para aplicar com segurança as credenciais às solicitações de autorização.

- Opere um aplicativo web que possa armazenar segredos no back-end do servidor.
- Adquira credenciais autenticadas do grupo de identidades.
- Proxy as solicitações do usuário por meio de uma access-token-authorized API e anexe AWS as credenciais à solicitação.

Acessando recursos com o API Gateway após o login

Um uso comum dos tokens de grupos de usuários do Amazon Cognito é autorizar solicitações para uma API REST do API Gateway. Os escopos do OAuth 2.0 nos tokens de acesso podem autorizar um método e um caminho, como for. `HTTP GET /app_assets` Os tokens de ID podem servir como autenticação genérica para uma API e transmitir atributos do usuário para o serviço de back-end. O API Gateway tem opções de autorização personalizadas adicionais, como autorizadores JWT para APIs HTTP e autorizadores 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 do OAuth 2.0 em um token de acesso.



Seu aplicativo deve coletar os tokens das sessões autenticadas e adicioná-los como tokens portadores a um `Authorization` cabeçalho 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 de um aplicativo são:

- O token de acesso contém o escopo correto do OAuth 2.0. 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 sequência de caracteres 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 realizar uma validação adicional do conteúdo do token de ID no seu servidor de aplicativos.
- Um grupo, declaração, atributo ou função em um token de acesso ou ID atende aos requisitos que você define em uma função Lambda. Um autorizador 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 Amazon Verified Permissions.

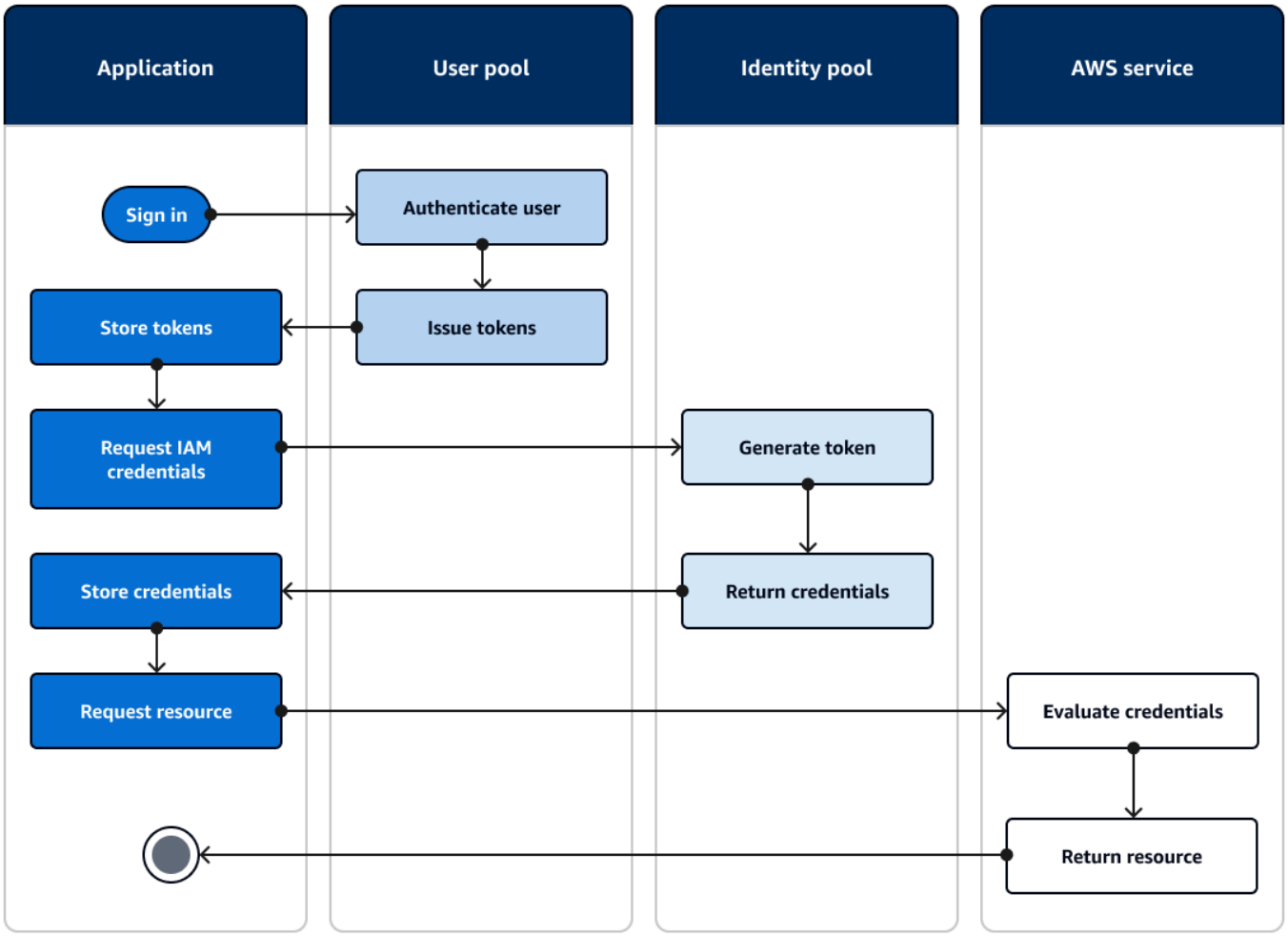
Você também pode autorizar solicitações para uma API AWS AppSync GraphQL com tokens de um grupo de usuários.

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.

Seu aplicativo 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 ter mais informações, consulte Fluxo de autenticação dos grupos de identidades (identidades federadas).

O diagrama a seguir mostra como um aplicativo faz login com um grupo de usuários, recupera as credenciais do grupo de identidades e solicita um ativo de um. AWS service (Serviço da AWS)



Você pode usar as credenciais do grupo de identidades para:

- Faça solicitações de autorização detalhadas para as Permissões Verificadas da Amazon 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.
- Conecte-se a um back-end de banco de dados, como Amazon DynamoDB ou Amazon RDS, que autoriza conexões com o IAM.
- Recupere ativos do aplicativo de um bucket do Amazon S3.
- Inicie uma sessão com um desktop WorkSpaces virtual da Amazon.

Os grupos de identidades não operam exclusivamente em uma sessão autenticada com um grupo de usuários. Eles também aceitam autenticação diretamente de provedores de identidade terceirizados 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, consulteConceitos de grupos de identidades.

Configurando um grupo de usuários com o AWS Management Console

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 AWS Management Console

O procedimento a seguir descreve como usar o AWS Management Console 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 dar aos usuários desse IdP a função padrão que você configurou ao configurar sua função autenticada, ou você pode escolher a função 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 escolher Escolher função com regras, insira a Declaração de origem da autenticação do seu usuário, o Operador que você deseja usar para comparar a declaração com a regra, o Valor que causará uma correspondência com essa opção de função e a Função que você deseja atribuir quando a atribuição de função corresponder. Selecione Adicionar outra para criar uma regra adicional com base em uma condição diferente.

ii. Se você escolher Escolher função com a reivindicação preferred_role em tokens, o Amazon Cognito emitirá credenciais para a função na reivindicação do seu 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 Save Changes (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)
```

Usar atributos de segurança de grupos de usuários do Amazon Cognito

Você pode adicionar a Multi-Factor Authentication (MFA – Autenticação multifator) a um grupo de usuários para proteger a identidade dos usuários. A MFA agrega um segundo fator de autenticação para que o grupo de usuários não dependa exclusivamente do nome do usuário e da senha. Você pode optar por usar mensagens de texto SMS ou senhas de uso único com marcação temporal (TOTPs) como fatores secundários para o login dos usuários. Você também pode usar a autenticação adaptável com o modelo baseado em risco para prever quando pode precisar de outro fator de autenticação. Os recursos de segurança avançada do grupo de usuários incluem autenticação adaptativa e proteções contra credenciais comprometidas.

Tópicos

- Adicionar MFA a um grupo de usuários
- Como adicionar segurança avançada a um grupo de usuários
- Associando uma ACL AWS WAF da web a um grupo de usuários
- Sensibilidade entre maiúsculas e minúsculas do grupo de usuários

- Proteção contra exclusão do grupo de usuários
- Gerenciar respostas de erro de existência do usuário

Adicionar MFA a um grupo de usuários

A autenticação multifator (MFA) aumenta a segurança de sua aplicação. Ela adiciona um fator de autenticação do tipo algo que você tem ao fator algo que você sabe do nome de usuário e senha. Você pode optar por mensagens de texto SMS ou senhas de uso único com marcação temporal (TOTP) como fatores secundários para o login dos usuários.

> **Note**
>
> Na primeira vez em que um novo usuário entra na aplicação, 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 a autenticação de segundo fator em resposta a um aumento no nível de risco. Para adicionar autenticação adaptável ao grupo de usuários, consulte Como adicionar segurança avançada a um grupo de usuários.

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, como SMS ou TOTP. Ao definir a MFA como `required`, você deve incluir a configuração da MFA na integração dos usuários para que seu grupo de usuários permita que eles façam login.

Se você ativar o SMS como fator de MFA, poderá exigir que os usuários forneçam números de telefone e façam a confirmação deles durante o cadastro. Se você tiver definido a MFA como `required` e apenas aceitar SMS como fator, os usuários precisarão fornecer números de telefone. Usuários sem números de telefone precisam de seu suporte para adicionar um número de telefone ao perfil deles para que possam fazer login. Você pode usar números de telefone não verificados para MFA SMS. Esses números receberão status verificado após o êxito da MFA.

Se você definiu a MFA como obrigatória e ativou SMS e TOTP como métodos de verificação compatíveis, o Amazon Cognito solicitará que novos usuários sem números de telefone configurem a MFA TOTP. Se tiver definido a MFA como obrigatória e o único método de MFA que você ativou for TOTP, o Amazon Cognito solicitará que todos os novos usuários configurem a MFA TOTP na segunda vez em que fizerem login. O Amazon Cognito gera um desafio para configurar o TOTP MFA em resposta InitiateAutha operações de API. AdminInitiateAuth

A interface de usuário hospedada solicita que os usuários configurem a MFA quando você a define como obrigatória. Quando você define a MFA como opcional no grupo de usuários, a interface de usuário hospedada 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.

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 Fluxo de autenticação de grupo de usuários.

Tópicos

Pré-requisitos

Antes de configurar a MFA, considere o seguinte:

- Ao ativar a MFA em seu grupo de usuários e escolher SMS text message (Mensagem de texto SMS) como um segundo fator, você pode enviar mensagens SMS para um atributo de número de telefone que você não verificou no Amazon Cognito. Depois que o usuário conclui a MFA SMS, o Amazon Cognito define o atributo `phone_number_verified` como `true`.
- 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 ter mais informações, consulte Configurações de mensagens SMS para grupos de usuários do Amazon Cognito.

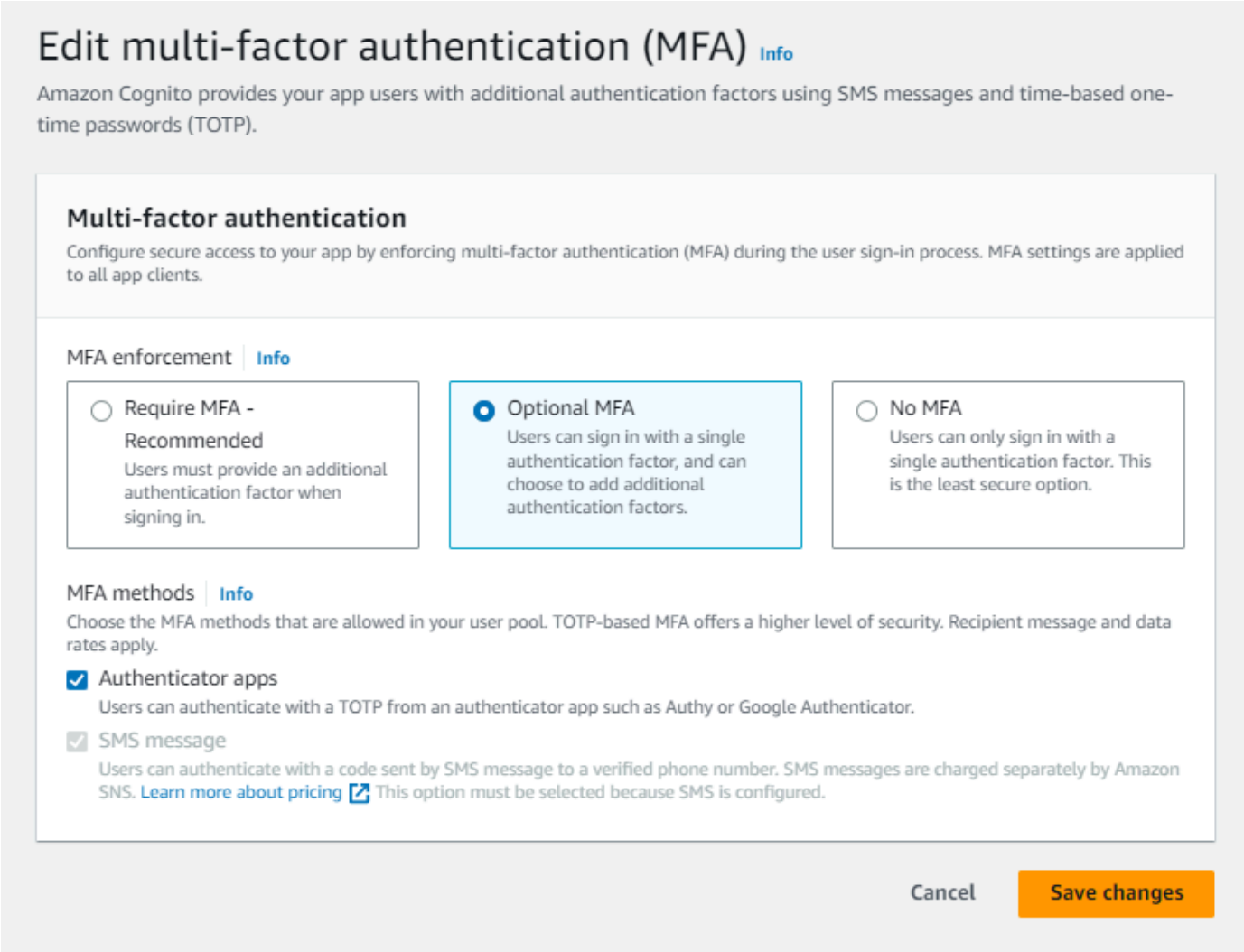
- Os recursos de segurança avançada exigem que você ative a MFA e a defina como opcional no console do grupo de usuários do Amazon Cognito. Para ter mais informações, consulte Como adicionar segurança avançada a um grupo de usuários.

Como configurar autenticação multifator

É possível configurar MFA no console do Amazon Cognito.

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. Escolha a guia Sign-in experience (Experiência de acesso). Encontre Multi-factor authentication (Autenticação multifator) e escolha Edit (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 SMS adicional ou um fator de senha de uso único com marcação temporal (TOTP).

b. Optional MFA (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 Como adicionar segurança avançada a um grupo de usuários.

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 SMS message (Mensagem SMS) ou Authenticator apps (Aplicações autenticadoras) geradoras de TOTP como segundo fator. Recomendamos que você implemente a MFA baseada em TOTP para que a recuperação da conta possa usar mensagens SMS.

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. Na guia Messaging (Sistema de mensagens) para o grupo de usuários, localize SMS e escolha Edit (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.
8. Escolha Salvar alterações.

MFA de mensagem de texto SMS

Quando um usuário faz login com a MFA habilitada, ele primeiramente insere e envia o nome de usuário e senha. O aplicativo cliente recebe uma resposta `getMFA` que indica onde o código de autorização foi enviado. A aplicação cliente indicará ao usuário onde procurar o código (por exemplo, para qual número de telefone o código foi enviado). Em seguida, ela fornece um formulário para inserir o código. Por fim, a aplicação cliente envia o código para concluir o processo de login. O destino é mascarado, o que esconde todos os dígitos do número de telefone, exceto os quatro últimos. Se uma aplicação estiver usando a interface do usuário hospedada do Amazon Cognito, ela mostrará uma página para que o usuário insira o código de MFA.

O código de autorização da mensagem de texto SMS é válido para a Authentication flow session duration (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 App integration (Integração de aplicações), quando você modifica o cliente da aplicação em App clients and analytics (Clientes e análise de aplicações). Você também pode definir a duração da sessão do fluxo de autenticação em uma solicitação de API `CreateUserPoolClient` ou `UpdateUserPoolClient`. Para ter mais informações, consulte Fluxo de autenticação de grupo de usuários.

Se um usuário não tiver mais acesso ao dispositivo no qual os códigos de MFA de mensagem de texto SMS são enviados, ele deve solicitar ajuda ao atendimento ao cliente. Um administrador com Conta da AWS as permissões necessárias pode alterar o número de telefone do usuário, mas somente por meio da API AWS CLI ou da API.

Quando um usuário percorre com êxito o fluxo de MFA de uma mensagem de texto SMS, seu número de telefone também é marcado como verificado.

> **Note**
>
> O SMS para MFA é cobrado separadamente. (Não há encargos para envio de códigos de verificação a endereços de e-mail.) Para obter informações sobre preços do Amazon SNS, consulte Preço mundial de SMS. Para obter a lista atualizada de países nos quais as mensagens SMS estão disponíveis, consulte Países compatíveis.

> **Important**
>
> Para garantir que serão enviadas mensagens SMS para verificar números de telefone e MFA de mensagem de texto SMS, você deverá solicitar um aumento do limite de gastos no Amazon SNS.
> O Amazon Cognito usa o Amazon SNS para enviar mensagens SMS aos usuários. O número de mensagens SMS que o Amazon SNS envia está sujeito a limites de gastos.
> Os limites de gastos podem ser especificados para uma AWS conta e para mensagens individuais, e os limites se aplicam somente ao custo do envio de mensagens SMS.
> O limite de gastos padrão por conta, caso não seja especificado, é de 1.00 USD por mês.
> Se você quiser aumentar o limite, envie um caso de aumento de limite do SNS no AWS Support Centro. Em New limit value (Novo valor limite), informe o limite de gastos mensal desejado. No campo Use Case Description (Descrição do caso de uso), explique que você está solicitando um aumento de limite de gastos mensal de SMS.

Para adicionar MFA ao grupo de usuários, consulte Adicionar MFA a um grupo de usuários. Para obter mais informações sobre mensagens SMS com o Amazon SNS em seu grupo de usuários, consulte. Configurações de mensagens SMS para grupos de usuários do Amazon Cognito

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 sua aplicação usar a interface do usuário hospedada 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 na UI hospedada, 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. Para obter mais informações sobre a experiência de cadastro na UI hospedada, consulte Como se cadastrar em uma nova conta na UI hospedada do Amazon Cognito.

> **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 da interface hospedada. Para criar uma regra que tenha uma ação de CAPTCHA e não afete a TOTP da UI hospedada, consulte Configurando sua ACL AWS WAF da web para UI hospedada TOTP MFA. Para obter mais informações sobre ACLs AWS WAF da web e o Amazon Cognito, consulte. Associando uma ACL AWS WAF da web a um grupo de usuários

Para implementar a MFA com TOTP em uma UI personalizada em que você usa a API do Amazon Cognito, consulte Configurar a MFA para um usuário na API de grupos de usuários do Amazon Cognito.

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 por meio da interface do usuário hospedada 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 da interface do usuário hospedada 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 a interface do usuário hospedada 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 de aplicações autenticadoras que geram códigos com a função hash SHA-1. Os códigos gerados com o hash SHA-256 geram um erro `Code mismatch`.

Configurar a MFA para um usuário na API de grupos de usuários do Amazon Cognito

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 a interface do usuário hospedada apresenta a TOTP para o Amazon Cognito nas respostas do desafio de MFA.

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. O usuário precisa inserir a chave em uma aplicação geradora de TOTP, como o Google Authenticator. Você pode usar libqrencode para gerar um código QR.
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, chame SetUserMFAPreference para modificar seu usuário para 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 MFAPreference. SetUser
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 UI hospedada TOTP MFA

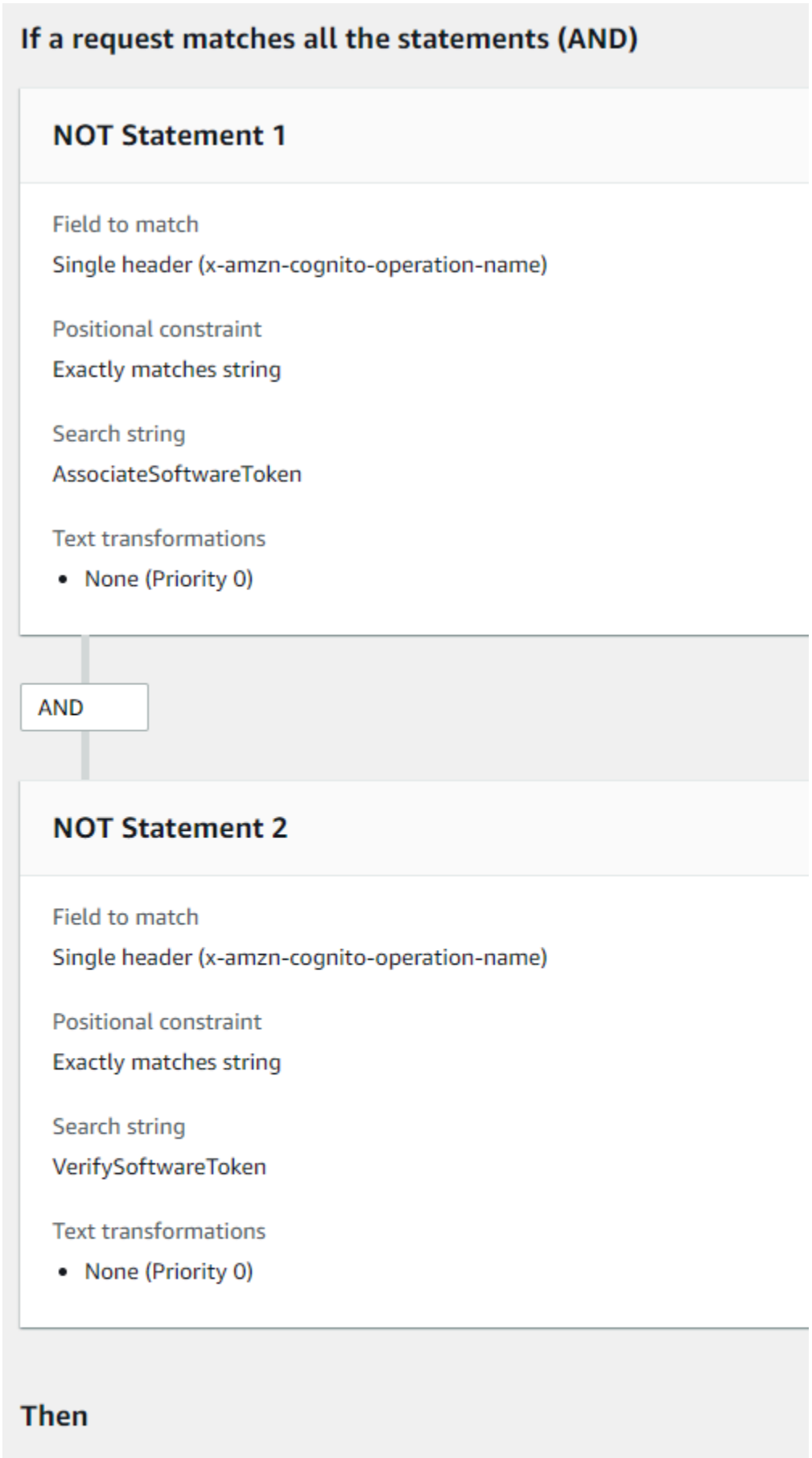
Quando você tem uma ACL AWS WAF da web associada a um grupo de usuários e uma regra na sua ACL da web apresenta um CAPTCHA, isso pode causar um erro irrecuperável no registro do TOTP da interface hospedada. AWS WAF Dessa forma, as regras de CAPTCHA afetam apenas o TOTP MFA na interface do usuário hospedada. 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 a TOTP da UI hospedada, 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 ACLs AWS WAF da web e o Amazon Cognito, consulte. Associando uma ACL AWS WAF da web a um grupo de usuários

Como adicionar segurança avançada a um grupo de usuários

Depois de criar o grupo de usuários, você terá acesso à Advanced security (Segurança avançada) na barra de navegação do console do Amazon Cognito. Você pode ativar os recursos de segurança avançada do grupo de usuários 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, os recursos avançados de segurança publicam métricas na Amazon CloudWatch. Você pode ver métricas de segurança avançadas depois que o Amazon Cognito gerar seu primeiro evento de segurança avançada. Consulte Como exibir métricas de segurança avançada.

Os recursos de segurança avançada incluem detecção de credenciais comprometidas e autenticação adaptativa.

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 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.

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.
>
> Quando você ativa a segurança avançada, 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.

Personalização do token de acesso

Ao ativar atributos avançados de segurança, é possível configurar o grupo de usuários para aceitar respostas a um evento de gatilho do Lambda versão 2. Com a versão 2, é possível personalizar escopos e outras declarações em tokens de acesso. Isso aumenta a capacidade de criar resultados de autorização flexíveis quando os usuários se autenticam. Para ter mais informações, consulte Personalizar o token de acesso.

Tópicos

- Considerações e limitações
- Pré-requisitos
- Como configurar recursos de segurança avançada
- Como verificar credenciais comprometidas
- Como usar a autenticação adaptável
- Como exibir métricas de segurança avançada
- Ativar a segurança avançada do grupo de usuários em sua aplicação

Considerações e limitações

- O preço adicional se aplica aos recursos de segurança avançada do Amazon Cognito. Consulte a página de preço do Amazon Cognito.
- O Amazon Cognito oferece suporte à autenticação adaptativa e à detecção de credenciais comprometidas com os seguintes fluxos de autenticação padrão:, e. `USER_PASSWORD_AUTH` `ADMIN_USER_PASSWORD_AUTH` `USER_SRP_AUTH` Não é possível utilizar segurança avançada com um fluxo `CUSTOM_AUTH` eAcionadores do Lambda de desafio personalizado de autenticação ou com login federado.
- Com os recursos de segurança avançada 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.
- 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. Os recursos de segurança avançada do Amazon Cognito não impedem ataques de negação distribuída de serviço (DDoS). Para implementar defesas contra ataques volumétricos em seus grupos de usuários, adicione AWS WAF ACLs da web. Para ter mais informações, consulte Associando uma ACL AWS WAF da web a um grupo de usuários.

- As concessões de credenciais do cliente são destinadas à autorização machine-to-machine (M2M) sem conexão com contas de usuário. Os recursos avançados de segurança monitoram 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 ter mais informações, consulte Associando uma ACL AWS WAF da web a um grupo de usuários.

Pré-requisitos

Antes de começar, você precisará fazer o seguinte:

- Um grupo de usuários com um cliente de aplicativo. Para ter 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 ter 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.

Como configurar recursos de segurança avançada

É possível configurar os recursos de segurança avançada do Amazon Cognito no AWS Management Console.

Para configurar segurança avançada 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. Escolha a guia App integration (Integração da aplicação). Localize Advanced security (Segurança avançada) e escolha Enable (Habilitar). Se tiver habilitado a segurança avançada anteriormente, escolha Edit (Editar).
5. Selecione Full function (Função completa) para configurar respostas de segurança avançada para credenciais comprometidas e autenticação adaptativa. Selecione Somente auditoria para coletar informações e enviar dados do grupo de usuários para CloudWatch o. O preço da segurança avançada se aplica a ambos os modos de Audit only (Somente auditoria) e de Full function (Função completa). Para mais informações, consulte Preços do Amazon Cognito.

   Recomendamos manter os recursos de segurança avançada 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.
6. 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 Custom (Personalizadas) para Compromised credentials (Credenciais comprometidas), escolha as ações do grupo de usuários que o Amazon Cognito usará para Event detection (Detecção de eventos) e as Compromised credentials responses (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.

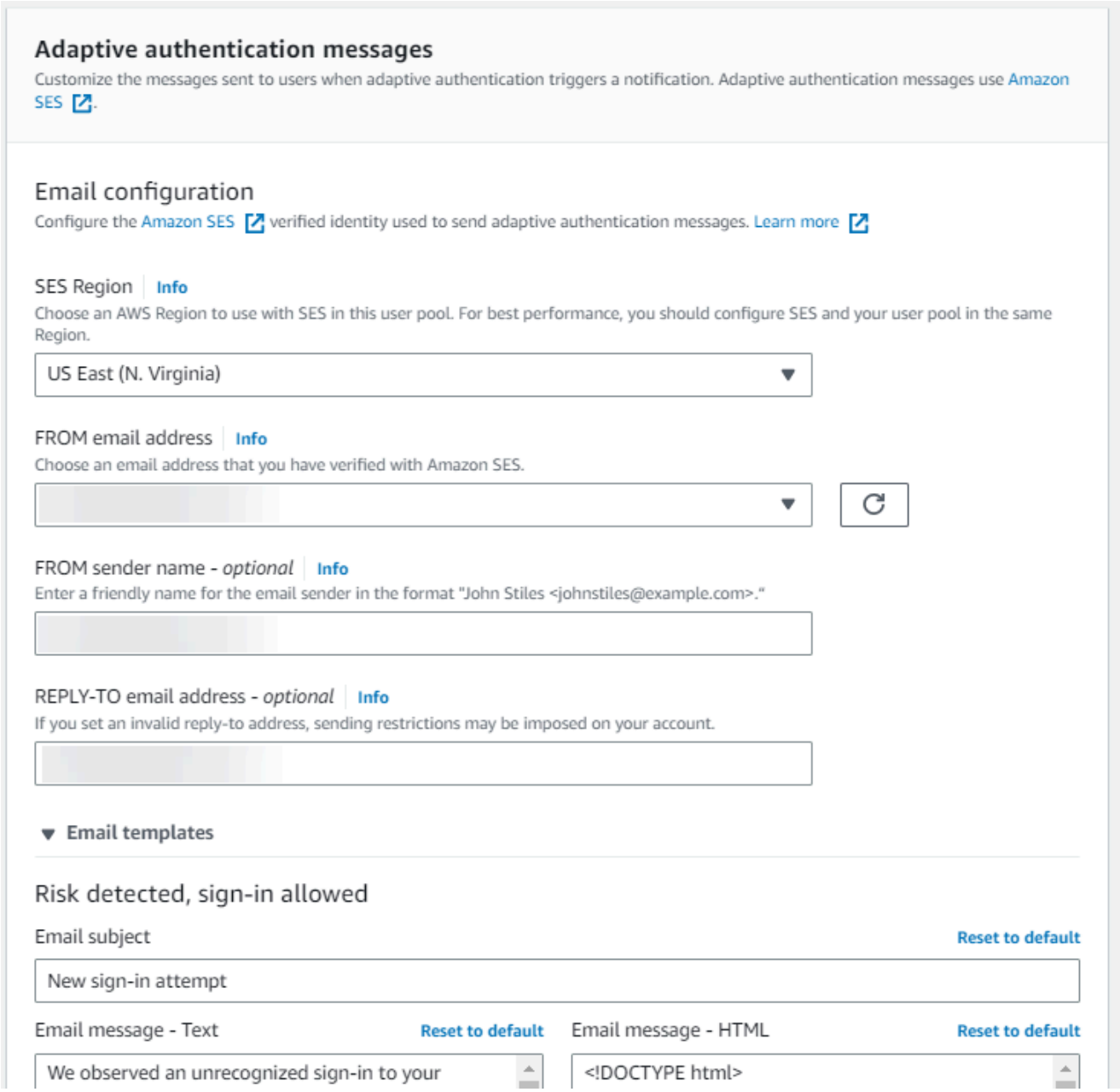
iii. Require MFA (Exigir MFA): se o usuário tiver a MFA configurada, o Amazon Cognito sempre vai exigir que o usuário forneça um fator adicional de SMS ou TOTP quando fizer o acesso. Se o usuário não tiver a MFA configurada, o Amazon Cognito solicitará que ele configure a MFA. Antes de exigir automaticamente 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.

7. 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 Email configuration (Configuração de e-mail), escolha os valores para SES Region (Região SES), FROM email address (Endereço de e-mail do remetente), FROM sender name (Nome do remetente) e REPLY-TO email address (Endereço de e-mail para a resposta) 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.

8. Expanda IP address exceptions (Exceções de endereço IP) para criar uma lista Always-allow (Permitir sempre) ou um Always-block (Bloquear sempre) de intervalos de endereços IPv4 ou IPv6 que sempre serão permitidos ou bloqueados, independentemente da avaliação de risco de segurança avançada. Especifique os intervalos de endereços IP em CIDR notation (Notação CIDR) (por exemplo, 192.168.100.0/24).

9. Escolha Salvar alterações.

Como verificar 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, na interface do usuário hospedada e com a API do Amazon Cognito. Um usuário local existe exclusivamente em seu diretório de grupo de usuários sem federação por meio de um IdP externo.

Em Advanced security (Segurança avançada), na guia App integration (Integração de aplicações) do console do Amazon Cognito, é possível configurar Compromised credentials (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 ter mais informações, consulte Como exibir métricas de segurança avançada. 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.

> **Note**
>
> 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.
> O Amazon Cognito verifica se há credenciais comprometidas em logins que usam a AdminInitiateAuthAPI com `ADMIN_USER_PASSWORD_AUTH` fluxo e a InitiateAuthAPI com `USER_PASSWORD_AUTH` fluxo.

Para adicionar proteções contra credenciais comprometidas ao grupo de usuários, consulte Como adicionar segurança avançada a um grupo de usuários.

Como usar a 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 que incluem informações do dispositivo e do usuário. 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 tentativas de login, seus níveis de risco e desafios fracassados para a Amazon. CloudWatch Para ter mais informações, consulte Como exibir métricas de segurança avançada.

Para adicionar autenticação adaptável ao grupo de usuários, consulte Como adicionar segurança avançada a um grupo de usuários.

Tópicos

- Visão geral da autenticação adaptável
- Adicionar dados de sessão e dispositivo do usuário a solicitações de API
- Como exibir o histórico de eventos do usuário
- Como fornecer feedback sobre eventos
- Como enviar mensagens de notificação

Visão geral da autenticação adaptável

Na página Segurança avançada na guia Integração da aplicação do console do Amazon Cognito, é possível selecionar as configurações de autenticação adaptativa, inclusive 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 segurança avançada global a todos os seus clientes de aplicações, mas aplicar uma configuração em 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 segurança avançada 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 segurança avançada do Amazon Cognito com ContextDataos parâmetros e. UserContextData A biblioteca de coleta de dados de contexto está incluída nos AWS SDKs. Para obter mais informações, consulte Como integrar o Amazon Cognito a aplicações Web e móveis. Você poderá enviar `ContextData` se tiver ativado recursos de segurança avançada em seu grupo de usuários. Para obter mais informações, consulte Configurar recursos de segurança avançada.

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` aos recursos de segurança avançada 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:

- Você ativou recursos de segurança avançada em seu grupo de usuários. Para obter mais informações, consulte Configurar recursos de segurança avançada.
- O cliente da aplicação tem um segredo do cliente. Para obter mais informações, consulte Configurar um cliente da aplicação do grupo de usuários.
- Você ativou a opção Accept additional user context data (Aceitar dados de contexto do usuário adicionais) no cliente da aplicação. Para ter mais informações, consulte Aceitar dados de contexto do usuário adicionais (AWS Management Console).

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 (AWS Management Console)

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, aos recursos de segurança avançada 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. Escolha a guia Integração do aplicativo ().
4. Em App clients and analytics (Clientes da aplicação e análise), 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 ativar a segurança avançada na aplicação web ou no aplicativo móvel, consulte Activating user pool advanced security from your app (Ativar a segurança avançada do grupo de usuários de sua aplicação). 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 SDKs do Amazon Cognito, consulte Instalar um SDK do grupo de usuários. A segurança avançada 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 parâmetro `IPaddress` é um endereço IPv4 ou IPv6 válido.

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 o histórico de eventos do usuário

> **Note**
>
> No console novo do Amazon Cognito, você pode visualizar o histórico de eventos do usuário na guia Users (Usuários).

Para ver o histórico de logins de um usuário, é possível selecionar o usuário em Users (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 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.

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 ter mais informações, consulte Como usar tokens com grupos de usuários.

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 Users (Usuários) e selecione Provide event feedback (Fornecer feedback sobre o 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 na guia Users and groups(Usuários e grupos). 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.

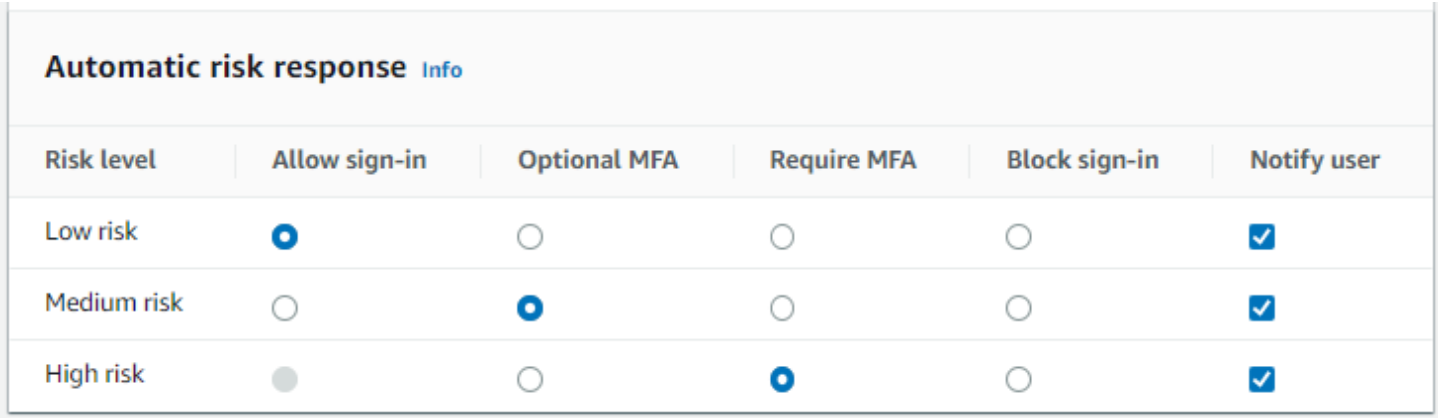
Ao selecionar Set as valid (Definir como válido) no console do Amazon Cognito ou fornecer um valor `FeedbackValue` de `valid` na API, você diz ao Amazon Cognito que confia em uma sessão de usuário em que o Amazon Cognito avaliou algum nível de risco. Ao selecionar Set as invalid (Definir como inválido) 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 as proteções de segurança avançada, 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.

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 Email templates (Modelos de e-mail) em Adaptive authentication messages (Mensagens de autenticação adaptável) em sua configuração de segurança avançada. Para saber mais sobre modelos de e-mail, consulte Modelos de mensagens.

Como exibir métricas de segurança avançada

O Amazon Cognito publica métricas para recursos avançados de segurança em sua conta na Amazon. CloudWatch O Amazon Cognito agrupa as métricas de segurança avançada com base nos níveis de risco e no nível de solicitação.

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	Descrição	Dimensões da métrica
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.
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. 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.

Métrica	Descrição	Dimensões da métrica
		UserPoolId: o identificador do grupo de usuários. RiskLevel: alto, médio ou baixo.
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.
NoRisk	Solicitações em que o Amazon Cognito não identific ou qualquer risco.	Operation: o tipo de operação, como `PasswordChange` , `SignIn` ou `SignUp`. UserPoolId: o identificador do grupo de usuários.

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	Descrição
Por classificação de risco	Solicitações que o Amazon Cognito identifica como arriscadas.
Por classificação de solicitação	Métricas agregadas por solicitação.

Ativar a segurança avançada do grupo de usuários em sua aplicação

Depois de configurar os recursos de segurança avançada para o grupo de usuários, você precisará ativá-los na aplicação Web ou no aplicativo móvel.

Usando segurança avançada com JavaScript

1. Adicione o SDK de identidade do Amazon Cognito JavaScript ao seu aplicativo.
2. Em CognitoUserPool.js, `AdvancedSecurityDataCollectionFlag` defina `true` como. Defina `UserPoolId` como o ID do grupo de usuários.
3. Adicione essa referência de origem ao JavaScript arquivo do seu aplicativo. `<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>
```

Usar segurança avançada com Android

1. Crie seu aplicativo com AWS Amplify para Android. Para ter mais informações, consulte Configuração do projeto no AWS Amplify Dev Center.
2. Com `userContextDataProvider`, inclua informações do usuário e do dispositivo em suas solicitações de autenticação.

Para ter informações sobre como adicionar dados de contexto do usuário no SDK antigo do Android, consulte aws-android-sdk-cognitoidentityprovider-asf.

Usar segurança avançada com iOS

1. Crie seu aplicativo com AWS Amplify Swift ou Flutter. Para ter mais informações, consulte Configuração do projeto Swift e Configuração do projeto Flutter no AWS Amplify Dev Center.
2. Inclua informações do usuário e do dispositivo em suas solicitações de autenticação. Para ver um exemplo para usar com a operação de InitiateAuthAPI, consulte `userContextData` em InitiateAuthInput+Amplify.swift on. GitHub

Para ter informações sobre como adicionar dados de contexto do usuário no SDK herdado do iOS, consulte AWSCognitoIdentityProviderASF.

Associando 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 à sua

interface de usuário hospedada e aos endpoints do 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 ACLs AWS WAF da web, consulte Gerenciando e usando uma lista de controle de acesso à web (ACL da web) 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 ACLs AWS WAF da web e o Amazon Cognito

- 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 ACL da web a um 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. Você implementa o recurso ATP ao adicionar o grupo de regras gerenciadas pela `AWS-AWSManagedRulesATPRuleSet`. Antes de associá-lo a um grupo de usuários, a ACL da web não pode usar 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 da interface hospedada. Para criar uma regra que tenha uma ação de CAPTCHA e não afete a TOTP da UI hospedada, consulte Configurando sua ACL AWS WAF da web para UI hospedada TOTP MFA.

AWS WAF inspeciona as solicitações para os seguintes endpoints.

Interface do usuário hospedada

Solicitações a todos os endpoints no Referência da interface do usuário hospedada e endpoints de federação 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 ter mais informações, consulte Operações de API autenticadas e não autenticadas de grupos de usuários do Amazon Cognito.

É possível configurar as regras na ACL da web com ações como Count (Contar), Allow (Permitir), Block (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 da interface do usuário hospedada. Você só pode apresentar um CAPTCHA para o usuário resolver na interface do usuário hospedada.
> - 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 os 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

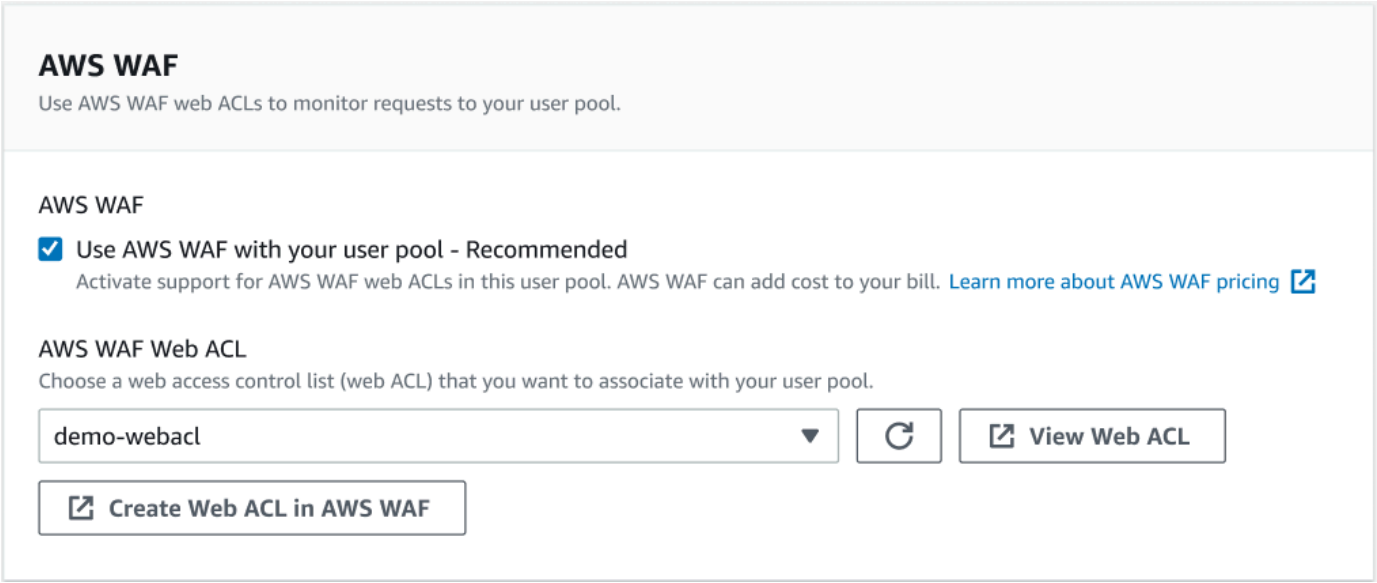
Para trabalhar com uma ACL da web em seu grupo de usuários, seu diretor AWS Identity and Access Management (IAM) deve ter as seguintes permissões do Amazon Cognito. Para obter informações sobre AWS WAF permissões, consulte Permissões de AWS WAF API no Guia do AWS WAF desenvolvedor.

- `cognito-idp:AssociateWebACL`
- `cognito-idp:DisassociateWebACL`
- `cognito-idp:GetWebACLForResource`
- `cognito-idp:ListResourcesForWebACL`

Embora você deva conceder permissões do IAM, as ações listadas são somente com permissão e não correspondem a uma 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. Escolha a guia User pool properties (Propriedades do grupo de usuários).
4. Escolha Edit (Editar) ao lado do AWS WAF.
5. Em AWS WAF, 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. AWS Management Console
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 ACLs da web

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 obter mais informações, consulte Teste e ajuste de suas AWS WAF proteções no Guia do AWS WAF desenvolvedor.

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 da interface do usuário hospedada incluem somente o cabeçalho do `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 as ACLs da web não estão sujeitas aos preços dos recursos avançados de segurança do Amazon Cognito. Os recursos de segurança do AWS WAF complementam os recursos avançados de segurança do Amazon Cognito. Você pode ativar os dois recursos em um grupo de usuários. AWS WAF cobra separadamente pela inspeção das solicitações do grupo de usuários. Para obter mais informações, consulte Preços do AWS WAF.

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 AWS Management Console 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.

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, consulteUsando 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`. Esse comportamento desse padrão é oposto ao do padrão dos grupos de usuários que você cria no AWS Management Console. Antes de 12 de fevereiro de 2020, grupos de usuários tinham distinção entre maiúsculas e minúsculas por padrão, independentemente da plataforma.

Você pode usar a guia Experiência de login AWS Management Console ou a operação da DescribeUserPoolAPI para revisar as configurações de distinção entre maiúsculas e minúsculas de cada grupo de usuários em sua conta.

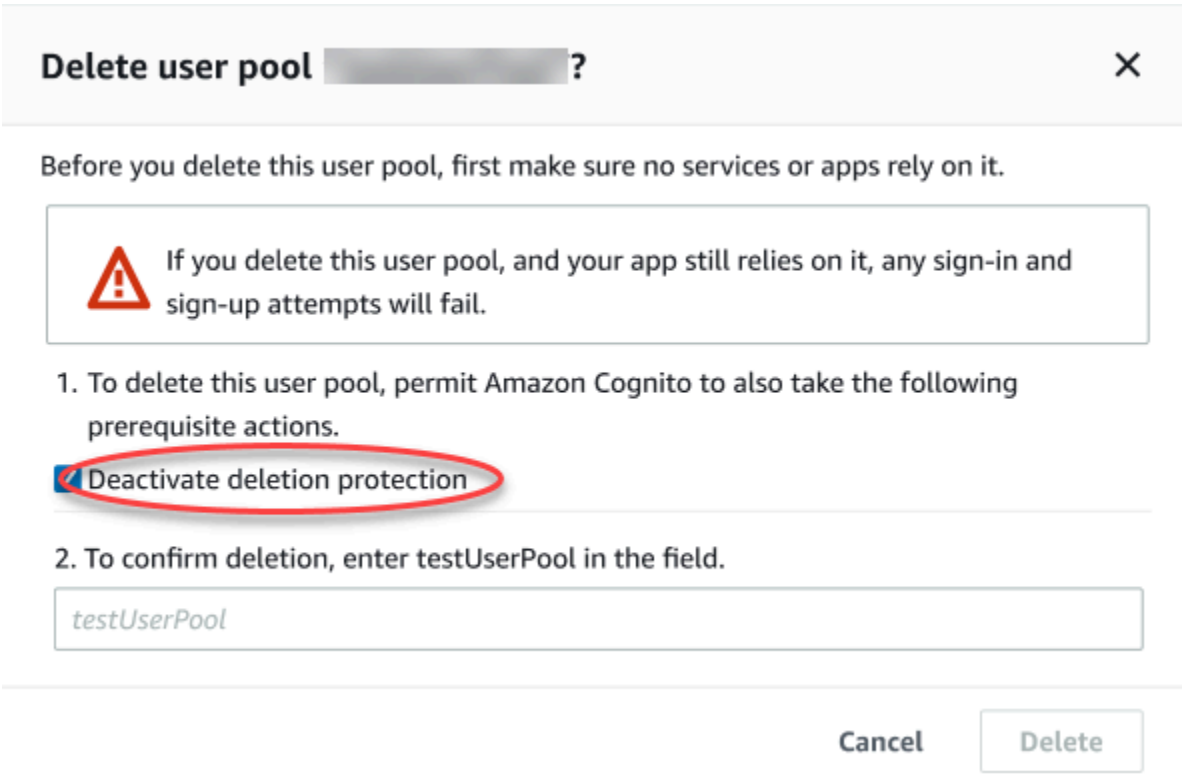
Migração para um novo grupo de usuários

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 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 AWS Management Console, você pode desativar, ao mesmo tempo, a proteção contra exclusão. 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 de API do Amazon Cognito, você deve primeiro mudar `DeletionProtection` para `Inactive` em uma solicitação UpdateUserPool. 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 solicitação DeleteUserPool.

O Amazon Cognito ativa a Deletion protection (Proteção contra exclusão) por padrão quando você cria um grupo de usuários no AWS Management Console. Quando você cria um grupo de usuários com a API `CreateUserPool`, a proteção contra exclusão fica inativa por padrão. Para usar esse recurso em grupos de usuários que você cria com a AWS CLI ou um AWS SDK, defina o parâmetro `DeletionProtection` como `True`.

É possível ativar ou desativar o status da proteção contra exclusão no contêiner Deletion protection (Proteção contra exclusão) na guia User pool settings (Configurações do grupo de usuários) no console do Amazon Cognito.

Como configurar a proteção contra exclusão

1. Acesse o console do Amazon Cognito. Podem ser solicitadas 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. Selecione a guia User pool settings (Configurações do grupo de usuários). Localize Deletion Protection (Proteção contra exclusão) e selecione Activate (Ativar) ou Deactivate (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. Quando você cria um novo aplicativo, o cliente com a API de grupos de usuários do Amazon Cognito `PreventUserExistenceErrors` é`LEGACY`, ou desativado, por padrão. No console do Amazon Cognito, a opção Evitar erros na existência do usuário — uma configuração de `ENABLED` para `PreventUserExistenceErrors` — é selecionada por padrão. Para atualizar sua `PreventUserExistenceErrors` configuração, faça o seguinte:

- Altere o valor de `PreventUserExistenceErrors` between `ENABLED` e `LEGACY` em uma solicitação de UpdateUserPoolClientAPI.
- Edite seu cliente de aplicativo no console do Amazon Cognito e altere o estado de Evitar erros de existência de usuário entre selecionado (`ENABLED`) e desmarcado (). `LEGACY`

Quando essa propriedade tem um valor de`LEGACY`, seu cliente do aplicativo retorna uma resposta de `UserNotFoundException` erro 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 de`ENABLED`, seu cliente do aplicativo não divulga a inexistência de uma conta de usuário em seu grupo de usuários com um `UserNotFoundException` erro. Uma `PreventUserExistenceErrors` configuração de `ENABLED` tem os seguintes efeitos:

- O Amazon Cognito responde com informações não específicas às solicitações de API em que, de outra forma, sua resposta poderia revelar a existência de um usuário válido.

- As APIs de login e esquecimento de senha do Amazon Cognito retornam uma resposta genérica de falha de autenticação. A resposta de erro informa que o nome de usuário ou a senha está incorreta.
- As APIs de confirmação de conta e recuperação de senha do Amazon Cognito retornam uma resposta indicando que um código foi enviado para um meio de entrega simulado, em vez de uma representação parcial das informações de contato do usuário.

As informações a seguir detalham os comportamentos das operações do grupo de usuários quando `PreventUserExistenceErrors` está definido como`ENABLED`.

Operações de autenticação e criação de usuários

Você pode configurar respostas de erro na autenticação de nome de usuário e senha remota segura (SRP). Também é possível personalizar os erros retornados com a autenticação personalizada. As seguintes APIs realizam essas operações de autenticação:

- `AdminInitiateAuth`
- `AdminRespondToAuthChallenge`
- `InitiateAuth`
- `RespondToAuthChallenge`

A lista a seguir demonstra como você pode personalizar as respostas de erro nas operações de autenticação do usuário.

Autenticação com nome de usuário e senha

Para fazer login de um usuário com `ADMIN_USER_PASSWORD_AUTH` e `USER_PASSWORD_AUTH`, inclua o nome de usuário e a senha em uma solicitação de API `AdminInitiateAuth` ou `InitiateAuth`. O Amazon Cognito retorna um erro genérico `NotAuthorizedException` quando o nome de usuário ou a senha está incorreta.

Autenticação baseada em senha remota segura (SRP)

Para fazer login de um usuário com `USER_SRP_AUTH`, inclua o nome de usuário e um parâmetro `SRP_A` em uma solicitação de API `AdminInitiateAuth` ou `InitiateAuth`. Em resposta, o Amazon Cognito devolve um `SRP_B` sal para o usuário. Quando um usuário não é localizado, 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

identificador universal exclusivo (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.

> **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

Se você usar o Acionador do Lambda do desafio de autenticação personalizada e habilitar respostas de erro, o `LambdaChallenge` retornará um parâmetro booliano chamado `UserNotFound`. Em seguida, ele é passado na solicitação de acionadores do Lambda `DefineAuthChallenge`, `VerifyAuthChallenge` e `CreateAuthChallenge`. Você pode usar esse acionador para simular desafios de autenticação personalizados para usuários não existentes. Se você chamar o acionador do Lambda de pré-autenticação para um usuário que não existe, o Amazon Cognito retornará `UserNotFound`.

A lista a seguir demonstra como você pode personalizar as respostas de erro nas operações de criação de usuários.

SignUp

A `SignUp` operação 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 Solicitar que os usuários importados redefinam suas senhas 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 Migrar o acionador do Lambda do usuário.

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`.

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 identidades de terceiros pode ser um provedor de OAuth 2.0 para consumidores (sociais), como Apple ou Google, um provedor de identidades SAML ou OIDC personalizado ou um esquema de autenticação personalizado, também chamado de provedor de desenvolvedor, criado por você.

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 OAuth do provedor social por credenciais. 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, OAuth e OIDC em uma solicitação de API para credenciais de curto prazo. AssumeRoleWithWebIdentity

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: Configurando o Login with Amazon como um IdP de grupos de identidades, Configurando o Facebook como um IdP de grupos de identidades, Configurando o Google como um IdP do pool de identidades, Configurando o Login com a Apple como um IdP de pool de identidade, Twitter.
- Grupos de usuários do Amazon Cognito
- Configurando um provedor OIDC como um IdP do pool de identidades
- Configurando um provedor SAML como um IdP do grupo de identidades
- Identidades autenticadas pelo desenvolvedor (bancos de identidades)

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

Como usar grupos de identidades (identidades federadas)

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. Um grupo de identidades é um repositório de dados de identidade de usuários específicos para sua conta.

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 ter 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 receber uma solicitação para fornecer informações do cliente da aplicação OAuth, selecionar um grupo de usuários do Amazon Cognito, escolher um IdP do IAM ou inserir um identificador personalizado para um provedor de desenvolvedor.

   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 você deseja comparar a declaração, 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. 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 ter mais informações, consulte Fluxo de autenticação dos grupos de identidades (identidades federadas).
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 Provedores externos de identidade de grupos de identidades.
- Para configurar seu próprio processo de autenticação de backend, consulte Identidades autenticadas pelo desenvolvedor (bancos de identidades).

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 Introdução aos grupos de identidade 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 ter mais informações, consulte Permissões e confiança de função.

C. Selecione Save Changes (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 Edit (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 ter mais informações, consulte Permissões e confiança de função.

5. Selecione Save Changes (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 ter 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 ter mais informações, consulte Provedores externos de identidade de grupos de identidades.

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 você deseja comparar a declaração, 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. 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 Save Changes (Salvar alterações).

Excluir um banco 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. Selecione a opção 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 Amazon Cognito 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 ter mais informações, consulte Permissões e confiança de função.
8. Selecione Save Changes (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 Amazon Cognito 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 Eventos do Amazon Cognito.

Para saber mais sobre AWS Lambda, consulte AWS Lambda.

Conceitos de grupos de identidades

É possível usar grupos de identidades do Amazon Cognito para criar identidades exclusivas para os usuários e autenticá-los com provedores de identidade. Com uma identidade, você pode obter AWS credenciais temporárias com privilégios limitados para acessar outras. Serviços da AWS Os grupos de identidades do Amazon Cognito aceitam provedores públicos de identidade, como Amazon, Apple, Facebook e Google, além de identidades não autenticadas. Ele é também compatível com as identidades autenticadas do desenvolvedor, que permitem a você registrar e autenticar usuários por meio de seu próprio processo de autenticação de backend.

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

Fluxo de autenticação dos grupos de identidades (identidades federadas)

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 grupos de identidades aceitam os seguintes artefatos dos provedores:

Provedor	Artefato de autenticação
Conjunto de usuários do Amazon Cognito	Token de ID
OpenID Connect (OIDC)	Token de ID
SAML 2.0	Afirmação SAML
Provedor social	Token de acesso

Fluxo de autenticação aprimorado (simplificado)

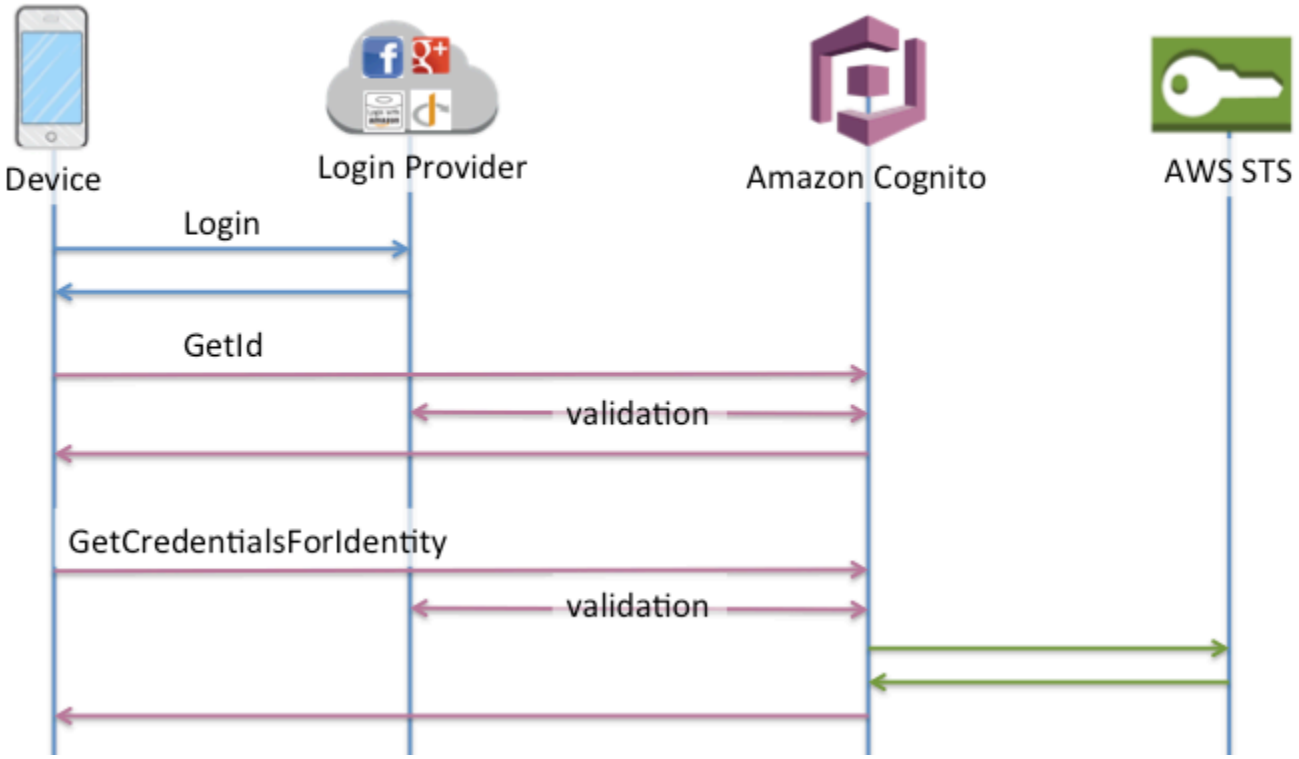
Quando você usa o fluxo de autenticação aprimorado, seu aplicativo primeiro apresenta uma prova de autenticação de um grupo de usuários autorizado do Amazon Cognito ou de um provedor de identidade terceirizado em GetIduma solicitação.

1. Seu aplicativo apresenta um comprovante de autenticação — um token web JSON ou uma declaração SAML — de um grupo de usuários autorizado do Amazon Cognito ou de um provedor de identidade terceirizado em uma solicitação GetID.
2. Seu grupo de identidades retorna uma 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 funções do IAM e da recuperação de credenciais na configuração do seu grupo de identidades. Você pode configurar seu grupo de identidades para selecionar uma função padrão, para aplicar os princípios de controle de acesso baseado em atributos (ABAC) ou controle de acesso baseado em função (RBAC) à seleção de funções. As AWS credenciais da autenticação avançada são válidas por uma hora.

Ordem das operações na autenticação avançada

1. `GetId`
2. `GetCredentialsForIdentity`



Fluxo de autenticação básico (clássico)

Quando você usa o fluxo de autenticação básico,

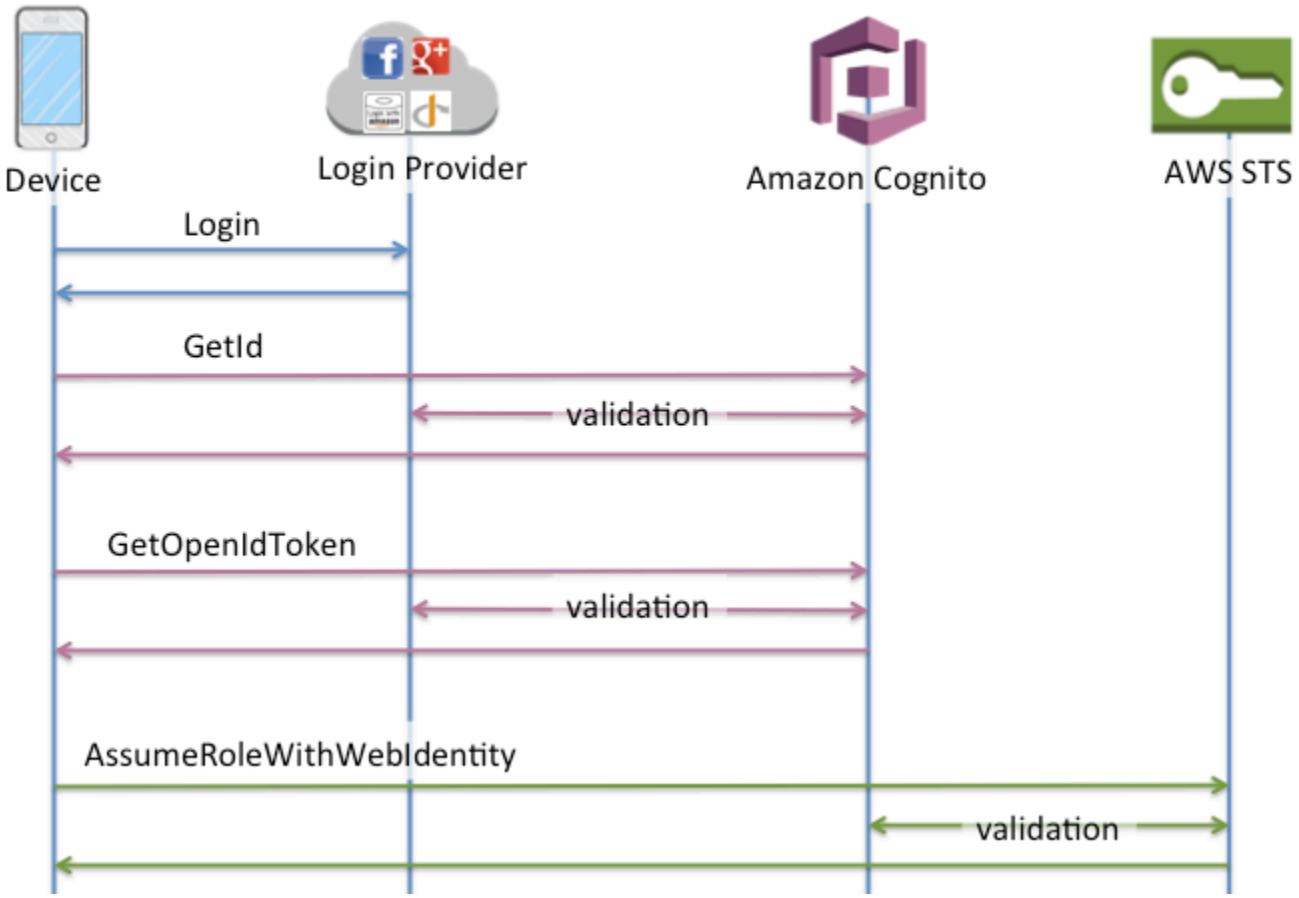
1. Seu aplicativo apresenta um comprovante de autenticação — um token web JSON ou uma declaração SAML — de um grupo de usuários autorizado do Amazon Cognito ou de um provedor de identidade terceirizado em uma solicitação GetID.
2. Seu grupo de identidades retorna uma ID de identidade.
3. Seu aplicativo 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.

Ordem das operações na autenticação básica

1. `GetId`
2. `GetOpenIdToken`
3. `AssumeRoleWithWebIdentity`



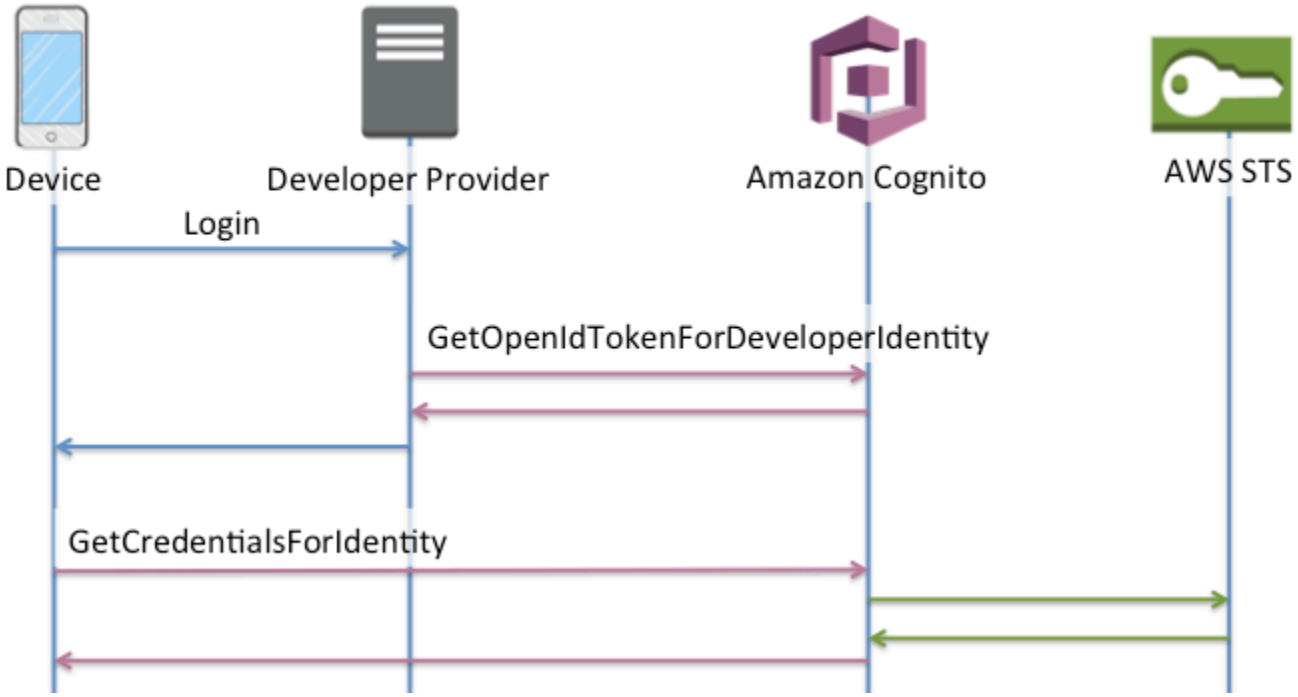
Fluxo de autenticação de identidades autenticadas pelo desenvolvedor

Ao usar Identidades autenticadas pelo desenvolvedor (bancos de identidades), o cliente utilizará outro fluxo de autenticação que incluirá o código fora do Amazon Cognito para validar o usuário em seu próprio sistema de autenticação. O código fora do Amazon Cognito é apropriadamente indicado.

Fluxo de autenticação aprimorado

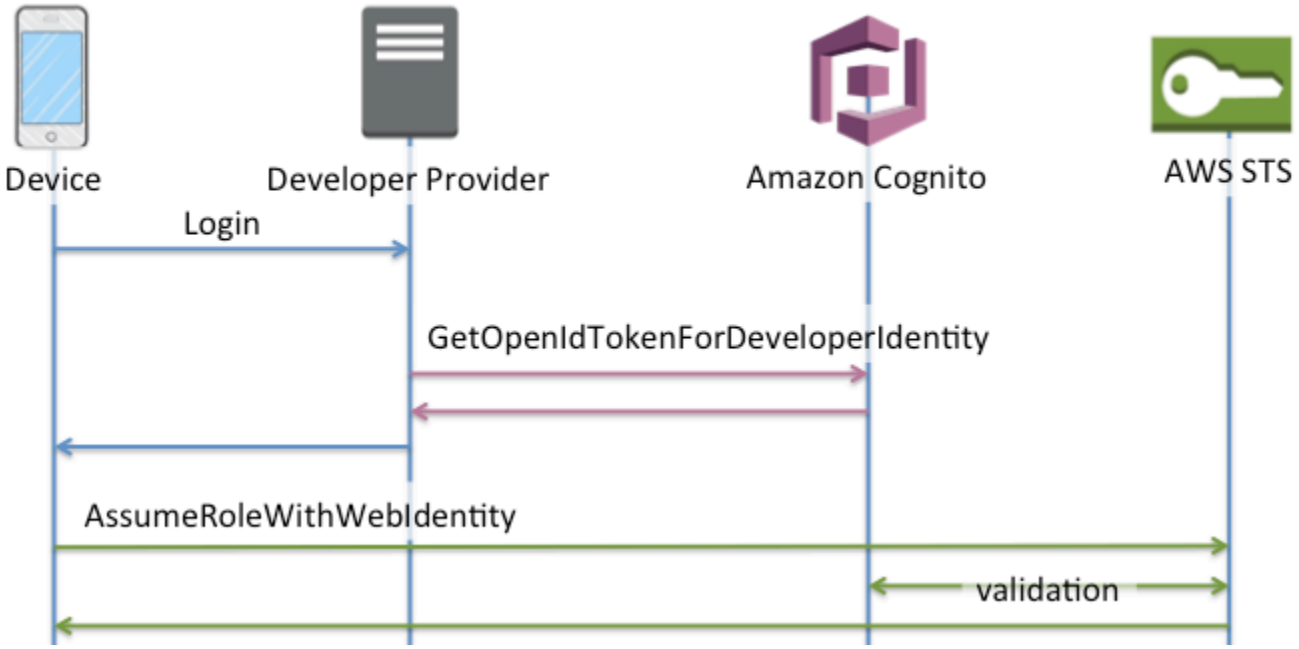
Ordem das operações na autenticação avançada com um provedor de desenvolvedores

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 de desenvolvedores

1. Implemente a lógica fora do pool de identidades para fazer login e gerar um identificador de desenvolvedor-provedor.
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 usar?

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.
- Seu aplicativo não precisa fazer solicitações adicionais de API para AWS STS o.
- Seu grupo de identidades avalia seus usuários quanto às credenciais da função do IAM que eles deveriam receber. Você não precisa incorporar lógica para seleção de funções em seu cliente.

> **Important**
>
> Ao criar um novo grupo 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 suas funções do IAM para identidades da web. Em seguida, incorpore a lógica para seleção de funções em seu cliente e proteja o cliente contra modificações por usuários.

O fluxo básico de autenticação delega a lógica da seleção da função do IAM ao seu aplicativo. 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 identidades e/ou em seu estado autenticado/não autenticado.

Da mesma forma, entenda que a autenticação do desenvolvedor é um atalho para a validação da autenticação do provedor de identidade. 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 do acesso dos usuários.

Resumo de APIs

GetId

A chamada de GetIdAPI 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. Os SDKs AWS móveis e o AWS SDK do navegador têm provedores de credenciais que gerenciam esse armazenamento JavaScript em cache para você.

Acesso autenticado

Quando você tiver configurado a aplicação para aceitar um provedor de login público (Facebook, Google +, Login with Amazon, Sign in with Apple), os usuários também poderão fornecer tokens (OAuth ou OpenID Connect) que os identificarão 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 GetCredentialsForIdentityAPI pode ser chamada depois que você estabelece 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 por meio da SetIdentityPoolRolesoperação.

GetOpenIdToken

Faça uma solicitação de GetOpenIdTokenAPI depois de estabelecer um ID de identidade. Armazene IDs de identidade em cache após sua primeira solicitação e inicie sessões básicas (clássicas) subsequentes para essa identidade com `GetOpenIdToken`.

A resposta a uma solicitação de API `GetOpenIdToken` é um token gerado pelo Amazon Cognito. Você pode enviar esse token como `WebIdentityToken` parâmetro em uma AssumeRoleWithWebIdentitysolicitação.

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. Uma vez mescladas, uma identidade se torna o pai/proprietário de todos os logins associados, enquanto o outro é desabilitado. Nesse caso, o ID de identidade do pai/proprietário é retornado. Você deverá atualizar seu cache local se esse valor for diferente. Os provedores nos SDKs AWS móveis 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 seu aplicativo assina solicitações para essa operação de API com AWS credenciais, o Amazon Cognito confia que o identificador de usuário fornecido na solicitação é válido. A autenticação do desenvolvedor substitui a validação do token que o Amazon Cognito executa com provedores externos.

A carga útil dessa API inclui um `logins` mapa. Esse mapa deve conter a chave do seu provedor de desenvolvimento e um valor como identificador para o usuário em seu 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 IDs de identidade de desenvolvedor em cache após sua primeira solicitação e inicie sessões básicas (clássicas) subsequentes para essa 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

Com identidades autenticadas pelo desenvolvedor, o Amazon Cognito suporta tanto a fusão implícita quanto a fusão explícita 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

diferentes funções do IAM para seu aplicativo: uma para usuários não autenticados e outra para usuários autenticados. O console do Amazon Cognito pode criar funções padrão quando você configura seu grupo de identidades pela primeira vez. Essas funções efetivamente não têm permissões concedidas. Modifique-os para atender às 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 no seu aplicativo, o Amazon Cognito gera AWS credenciais temporárias 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 seus AWS recursos.

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 ter 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
- Políticas de acesso

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 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.

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.

```
{
    "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 funções do IAM na autenticação básica (clássica)

Você deve aplicar pelo menos uma condição que limite as políticas de confiança para funções que você usa com grupos de identidades. Quando você cria ou atualiza políticas de confiança de

funções para grupos de identidades, o IAM retorna um erro se você tentar salvar suas alterações sem pelo menos uma chave de condição que limite as identidades de origem. AWS STS não permite AssumeRoleWithWebIdentityoperações entre contas, de grupos de identidades a funções do IAM que não tenham uma condição desse tipo.

Este tópico inclui várias condições que limitam as identidades de origem para grupos de identidades. Para obter uma lista completa, consulte Chaves disponíveis para federação de identidades AWS da web.

Na autenticação básica ou clássica com um grupo de identidades, você pode assumir qualquer função do IAM AWS STS se ela tiver a política de confiança correta. Os perfis do IAM para bancos de identidades do Amazon Cognito confiam na entidade principal do serviço `cognito-identity.amazonaws.com` para assumir o perfil. Essa configuração não é suficiente para proteger suas funções do IAM contra o acesso não intencional aos recursos. Funções desse tipo devem aplicar uma condição adicional à política de confiança da função. Você não pode criar ou modificar funções para grupos de identidades sem pelo menos uma das seguintes condições.

`cognito-identity.amazonaws.com:aud`

Restringe a função às operações de um ou mais grupos de identidades. O Amazon Cognito indica o grupo de identidade de origem na `aud` declaração no token do grupo de identidades.

`cognito-identity.amazonaws.com:amr`

Restringe a função a um `authenticated` ou a usuários `unauthenticated` (convidados). O Amazon Cognito indica o estado de autenticação na `amr` declaração no token do grupo de identidades.

`cognito-identity.amazonaws.com:sub`

Restringe a função a um ou mais usuários por UUID. Esse UUID é o ID de identidade do usuário no grupo de identidades. Esse valor não é o `sub` valor do provedor de identidade original do usuário. O Amazon Cognito indica esse UUID na `sub` declaração no token do grupo de identidades.

A autenticação de fluxo aprimorado exige que a função do IAM esteja na Conta da AWS mesma do grupo de identidades, mas esse não é o caso na autenticação básica.

Considerações adicionais se aplicam aos bancos de identidades do Amazon Cognito que assumem perfis do IAM entre contas. As políticas de confiança dessas funções devem aceitar o diretor do `cognito-identity.amazonaws.com` serviço e devem conter a `cognito-`

`identity.amazonaws.com:aud` condição específica. Para evitar o acesso não intencional aos seus AWS recursos, a chave de `aud` condição restringe a função aos usuários dos grupos de identidades no valor da condição.

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 essas condições não estiverem presentes na política de confiança da função. 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 das funções do seu grupo de identidades.

Condições adicionais da política de confiança

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 condição `sub` para restringir ações aos IDs de identidade do Amazon Cognito nas 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
Análises	Amazon Data Firehose Amazon Managed Service for Apache Flink
Integração de aplicativo	Amazon Simple Queue Service
AR e VR	Amazon Sumerian[1]
Aplicativos de negócios	Amazon Mobile Analytics Amazon Simple Email Service
Computação	AWS Lambda

Categoria	Serviço
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
Desenvolvimento de jogos	Amazon GameLift
Internet das Coisas (IoT)	AWS IoT
Machine Learning	Amazon CodeWhisperer
	Amazon Comprehend
	Amazon Lex
	Amazon Machine Learning
	Amazon Personalize
	Amazon Polly
	Amazon Rekognition
	Amazon SageMaker [1]
	Amazon Textract[1]
	Amazon Transcribe
	Amazon Translate

Categoria	Serviço
Gerenciamento 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`
	`ReEncrypt`
	`GenerateDataKey`

Amazon SageMaker	`InvokeEndpoint`
Amazon Textract	`DetectDocumentText` `AnalyzeDocument`
Amazon Sumerian	`View*`

Para conceder acesso Serviços da AWS além dessa lista, ative o fluxo de autenticação básico (clássico) em seu grupo de identidades. Se os usuários receberem erros `NotAuthorizedException` de Serviços da AWS que forem permitidos pelas políticas atribuídas ao perfil do IAM para usuários não autenticados, avalie se você pode remover esse serviço de seu caso de uso. Se você não conseguir, mude para o fluxo básico.

A política de sessão em linha

A política de sessão em linha impede que as permissões efetivas de seu usuário incluam o acesso a qualquer pessoa Serviços da AWS externa 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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:*",
                "logs:*",
                "dynamodb:*",
                "kinesis:*",
                "mobileanalytics:*",
                "s3:*",
                "ses:*",
                "sns:*",
                "sqs:*",
                "lambda:*",
                "machinelearning:*",
                "execute-api:*",
                "iot:*",
                "gamelift:*",
                "scs:*",
                "cognito-identity:*",
                "cognito-idp:*",
                "lex:*",
                "polly:*",
                "comprehend:*",
                "translate:*",
                "transcribe:*",
                "rekognition:*",
                "mobiletargeting:*",
                "firehose:*",
```

```
                "appsync:*",
                "personalize:*",
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "sagemaker:InvokeEndpoint",
                "cognito-sync:*",
                "sumerian:View*",
                "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 sessão AWS gerenciada

O Amazon Cognito também limita o escopo das permissões de usuários não autenticados com a política `AmazonCognitoUnAuthedIdentitiesSessionPolicy` gerenciada pela AWS para os usuários não autenticados no fluxo aprimorado. Você também deve conceder essa permissão nas políticas vinculadas ao seu perfil do IAM não autenticado.

A política gerenciada `AmazonCognitoUnAuthedIdentitiesSessionPolicy` contém as permissões a seguir.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "rum:PutRumEvents",
            "polly:*",
            "comprehend:*",
            "translate:*",
            "transcribe:*",
            "rekognition:*",
            "mobiletargeting:*",
            "firehose:*",
            "personalize:*",
            "sagemaker:InvokeEndpoint"
        ],
        "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.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/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:::mybucket/us-east-1:12345678-1234-1234-1234-123456790ab`. No entanto, a identidade não receberia acesso a outros objetos no `arn:aws:s3:::mybucket`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": ["s3:ListBucket"],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket"],
      "Condition": {"StringLike": {"s3:prefix": ["${cognito-
identity.amazonaws.com:sub}/*"]}}
    },
    {
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/${cognito-identity.amazonaws.com:sub}/*"]
```

```
    }
  ]
}
```

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.

```
{
  "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.

```
{
    "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.

```
{
    "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.

```
{
```

```
  "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:

```
{
  "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 grupo de identidades, de acordo com o grupo 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.

Melhores práticas de segurança para grupos de identidade 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 grupo de identidades é garantir que seu aplicativo possa realizar o trabalho sem privilégios excessivos ou não intencionais. Para se proteger contra configurações incorretas de segurança, revise essas recomendações antes do lançamento de cada aplicativo que você deseja lançar para produção.

Tópicos

- Melhores práticas de configuração do IAM
- Melhores práticas de configuração do pool de identidades

Melhores práticas de configuração do IAM

Quando um convidado ou usuário autenticado inicia uma sessão em seu aplicativo que exige credenciais do grupo de identidades, seu aplicativo recupera AWS credenciais temporárias para uma função do IAM. As credenciais podem ser para uma função padrão, uma função escolhida pelas regras na configuração do seu grupo de identidades ou para uma função personalizada escolhida pelo seu aplicativo. Com as permissões atribuídas a cada função, seu usuário ganha acesso aos seus AWS recursos.

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.

Use condições de política de confiança nas funções do IAM

O IAM exige que as funções dos grupos 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 todas as funções do IAM que confiam no diretor do serviço de grupos de identidades`cognito-identity.amazonaws.com`.

- `cognito-identity.amazonaws.com:aud`: a declaração aud no token do grupo de identidades deve corresponder a um ID confiável do grupo de identidades.
- `cognito-identity.amazonaws.com:amr`: a declaração amr no token do grupo de identidade deve ser autenticada ou não autenticada. Com essa condição, você pode reservar o acesso a uma função somente para convidados não autenticados ou somente para usuários autenticados. Você pode refinar ainda mais o valor dessa condição para restringir a função aos usuários de um provedor específico, por exemplo`graph.facebook.com`.

O exemplo de política de confiança de função a seguir concede acesso a uma função sob as seguintes condições:

```
{
  "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 grupos de identidades

- `"Federated": "cognito-identity.amazonaws.com"`: os usuários devem vir de um grupo de identidades.
- `"cognito-identity.amazonaws.com:aud": "us-east-1:a1b2c3d4-5678-90ab-cdef-example11111"`: os usuários devem vir do grupo de identidades específicous-east-1:a1b2c3d4-5678-90ab-cdef-example11111.
- `"cognito-identity.amazonaws.com:amr": "authenticated"`: Os usuários devem estar autenticados. Usuários convidados não podem assumir a função.

Aplique permissões de privilégio mínimo

Ao definir permissões com políticas do IAM para acesso autenticado ou acesso de convidado, conceda somente as permissões específicas necessárias para realizar tarefas específicas, ou permissões de privilégios mínimos. O exemplo de política do IAM a seguir, quando aplicado a uma função, concede acesso somente para leitura a um único arquivo de imagem em um bucket do Amazon S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
```

```
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::mybucket/assets/my_picture.jpg"]
    }
  ]
}
```

Melhores práticas de configuração do pool de identidades

Os grupos de identidades têm opções flexíveis para a geração de AWS credenciais. Não use atalhos de design quando seu aplicativo pode funcionar com os métodos mais seguros.

Entenda os efeitos do acesso de convidados

O acesso de convidado não autenticado permite que os usuários recuperem seus dados Conta da AWS antes de entrarem. Qualquer pessoa que saiba o ID do seu grupo de identidades pode solicitar credenciais não autenticadas. Seu ID do grupo 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 após a autenticação dos usuários. Se seu aplicativo exigir acesso aos recursos antes do login, tome as seguintes precauções.

- Familiarize-se com as limitações automáticas impostas às funções não autenticadas.
- Monitore e ajuste as permissões de suas funções não autenticadas do IAM para atender às necessidades específicas do seu aplicativo.
- Conceda acesso a recursos específicos.
- Proteja a política de confiança da sua função padrão não autenticada do IAM.
- Ative o acesso de convidado somente quando tiver certeza de que concederia as permissões em sua função do IAM a qualquer pessoa na Internet.

Use a autenticação aprimorada por padrão

Com a autenticação básica (clássica), o Amazon Cognito delega a seleção da função do IAM ao seu aplicativo. Por outro lado, o fluxo aprimorado usa a lógica centralizada em seu grupo de identidades para determinar a função 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, consulteFluxo de autenticação dos grupos de identidades (identidades federadas).

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 a função por trás da automação do pool de identidades.

Ao configurar a autenticação básica, aplique as melhores práticas do IAM às suas funções do IAM e às permissões delas.

Use provedores de desenvolvedores com segurança

As identidades autenticadas do desenvolvedor são um recurso dos grupos de identidades para aplicativos 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 grupos de identidades não impõem nenhuma 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 funções vinculadas a serviços, como perfis de instância do EC2 e funções de execução do Lambda.
- Evite misturar fontes de confiança internas e externas no mesmo grupo de identidades. Adicione seu provedor de desenvolvedores e seus provedores de login único (SSO) em grupos de identidade 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 conta da AWS
- 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 o perfil 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 Save Changes (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

```
   {
    "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

```
{
  "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 Save Changes (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.

Provedor	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.com

Provedor	Tipo de token	Valores de tag da entidade principal	Exemplo
			pute-1.client", "001968.a6ca34e9c1e742458a26cf8005854be9.0733"
Amazon	Token de acesso	aud (ID do cliente no Amzn Dev Ac), user_id (ID do usuário)	"amzn1.application-oa2-client.9d70d9382d3446108aaee3dd763a0fa6", "amzn1.account.AGHNIFJQMFSBG3G6XCPVB35ORQAA"
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)	“DfwifTtKEX1FiIBRnOTIR0CFK; Xgj5xb8xIrIVCPjXgLIdkW7fXmwcJJrFvnoK9gwZkLexo1y5z1”, “1269003884292222976”
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:

```
{
  "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 grupo de identidades, de acordo com o grupo 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`.

```
{
    "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 solicitação `cognito:roles` é uma string separada por vírgulas que contém um conjunto de ARNs de função permitidos.

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 Atributos de grupo de usuários.

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.

Você pode configurar o mapeamento baseado em regras para OpenID Connect (OIDC) e provedores de identidade SAML na API AWS CLI ou com o campo do tipo. `RulesConfiguration` RoleMapping Você pode especificar esse campo no `RoleMappings` parâmetro da SetIdentityPoolRolesAPI. AWS Management Console Atualmente, o não permite que você adicione regras para provedores OIDC ou SAML.

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 do aplicativo
- `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 do aplicativo
- `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 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.

Os provedores de credenciais de identidade da web fazem parte da cadeia de provedores de credenciais padrão nos 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 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 for JavaScript
- Provedor de credenciais do Amazon Cognito no Guia do desenvolvedor AWS SDK for .NET
- Especifique as credenciais programaticamente no Guia do desenvolvedor AWS SDK for Go
- Forneça credenciais temporárias em código no Guia do AWS SDK for Java 2.x desenvolvedor
- assumeRoleWithWebIdentityCredentialProviderprovedor no Guia do AWS SDK for PHP desenvolvedor
- Assumir o perfil de provedor de identidades na web na documentação do AWS SDK for 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 alguns AWS SDKs legados.

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 realizar E/S de rede 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 grupos 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 AWS SDK for .NET. Para começar a usar o Amazon Cognito no AWS SDK for .NET, consulte o provedor de credenciais do Amazon Cognito no Guia do desenvolvedor. AWS SDK for .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 AWS SDK for .NET. Para começar a usar o Amazon Cognito no AWS SDK for .NET, consulte o provedor de credenciais do Amazon Cognito no Guia do desenvolvedor. AWS SDK for .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 AWS serviços

Depois de configurar seu provedor de credenciais do Amazon Cognito e recuperar as AWS credenciais, você pode criar um cliente. AWS service (Serviço da AWS)

AWS Recursos do SDK para criar um cliente

- AWS Configuração do cliente no Guia do AWS SDK for C++ desenvolvedor

- Usando a AWS SDK for Go V2 com o Serviços da AWS Guia do AWS SDK for 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 for JavaScript desenvolvedor
- Criação de clientes na AWS SDK for Python (Boto3) documentação
- Criação de um cliente de serviço no Guia do AWS SDK para Rust desenvolvedor
- Usando clientes no Guia do AWS SDK for 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 AWS SDK for .NET. Para começar a usar o Amazon Cognito no AWS SDK for .NET, consulte o provedor de credenciais do Amazon Cognito no Guia do desenvolvedor. AWS SDK for .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 AWS SDK for .NET. Para começar a usar o Amazon Cognito no AWS SDK for .NET, consulte o provedor de credenciais do Amazon Cognito no Guia do desenvolvedor. AWS SDK for .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.

Provedores externos de identidade de grupos de identidades

Usando a propriedade `logins`, é possível definir as credenciais recebidas de um provedor de identidade (IdP). Além disso, você 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

- Configurando o Facebook como um IdP de grupos de identidades
- Configurando o Login with Amazon como um IdP de grupos de identidades
- Configurando o Google como um IdP do pool de identidades
- Configurando o Login com a Apple como um IdP de pool de identidade
- Configurando um provedor OIDC como um IdP do pool de identidades
- Configurando um provedor SAML como um IdP do grupo de identidades

Configurando o Facebook como um IdP de grupos de identidades

Os grupos de identidade do Amazon Cognito se integram ao Facebook para fornecer autenticação federada aos usuários da aplicação móvel. Esta seção explica como inscrever e configurar a aplicação com o Facebook como IdP.

Configurar o Facebook

É necessário inscrever a aplicação com o Facebook para que você possa começar a autenticar usuários do Facebook e interagir com as respectivas 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:

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.

Unity

Para obter mais informações sobre como integrar aplicações Unity ao Login do Facebook, consulte o Guia de conceitos básicos do Facebook.

Xamarin

Para adicionar a autenticação do Facebook, primeiro siga o fluxo apropriado a seguir para integrar o SDK do Facebook à aplicação. Os grupos de identidades do Amazon Cognito usam o token de acesso do Facebook para gerar um identificador exclusivo do usuário que está associado a uma identidade do Amazon Cognito.

- SDK do Facebook para iOS por Xamarin
- SDK do Facebook para Android por Xamarin

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 da aplicação do projeto OAuth que você criou em 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 você deseja comparar a declaração, 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. 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 Save Changes (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 token OAuth que o Amazon Cognito usa para AWS gerar 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).

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 a Web para adicionar 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 token OAuth que o Amazon Cognito usa para gerar AWS 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
    }
  });
}
```

Configurando o Login with Amazon como um IdP de grupos de identidades

O Amazon Cognito se integra ao 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 uma aplicação OAuth para integrá-la ao grupo de identidades, localizar a documentação do Login with Amazon e baixar 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 o 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 da aplicação do projeto OAuth que você criou em Login for 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 você deseja comparar a declaração, 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. 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 Save Changes (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 no método requestDidSucceed da AMZN: AccessTokenDelegate

```
- (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'
   }
});
```

Use o Login with Amazon: Xamarin

Xamarin para Android

```
AmazonAuthorizationManager manager = new AmazonAuthorizationManager(this,
 Bundle.Empty);

var tokenListener = new APIListener {
  Success = response => {
    // Get the auth token
    var token = response.GetString(AuthzConstants.BUNDLE_KEY.Token.Val);
    credentials.AddLogin("www.amazon.com", token);
  }
};

// Try and get existing login
manager.GetToken(new[] {
  "profile"
}, tokenListener);
```

Xamarin para iOS

Em `AppDelegate.cs`, insira o seguinte:

```
public override bool OpenUrl (UIApplication application, NSUrl url, string
 sourceApplication, NSObject annotation)
{
    // Pass on the url to the SDK to parse authorization code from the url
    bool isValidRedirectSignInURL = AIMobileLib.HandleOpenUrl (url, sourceApplication);
    if(!isValidRedirectSignInURL)
        return false;

    // App may also want to handle url
    return true;
}
```

Depois, em `ViewController.cs`, faça o seguinte:

```
public override void ViewDidLoad ()
{
    base.LoadView ();

    // Here we create the Amazon Login Button
    btnLogin = UIButton.FromType (UIButtonType.RoundedRect);
    btnLogin.Frame = new RectangleF (55, 206, 209, 48);
    btnLogin.SetTitle ("Login using Amazon", UIControlState.Normal);
```

```
    btnLogin.TouchUpInside += (sender, e) => {
        AIMobileLib.AuthorizeUser (new [] { "profile"}, new AMZNAuthorizationDelegate
 ());
    };
    View.AddSubview (btnLogin);
}

// Class that handles Authentication Success/Failure
public class AMZNAuthorizationDelegate : AIAuthenticationDelegate
{
  public override void RequestDidSucceed(ApiResult apiResult)
    {
      // Your code after the user authorizes application for requested scopes
      var token = apiResult["access_token"];
      credentials.AddLogin("www.amazon.com",token);
    }

    public override void RequestDidFail(ApiError errorResponse)
    {
      // Your code when the authorization fails
      InvokeOnMainThread(() => new UIAlertView("User Authorization Failed",
 errorResponse.Error.Message, null, "Ok", null).Show());
    }
}
```

Configurando o Google como um IdP do pool de identidades

O Amazon Cognito se integra com o Google para fornecer autenticação federada para os usuários do aplicativo 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 IDs de cliente criados como valores de público extras para melhorar a 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 & Services (APIs e serviços) e, em seguida, OAuth consent screen (Tela de consentimento OAuth). 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 OAuth client ID (ID de cliente OAuth). 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 seu aplicativo Android, consulte Autenticar usuários com o Sign in with 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 projeto OAuth que você criou no Google Cloud Platform. Para ter mais informações, consulte Configurar o OAuth 2.0 na Ajuda do console da 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 você deseja comparar a declaração, 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. 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 Save Changes (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 IDs de cliente criados como valores de público extras para melhorar a 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 & Services (APIs e serviços) e, em seguida, OAuth consent screen (Tela de consentimento OAuth). 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 OAuth client ID (ID de cliente OAuth). 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 projeto OAuth que você criou no Google Cloud Platform. Para ter mais informações, consulte Configurar o OAuth 2.0 na Ajuda do console da 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 você deseja comparar a declaração, 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. 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 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`, 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 IDs de cliente criados como valores de público extras para melhorar a 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 & Services (APIs e serviços) e, em seguida, OAuth consent screen (Tela de consentimento OAuth). 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 OAuth client ID (ID de cliente OAuth). 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 IDs de cliente criados como valores de público extras para melhorar a 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 & Services (APIs e serviços) e, em seguida, OAuth consent screen (Tela de consentimento OAuth). 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 OAuth client ID (ID de cliente OAuth). 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 projeto OAuth que você criou no Google Cloud Platform. Para ter mais informações, consulte Configurar o OAuth 2.0 na Ajuda do console da 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 você deseja comparar a declaração, 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. 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 Save Changes (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.
     });
```

```
  }
}
```

Unity

Como configurar o Google

Para habilitar o Login do Google para uma aplicação Unity, 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 & Services (APIs e serviços) e, em seguida, OAuth consent screen (Tela de consentimento OAuth). 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 OAuth client ID (ID de cliente OAuth). 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. Para Unity, crie um ID de cliente OAuth adicional para Android e outro para iOS.
5. 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).
6. Conceda à conta de serviço acesso ao seu projeto. Conceda aos usuários acesso à conta de serviço conforme a aplicação exigir.
7. 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.

Criar um provedor OpenID no console do IAM

1. Crie um provedor OpenID no console do IAM. Para obter informações sobre como configurar um provedor OpenID, consulte Usar provedores de identidade do OpenID Connect.
2. Ao receber a solicitação para o URL do provedor, insira `"https://accounts.google.com"`.
3. Quando solicitado a informar um valor no campo Público, insira qualquer um dos três IDs de cliente criados nas etapas anteriores.

4. Escolha o nome do provedor e adicione mais dois públicos com os outros dois IDs de cliente.

Configurar o provedor externo no console do Amazon Cognito

Escolha Manage Identity Pools (Gerenciar grupos de identidades) na página inicial do 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 projeto OAuth que você criou no Google Cloud Platform. Para ter mais informações, consulte Configurar o OAuth 2.0 na Ajuda do console da 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 você deseja comparar a declaração, 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. 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 Save Changes (Salvar alterações).

Instalar o Unity Google Plugin

1. Adicione o plugin do Unity da Google Play Games ao projeto Unity.
2. Em Unity, no menu Windows, use os três IDs para as plataformas Android e iOS para configurar o plugin.

Usar o Google

O código de exemplo a seguir mostra como recuperar o token de autenticação do Google Play Service:

```
void Start()
{
  PlayGamesClientConfiguration config = new
 PlayGamesClientConfiguration.Builder().Build();
  PlayGamesPlatform.InitializeInstance(config);
  PlayGamesPlatform.DebugLogEnabled = true;
  PlayGamesPlatform.Activate();
  Social.localUser.Authenticate(GoogleLoginCallback);
}

void GoogleLoginCallback(bool success)
{
  if (success)
  {
    string token = PlayGamesPlatform.Instance.GetIdToken();
    credentials.AddLogin("accounts.google.com", token);
  }
  else
  {
    Debug.LogError("Google login failed. If you are not running in an actual Android/
iOS device, this is expected.");
  }
}
```

Xamarin

> **Note**
>
> O Amazon Cognito não comporta nativamente o Google na plataforma Xamarin. No momento, a integração requer o uso de uma visualização da web para passar pelo fluxo de login do navegador. Para saber como a integração com o Google funciona com outros SDKs, selecione outra plataforma.

Para habilitar o login com o Google na aplicação, será autentique os usuários e obtenha deles um token OpenID Connect. O Amazon Cognito usa esse token para gerar um identificador exclusivo do usuário que está associado a uma identidade do Cognito. Infelizmente, o Google SDK para Xamarin não permite que você recupere o token OpenID Connect. Por isso, use um cliente alternativo ou o fluxo da Web em uma visualização da Web.

Depois que tiver o token, você poderá configurá-lo em `CognitoAWSCredentials`:

```
credentials.AddLogin("accounts.google.com", token);
```

> **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 IDs de cliente criados como valores de público extras para melhorar a integração. Para saber mais sobre o modelo de identidade entre clientes do Google, consulte Identidade em vários clientes.

Configurando o Login com a Apple como um IdP de pool de identidade

O Amazon Cognito se integra ao 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 à esquerda, 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 Register a New Identifier (Registrar um novo identificador), escolha App IDs (IDs de aplicação) e selecione Continue (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 Identifiers (Identificadores), escolha o menu App IDs (IDs de aplicação) e, em seguida, Services IDs (IDs de serviços). Escolha o ícone +.
11. Na página Register a New Identifier (Registrar um novo identificador), escolha Services IDs (IDs de serviços) e selecione Continue (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 Website URLs (URLs de site), escolha o ícone +. Em Domínios e subdomínios, insira o nome de domínio do seu aplicativo. Em Return URLs, (URLs de retorno), insira o URL de retorno de chamada no qual a autorização redireciona o usuário depois que ele se autentica por meio do Sign in with Apple.

e. Selecione Next (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 projeto OAuth que você criou em Meta for Developers. 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 você deseja comparar a declaração, 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. 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 Save Changes (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 (Fazer login com a Apple) à interface de usuário do Android, siga Displaying and Configuring Sign In with Apple Buttons 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 Entrar com a Apple à sua interface de JavaScript usuário, siga Exibindo e configurando os botões de login com a Apple na documentação da Apple.
- Para autenticar usuários com segurança usando Sign in with Apple, siga Configuring Your Web page for Sign in with Apple (Configurar sua página da Web para 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.
    });
}
```

Usar o recurso Fazer login com a Apple: Xamarin

Não temos um SDK compatível com o recurso Fazer login com a Apple para Xamarin. 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 interface de usuário do Xamarin, siga Displaying and Configuring Sign In with Apple Buttons na documentação da Apple.
- Para autenticar usuários com segurança usando Sign in with Apple, siga Configuring Your Web page for Sign in with Apple (Configurar sua página da Web para 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

Depois que tiver o token, você poderá configurá-lo em `CognitoAWSCredentials`:

```
credentials.AddLogin("appleid.apple.com", token);
```

Configurando um provedor OIDC como um IdP do pool 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 você deseja comparar a declaração, 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. 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 Save Changes (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 defasagem do relógio entre 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.
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 parâmetro `azp` estiver presente, confirme esse valor em relação aos IDs de cliente listados no provedor OIDC.
5. Se o parâmetro `azp` não estiver presente, verifique o parâmetro `aud` em relação aos IDs de cliente listados no 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 }
```

iOS – Swift

Para fornecer o token de ID do OIDC ao Amazon Cognito, implemente o protocolo `AWSIdentityProviderManager`.

Na implementação do método `logins`, retorne um dicionário contendo o nome do provedor OIDC que você configurou. Esse dicionário atua como a chave, ao passo que o token de ID atual do usuário autenticado atua como o valor, conforme mostrado no exemplo de código a seguir.

```
class OIDCProvider: NSObject, AWSIdentityProviderManager {
    func logins() -> AWSTask<NSDictionary> {
        let completion = AWSTaskCompletionSource<NSString>()
        getToken(tokenCompletion: completion)
        return completion.task.continueOnSuccessWith { (task) -> AWSTask<NSDictionary>?
 in
            //login.provider.name is the name of the OIDC provider as setup in the
 Amazon Cognito console
            return AWSTask(result:["login.provider.name":task.result!])
        } as! AWSTask<NSDictionary>

    }

    func getToken(tokenCompletion: AWSTaskCompletionSource<NSString>) -> Void {
        //get a valid oidc token from your server, or if you have one that hasn't
 expired cached, return it

        //TODO code to get token from your server
        //...

        //if error getting token, set error appropriately
        tokenCompletion.set(error:NSError(domain: "OIDC Login", code: -1 , userInfo:
["Unable to get OIDC token" : "Details about your error"]))
        //else
        tokenCompletion.set(result:"result from server id 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

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
 IdentityPoolId: 'IDENTITY_POOL_ID',
 Logins: {
    'login.provider.com': token
 }
});
```

Unity

```
credentials.AddLogin("login.provider.com", token);
```

Xamarin

```
credentials.AddLogin("login.provider.com", token);
```

Configurando um provedor SAML como um IdP do grupo de identidades

O Amazon Cognito oferece suporte à autenticação com provedores de identidade (IdPs) por meio da Security Assertion Markup Language 2.0 (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 você deseja comparar a declaração, 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. 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 Save Changes (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 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 grupo de usuários determina que a declaração do 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. Assim que autenticar o usuário, você poderá usar APIs do Amazon Cognito para fornecer a declaração SAML resultante à identidade do Amazon Cognito.

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 GetOpen ID. TokenForDeveloperIdentity 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`.

Android

Se você usar o SDK do Android, será possível preencher o mapa de logins com a declaração do SAML da forma a seguir.

```
Map logins = new HashMap();
logins.put("arn:aws:iam::aws account id:saml-provider/name", "base64 encoded assertion
 response");
// Now this should be set to CognitoCachingCredentialsProvider object.
CognitoCachingCredentialsProvider credentialsProvider = new
 CognitoCachingCredentialsProvider(context, identity pool id, region);
credentialsProvider.setLogins(logins);
// If SAML assertion contains multiple roles, resolve the role by setting the custom
 role
credentialsProvider.setCustomRoleArn("arn:aws:iam::aws account id:role/
customRoleName");
// This should trigger a call to the Amazon Cognito service to get the credentials.
credentialsProvider.getCredentials();
```

iOS

Caso esteja usando o SDK do iOS, é possível fornecer a declaração do SAML em `AWSIdentityProviderManager` da forma a seguir.

```
- (AWSTask<NSDictionary<NSString*,NSString*> *> *) logins {
    //this is hardcoded for simplicity, normally you would asynchronously go to your
 SAML provider
    //get the assertion and return the logins map using a AWSTaskCompletionSource
    return [AWSTask taskWithResult:@{@"arn:aws:iam::aws account id:saml-provider/
name":@"base64 encoded assertion response"}];
}

// If SAML assertion contains multiple roles, resolve the role by setting the custom
 role.
// Implementing this is optional if there is only one role.
- (NSString *)customRoleArn {
    return @"arn:aws:iam::accountId:role/customRoleName";
}
```

Identidades autenticadas pelo desenvolvedor (bancos de identidades)

O Amazon Cognito é compatível com identidades autenticadas pelo desenvolvedor, além da federação de identidades da web por meio de Configurando o Facebook como um IdP de grupos de identidades, Configurando o Google como um IdP do pool de identidades, Configurando o Login with Amazon como um IdP de grupos de identidades e Configurando o Login com a Apple como um IdP de pool de identidade. 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 `Logins` mapa é um nome de provedor do desenvolvedor do pool de identidades `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 grupo de identidades.

Autenticação básica

Chame a operação da AssumeRoleWithWebIdentityAPI e solicite a `RoleArn` de 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 grupos de identidades (identidades federadas)

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 Save Changes (Salvar alterações).

Observação: depois que o nome do provedor for definido, ele não poderá ser alterado.

Para obter instruções adicionais sobre como trabalhar com o console do Amazon Cognito, consulte Como usar o console do Amazon Cognito.

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.

Como alternar usuários não autenticados para usuários autenticados (grupos de identidades)

Os grupos de identidade do Amazon Cognito são compatíveis com usuários autenticados e não autenticados. Os usuários não autenticados recebem acesso aos recursos da AWS mesmo que não tenham feito login com um 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. Por fim, talvez eles decidam fazer login usando um dos IdPs com suporte. 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. Por fim, talvez eles decidam fazer login usando um dos IdPs com suporte. 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. Por fim, talvez eles decidam fazer login usando um dos IdPs com suporte. 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 objeto `CognitoIdentityCredentials`, consulte AWSCognitoIdentityCredentials na Referência de API do AWS SDK for JavaScript.

Unity

Os usuários podem fazer login em seu aplicativo como convidados não autenticados. Por fim, talvez eles decidam fazer login usando um dos IdPs com suporte. 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. Por fim, talvez eles decidam fazer login usando um dos IdPs com suporte. 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. Como o Amazon Cognito Sync, o AWS AppSync é um serviço para sincronizar dados de aplicações 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 service (Serviço da AWS) e uma biblioteca de clientes que permite a sincronização de dados de usuários relacionados a aplicações entre 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. Como o Amazon Cognito Sync, o AWS AppSync é um serviço para sincronizar dados de aplicações 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 serviço da AWS e uma biblioteca de clientes que permite a sincronização dos dados de usuários relacionados a aplicações entre dispositivos. 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 Introdução aos grupos de identidade 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.

Como sincronizar dados

> ⚠ Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Como o Amazon Cognito Sync, o AWS AppSync é um serviço para sincronizar dados de aplicações 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 credenciais temporárias da AWS para permitir que a aplicação acesse os recursos da 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 o CocoaPods, substitua `<AWSiOSSDKv2/AWSCore.h>` 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. Faça download do Gerenciador do Amazon Cognito Sync 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.

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.

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.

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.

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.

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.

Como manipular retornos de chamada

> ⚠ Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Como o Amazon Cognito Sync, o AWS AppSync é um serviço para sincronizar dados de aplicações 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.

Esta seção descreve como manipular retornos de chamada.

Android

Interface SyncCallback

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. `userInfo` conterá o conjunto de dados de chaves, que equivale ao conjunto de dados que está sendo sincronizado, e as chaves, que conterão um NSArray das chaves de registro que foram enviadas.

AWSCognitoDidChangeLocalValueFromRemoteNotification

Chamado quando um valor local é alterado devido a uma operação de sincronização. `userInfo` conterá o conjunto de dados de chaves, que equivale ao conjunto de dados que está sendo sincronizado, e as chaves, que conterão um 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: [conflito resolveWithLocalRecord], o registro remoto: [conflito resolveWithRemoteRecord] ou um novo valor: [conflito resolveWithValue: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 = ^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. `userInfo` conterá o conjunto de dados de chaves, que equivale ao conjunto de dados que está sendo sincronizado, e as chaves, que conterão um NSArray das chaves de registro que foram enviadas.

AWSCognitoDidChangeLocalValueFromRemoteNotification

Chamado quando um valor local é alterado devido a uma operação de sincronização. `userInfo` conterá o conjunto de dados de chaves, que equivale ao conjunto de dados que está sendo sincronizado, e as chaves, que conterão um 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: [conflito resolveWithLocalRecord], o registro remoto: [conflito resolveWithRemoteRecord] ou um novo valor: [conflito resolveWithValue: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;
}
```

Sincronização por push

⚠ Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Como o Amazon Cognito Sync, o AWS AppSync é um serviço para sincronizar dados de aplicações 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**
>
> Não há suporte para a sincronização por push em JavaScript, Unity ou Xamarin.

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 console do AWS Identity and Access Management, configure suas funções do IAM para ter acesso completo ao Amazon SNS ou crie uma nova função que tenha acesso completo 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`.

```
{
    "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 objeto Android BroadcastReceiver, você pode verificar a versão mais recente do conjunto de dados modificados e decidir se seu aplicativo precisa se sincronizar 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. 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 - Objective-C

Para obter um token de dispositivo para o aplicativo, siga a documentação da Apple sobre registro de notificações remotas. Assim que você receber o token do dispositivo como um objeto NSData dos APNs, precisará registrar o dispositivo no Amazon Cognito usando o método `registerDevice:` 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, o dispositivo será registrado por meio do sandbox dos APNs; no modo de lançamento, ele será registrado por meio dos 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. Assim que você receber o token do dispositivo como um objeto NSData dos APNs, 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, o dispositivo será registrado por meio do sandbox dos APNs; no modo de lançamento, ele será registrado por meio dos 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.

Amazon Cognito Streams

> ⚠ Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Como o Amazon Cognito Sync, o AWS AppSync é um serviço para sincronizar dados de aplicações 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.

Eventos do Amazon Cognito

⚠ Se você for novo com o Amazon Cognito Sync, use o AWS AppSync. Como o Amazon Cognito Sync, o AWS AppSync é um serviço para sincronizar dados de aplicações 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.

Os eventos do Amazon Cognito permitem que você execute uma função do AWS Lambda em resposta a eventos importantes do 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 chamar `UpdateRecords` com as credenciais da conta da AWS (credenciais do desenvolvedor), a função Lambda não será invocada.

Como criar uma função no 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 Select blueprint, procure 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. É possível adicionar essa permissão no console do Lambda (consulte Uso de políticas baseadas em recursos para o AWS Lambda) ou usando a operação AddPermission do Lambda.
> 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`.
>
> ```
> {
>     "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": "<your Lambda function ARN>",
            "Condition": {
                "StringEquals": {
                    "AWS:SourceAccount": "<your account number>"
                },
                "ArnLike": {
                    "AWS:SourceArn": "<your identity pool ARN>"
                }
            }
        }
    ]
}
```

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 chamar sua função do Lambda durante UpdateRecords, ela deverá responder em até cinco 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);
};
```

Como usar o console do Amazon Cognito

Você pode usar o console do Amazon Cognito para criar e gerenciar grupos de usuários e grupos de identidades.

Este guia fornece step-by-step orientações para tarefas comuns do grupo de usuários do Amazon Cognito no console do Amazon Cognito.

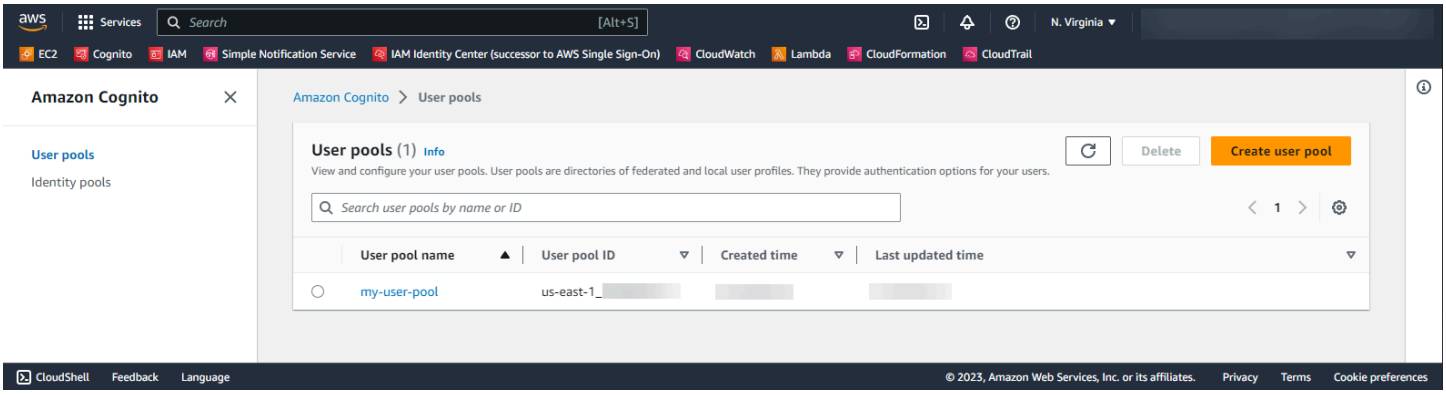
Como usar o console do Amazon Cognito

1. Para usar o Amazon Cognito, você precisa se inscrever em uma AWS conta.
2. Acesse o console do Amazon Cognito. Você pode ser solicitado a fornecer suas AWS credenciais.
3. Para criar ou editar um grupo de usuários, escolha User Pools (Grupos de usuários) no painel de navegação à esquerda.

   Para ter mais informações, consulte Conceitos básicos dos grupos de usuários.
4. Para criar ou editar um banco de identidades, selecione Bancos de identidades. Você será direcionado para o console original para grupos de identidade do Amazon Cognito.

   Para ter mais informações, consulte Introdução aos grupos de identidade do Amazon Cognito.

O console do Amazon Cognito faz parte do AWS Management Console, que fornece informações sobre sua conta e faturamento. Para obter mais informações, consulte Como trabalhar com o AWS Management Console.



Tópicos

- O console dos grupos de usuários

- O console de bancos de identidades

O console dos grupos de usuários

Na visualização Grupos de usuários do console do Amazon Cognito, selecione um grupo de usuários na lista para ver os detalhes. Na visualização detalhada, a Visão geral do grupo de usuários na parte superior do console contém informações básicas sobre seu grupo de usuários. As guias a seguir organizam a configuração do grupo de usuários em funções relacionadas.

Usuários

A guia Usuários contém informações sobre usuários e importações de usuários de arquivos CSV. Você pode adicionar, remover e editar usuários nessa guia.

Referências

- Como gerenciar usuários em seu grupo de usuários
- Como importar usuários para grupos de usuários com base em um arquivo CSV

Grupos

A guia Grupos contém informações sobre grupos de usuários. Você pode adicionar, modificar e alterar a associação em grupos e alterar os perfis do IAM associados aos grupos para a integração do banco de identidades.

Referências

- Como adicionar grupos a um grupo de usuários

Experiência de login

A guia Experiência de login contém informações sobre como os usuários fazem login no seu grupo de usuários. Nessa guia encontram-se provedores de identidades de terceiros, opções de nome de usuário, política de senha, configuração de autenticação multifator (MFA), comportamento de esquecimento de senha e memorização de dispositivos. Você pode adicionar e modificar provedores de identidades e alterar o comportamento geral de login do seu grupo de usuários.

Referências

- Como adicionar acesso a grupo de usuários por meio de terceiros
- Personalização dos atributos de login

- Como adicionar requisitos de senha do grupo de usuários
- Adicionar MFA a um grupo de usuários
- Como recuperar contas de usuário
- Trabalhar com dispositivos de usuários no grupo de usuários

Experiência de login

A guia Experiência de login contém informações sobre inscrição por autoatendimento, atributos obrigatórios, verificação de números de telefone e endereços de e-mail e atributos personalizados.

Referências

- Como cadastrar e confirmar contas de usuários
- Atributos de grupo de usuários
- Como verificar informações de contato no cadastro

Sistema de mensagens

A guia Sistema de mensagens contém informações sobre os Serviços da AWS que você deseja usar para enviar mensagens de e-mail e SMS aos usuários e o formato das mensagens que você deseja enviar a eles.

Referências

- Configurações de e-mail para grupos de usuários do Amazon Cognito
- Configurações de mensagens SMS para grupos de usuários do Amazon Cognito
- Como configurar mensagens de verificação de SMS e de e-mail, e mensagens de convite de usuário

Integração de aplicações

A guia Integração de aplicações contém informações sobre clientes de aplicação de grupos de usuários, o domínio atribuído aos endpoints de serviço do grupo de usuários, servidores de recursos de API, a interface de usuário hospedada e segurança avançada. Você pode detalhar cada cliente da aplicação para configurar o seguinte.

1. Configurações de token
2. URLs de retorno de chamada
3. Fluxos de autenticação
4. Permissões de atributo

5. Segurança avançada específica da aplicação e configurações de interface de usuário hospedada
6. Análise do Amazon Pinpoint

Referências

- Clientes de aplicações de grupos de usuários
- Configurar e usar a interface de usuário hospedada e endpoints de federação do Amazon Cognito
- Como configurar um domínio de grupo de usuários
- Escopos, M2M e autorização de API com servidores de recursos
- Como adicionar segurança avançada a um grupo de usuários
- Como usar análise do Amazon Pinpoint com grupos de usuários do Amazon Cognito

Propriedades do grupo de usuários

A guia Propriedades do grupo de usuários contém informações sobre a configuração do grupo de usuários não diretamente relacionadas aos usuários: acionadores Lambda, proteção de ACL da AWS WAF web, proteção contra exclusão e tags de recursos.

Referências

- Como personalizar fluxos de trabalho do grupo de usuários com acionadores do Lambda
- Associando uma ACL AWS WAF da web a um grupo de usuários
- Proteção contra exclusão do grupo de usuários
- Marcando seus recursos AWS

O console de bancos de identidades

Na visualização Bancos de identidades do console do Amazon Cognito, selecione um banco de identidades na lista para ver os detalhes. Na visualização detalhada, a Visão geral do grupo de identidades na parte superior do console contém informações básicas sobre seu grupo de usuários. As guias a seguir organizam a configuração do grupo de usuários em funções relacionadas.

Estatísticas do usuário

A guia Estatísticas do usuário exibe informações estatísticas sobre os usuários que geraram identidades em seu banco de identidades. Não é possível definir nenhuma configuração do banco de identidades nessa guia.

Navegador de identidade

A guia Navegador de identidade contém informações sobre as identidades individuais que os usuários geraram em seu banco de identidades. Você pode visualizar e excluir identidades.

Referências

- Introdução aos grupos de identidade do Amazon Cognito

Acesso do usuário

A guia Acesso de usuário contém informações sobre os provedores de identidades vinculados ao banco de identidades, os provedores de desenvolvedores, os perfis padrão do IAM atribuídos às identidades e a configuração de acesso de convidados não autenticados. Você pode detalhar cada provedor de identidades para configurar o seguinte:

1. Controle de acesso baseado em perfis com Seleção de função do IAM
2. Controle de acesso baseado em atributos com Atributos para controle de acesso

Referências

- Provedores externos de identidade de grupos de identidades
- Perfis do IAM
- Identidades autenticadas e não autenticadas
- Identidades autenticadas pelo desenvolvedor (bancos de identidades)
- Controle de acesso com base em perfil
- Usar atributos para controle de acesso

Propriedades do banco de identidades

A guia Propriedades do grupo de identidades contém informações sobre configurações diversas do banco de identidades: autenticação básica (clássica) e tags de recursos.

- Fluxo de autenticação dos grupos de identidades (identidades federadas)
- Marcando seus recursos AWS

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 no escopo do programa de conformidade AWS .
- 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 seu 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 os recursos. AWS
- 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 os AWS CLI 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.

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 ser compatíveis 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, suporta esses modos.

Além disso, as solicitações devem ser assinadas utilizando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou é possível usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

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 ter mais informações, consulte Operações de API autenticadas e não autenticadas de grupos de usuários do Amazon Cognito.

> **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

A forma como você usa AWS Identity and Access Management (IAM) difere, dependendo do trabalho que você faz no Amazon Cognito.

Usuário do serviço: se você usar o serviço do Amazon Cognito para fazer o trabalho, o administrador fornecerá as credenciais e as permissões necessárias. À medida você que usar mais atributos do Amazon Cognito para fazer seu trabalho, poderá precisar de permissões adicionais. Entender como o acesso é gerenciado pode ajudar você a solicitar as permissões corretas ao seu administrador. Se não for possível acessar um atributo no Amazon Cognito, consulte Solução de problemas de identidade e acesso do Amazon Cognito.

Administrador do serviço: se você for o responsável pelos recursos do Amazon Cognito na empresa, provavelmente terá acesso total a esse serviço. Cabe a você determinar quais atributos e recursos do Amazon Cognito os usuários do seu serviço devem acessar. Assim, você deve enviar solicitações ao administrador do IAM para alterar as permissões dos usuários de seu serviço. Revise as informações nesta página para entender os Introdução ao IAM. Para saber mais sobre como sua empresa pode usar o IAM com o Amazon Cognito, consulte Como o Amazon Cognito funciona com o IAM.

Administrador do IAM: se você for um administrador do IAM, talvez queira saber detalhes sobre como pode escrever políticas para gerenciar o acesso ao Amazon Cognito. Para visualizar exemplos

de políticas baseadas em identidade do Amazon Cognito que podem ser usadas no IAM, consulte Exemplos de políticas baseadas em identidade para o Amazon Cognito.

Autenticando com identidades

A autenticação é como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado (conectado AWS) como o Usuário raiz da conta da AWS, como usuário do IAM ou assumindo uma função do IAM.

Você pode entrar AWS como uma identidade federada usando credenciais fornecidas por meio de uma fonte de identidade. AWS IAM Identity Center Usuários (IAM Identity Center), a autenticação de login único da sua empresa e suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como identidade federada, o administrador já configurou anteriormente a federação de identidades usando perfis do IAM. Ao acessar AWS usando a federação, você está assumindo indiretamente uma função.

Dependendo do tipo de usuário que você é, você pode entrar no AWS Management Console ou no portal de AWS acesso. Para obter mais informações sobre como fazer login AWS, consulte Como fazer login Conta da AWS no Guia do Início de Sessão da AWS usuário.

Se você acessar AWS programaticamente, AWS fornece um kit de desenvolvimento de software (SDK) e uma interface de linha de comando (CLI) para assinar criptograficamente suas solicitações usando suas credenciais. Se você não usa AWS ferramentas, você mesmo deve assinar as solicitações. Para obter mais informações sobre como usar o método recomendado para assinar solicitações por conta própria, consulte Assinatura de solicitações de AWS API no Guia do usuário do IAM.

Independente do método de autenticação usado, também pode ser exigido que você forneça informações adicionais de segurança. Por exemplo, AWS recomenda que você use a autenticação multifator (MFA) para aumentar a segurança da sua conta. Para saber mais, consulte Autenticação Multifator no AWS IAM Identity Center Guia do Usuário. Usar a autenticação multifator (MFA) na AWS no Guia do Usuário do IAM.

Conta da AWS usuário root

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

completa das tarefas que exigem login como usuário raiz, consulte Tarefas que exigem credenciais de usuário raiz no Guia do usuário do IAM.

Identidade federada

Como prática recomendada, exija que usuários humanos, incluindo usuários que precisam de acesso de administrador, 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 de usuários corporativo, de um provedor de identidade da web AWS Directory Service, do diretório do Identity Center ou de qualquer usuário que acesse usando credenciais fornecidas Serviços da AWS por meio de uma fonte de identidade. Quando as identidades federadas acessam Contas da AWS, elas assumem funções, e as funções fornecem credenciais temporárias.

Para o gerenciamento de acesso centralizado, recomendamos usar o . AWS IAM Identity Center Você pode criar usuários e grupos no IAM Identity Center ou pode se conectar e sincronizar com um conjunto de usuários e grupos em sua própria fonte de identidade para uso em todos os seus Contas da AWS aplicativos. Para obter mais informações sobre o Centro de Identidade do IAM, consulte O que é o Centro de Identidade do IAM? no AWS IAM Identity Center Manual do Usuário do.

Usuários e grupos do IAM

Um usuário do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas para uma única pessoa ou aplicativo. Sempre que possível, recomendamos depender de credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM, recomendamos alternar as chaves de acesso. Para obter mais informações, consulte Alterne Chaves de Acesso Regularmente para Casos de Uso que exijam Credenciais de Longo Prazo no Guia do Usuário do IAM.

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

Usuários são diferentes de perfis. Um usuário é exclusivamente associado a uma pessoa ou a um aplicativo, mas uma função pode ser assumida por qualquer pessoa que precisar dela. Os usuários têm credenciais permanentes de longo prazo, mas os perfis fornecem credenciais temporárias.

Para saber mais, consulte Quando Criar um Usuário do IAM (Ao Invés de uma Função) no Guia do Usuário do IAM.

Perfis do IAM

Uma função do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas. Ele é semelhante a um usuário do IAM, mas não está associado a uma pessoa específica. Você pode assumir temporariamente uma função do IAM no AWS Management Console trocando de funções. Você pode assumir uma função chamando uma operação de AWS API AWS CLI ou usando uma URL personalizada. Para obter mais informações sobre métodos para usar perfis, consulte Usando Funções do IAM no Guia do Usuário do IAM.

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

- Acesso de usuário federado: para atribuir permissões a identidades federadas, você pode criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas pelo mesmo. Para obter mais informações sobre perfis para federação, consulte Criando um Perfil para um Provedor de Identidades Terceirizado no Guia do Usuário do IAM. Se você usa o IAM Identity Center, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o IAM Identity Center correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte Conjuntos de Permissões no AWS IAM Identity Center Manual do Usuário.
- Permissões de usuários temporárias do IAM: um usuário ou perfil do IAM pode assumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso entre contas: você pode usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) acesse recursos na sua conta de uma conta diferente. As funções são a forma primária de conceder acesso entre contas. No entanto, com alguns Serviços da AWS, você pode anexar uma política diretamente a um recurso (em vez de usar uma função como proxy). Para aprender a diferença entre funções e políticas baseadas em recurso para acesso entre contas, consulte Como as Funções do IAM Diferem das Políticas Baseadas em Recurso no Guia do Usuário do IAM.
- Acesso entre serviços — Alguns Serviços da AWS usam recursos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicativos no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões de chamada da entidade principal, uma função de serviço ou uma função vinculada ao serviço.

- Sessões de acesso direto (FAS) — Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.
- Função de Serviço: uma função de serviço é uma função do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criando um Perfil para Delegar Permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.
- Função vinculada ao serviço — 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 pode editar as permissões para funções vinculadas a serviço.

- Aplicativos em execução no Amazon EC2 — Você pode usar uma função do IAM para gerenciar credenciais temporárias para aplicativos que estão sendo executados em uma instância do EC2 e fazendo AWS CLI solicitações de API. AWS É preferível fazer isso a armazenar chaves de acesso na instância do EC2. Para atribuir uma AWS função a uma instância do EC2 e disponibilizá-la para todos os seus aplicativos, você cria um perfil de instância anexado à instância. Um perfil de instância contém a perfil e permite que os programas em execução na instância do EC2 obtenham credenciais temporárias. Para mais informações, consulte Usar uma função do IAM para conceder permissões a aplicativos em execução nas instâncias do Amazon EC2 no Guia do usuário do IAM.

Para aprender se deseja usar perfis do IAM, consulte Quando Criar uma Função do IAM (em Vez de um Usuário) no Guia do Usuário do IAM.

Gerenciando acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política é um objeto AWS que, quando associada a uma identidade ou recurso, define suas permissões. AWS avalia essas políticas quando um principal (usuário, usuário raiz ou sessão de função) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida

ou negada. A maioria das políticas é armazenada AWS como documentos JSON. Para obter mais informações sobre a estrutura e o conteúdo de documentos de políticas JSON, consulte Visão Geral das Políticas JSON no Guia do Usuário do IAM.

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.

Por padrão, usuários e funções não têm permissões. Para conceder aos usuários permissão para executar ações nos recursos de que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM às funções e os usuários podem assumir as funções.

As políticas do IAM definem permissões para uma ação, independente do método usado para executar a operação. Por exemplo, suponha que você tenha uma política que permite a ação `iam:GetRole`. Um usuário com essa política pode obter informações de função da AWS Management Console AWS CLI, da ou da AWS API.

Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário do IAM, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e funções podem realizar, em quais recursos e em quais condições. Para saber como criar uma política baseada em identidade, consulte Criar políticas do IAM no Guia do usuário do IAM.

As políticas baseadas em identidade também podem ser categorizadas como políticas em linha ou políticas gerenciadas. As políticas em linha são incorporadas diretamente a um único usuário, grupo ou função. As políticas gerenciadas são políticas autônomas que você pode associar a vários usuários, grupos e funções em seu Conta da AWS. As políticas AWS gerenciadas incluem políticas gerenciadas e políticas gerenciadas pelo cliente. Para saber como selecionar entre uma política gerenciada ou uma política em linha, consulte Selecionar 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. São exemplos de políticas baseadas em recursos as políticas de confiança de função do IAM e as políticas do bucket do Amazon S3. Em serviços que suportem políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para

o recurso ao qual a política está anexada, a política define quais ações uma entidade principal especificada pode executar nesse recurso e em que condições. Você deve especificar uma entidade principal em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Políticas baseadas em atributos são políticas em linha que estão localizadas nesse serviço. Você não pode usar políticas AWS gerenciadas do IAM em uma política baseada em recursos.

Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou funções da conta) têm permissão para acessar um recurso. As ACLs são semelhantes as políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

O Amazon S3 e o Amazon VPC são exemplos de serviços que oferecem suporte a ACLs. AWS WAF Saiba mais sobre ACLs em Configurações da lista de controle de acesso (ACL) no Guia do Desenvolvedor do Amazon Simple Storage Service.

Outros tipos de política

AWS oferece suporte a tipos de políticas adicionais menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- Limites de permissões: um limite de permissões é um recurso avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou perfil do IAM). É possível definir um limite de permissões para uma entidade. As permissões resultantes são a interseção das políticas baseadas em identidade de uma entidade e dos seus limites de permissões. As políticas baseadas em atributo que especificam o usuário ou o perfil no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações sobre limites de permissões, consulte Limites de Permissões para Entidades do IAM no Guia do Usuário do IAM.
- Políticas de controle de serviço (SCPs) — SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (OU) em. AWS Organizations AWS Organizations é um serviço para agrupar e gerenciar centralmente vários Contas da AWS que sua empresa possui. Se você habilitar todos os atributos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. O SCP limita as permissões para entidades nas contas dos membros, incluindo cada

uma Usuário raiz da conta da AWS. Para obter mais informações sobre o Organizações e SCPs, consulte Como os SCPs Funcionam no AWS Organizations Manual do Usuário do.

- Políticas de sessão: são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para uma função ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou do perfil e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em atributo. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações, 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

Atributo do IAM	Suporte ao Amazon Cognito
Políticas baseadas em identidade	Sim
Políticas baseadas em recursos	Não
Ações das políticas	Sim
atributos de políticas	Sim
Chaves de condição de políticas	Sim
ACLs	Não

Atributo do IAM	Suporte ao Amazon Cognito
ABAC (tags em políticas)	Parcial
Credenciais temporárias	Sim
Permissões de entidade principal	Não
Perfis de serviço	Sim
Funções vinculadas 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

Suporta com políticas baseadas em identidade	Sim

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário IAM, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e funções podem realizar, em quais recursos e em quais condições. Saiba como criar uma política baseada em identidade consultando Criando Políticas do IAM no Guia do Usuário do IAM.

Com as políticas baseadas em identidade do IAM, é possível especificar ações ou recursos permitidos ou negados, assim como as condições sob as quais as ações são permitidas ou negadas. Você não pode especificar a entidade principal em uma política baseada em identidade porque ela se aplica ao usuário ou função à qual ela está anexada. Para saber mais sobre todos os elementos que podem ser usados em uma política JSON, consulte Referência de elementos da 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

Oferece suporte a 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 função do IAM e as políticas do bucket do Amazon S3. Em serviços que suportem políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o recurso ao qual a política está anexada, a política define quais ações uma entidade principal especificada pode executar nesse recurso e em que condições. Você deve 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, você pode especificar uma conta inteira ou as entidades do IAM em outra conta como a entidade principal em uma política baseada em atributo. Adicionar uma entidade principal entre contas à política baseada em atributo é apenas metade da tarefa de estabelecimento da relação de confiança. Quando o principal e o recurso são diferentes Contas da AWS, um administrador do IAM na conta confiável também deve conceder permissão à entidade principal (usuário ou função) para acessar o recurso. Eles concedem permissão ao anexar uma política baseada em identidade para a entidade. No entanto, se uma política baseada em atributo conceder acesso a uma entidade principal na mesma conta, nenhuma política baseada em identidade adicional será necessária. Para obter mais informações, consulte Como os perfis do IAM diferem de políticas baseadas em recursos no Guia do usuário do IAM.

Ações de políticas para o Amazon Cognito

Oferece suporte a 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 você pode usar para permitir ou negar acesso em uma política. As ações de política geralmente têm o mesmo nome da operação de AWS API associada. Existem algumas exceções, como ações somente de permissão, que não têm

uma operação de API correspondente. Há também algumas operações que exigem várias ações em uma política. Essas ações adicionais são chamadas de ações dependentes.

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 instrução, separe-as com vírgulas.

```
"Action": [
      "cognito-identity:action1",
      "cognito-identity:action2"
         ]
```

APIs assinadas vs. APIs não assinadas

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 de consumidor para aplicações a serem disponibilizadas ao público, você tem acesso às APIs não assinadas a seguir. Sua aplicação faz essas solicitações de API aos seus usuários e usuários em potencial. Algumas APIs não exigem autorização prévia, como a `InitiateAuth`, destinada a iniciar uma nova sessão de autenticação. Algumas APIs usam tokens de acesso ou chaves de sessão para autorização, como a `VerifySoftwareToken`, destinada a concluir a configuração de MFA para um usuário que tenha 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 Operações de API autenticadas e não autenticadas de grupos de usuários do Amazon Cognito.

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

Oferece suporte a atributos 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 `Resource` de política JSON especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou um elemento `NotResource`. Como prática recomendada, especifique um recurso usando seu nome do recurso da Amazon (ARN). Isso pode ser feito para ações que oferecem suporte a um tipo de atributo específico, conhecido como permissões em nível de atributo.

Para ações não compatíveis com permissões no nível de recurso, como operações de listagem, use um curinga (*) para indicar que a instrução se aplica a todos os recursos.

```
"Resource": "*"
```

Nomes do recurso 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 mais informações sobre ARNs, consulte Identificadores do IAM.

```
arn:aws:cognito-identity:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID
```

ARNS para o 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 as APIs que operam em um grupo de identidades, o formato do ARN do grupo de identidades é o mesmo que para as identidades federadas do Amazon Cognito, exceto pelo fato de que o nome do serviço é `cognito-sync` em vez de `cognito-identity`:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID
```

Para as APIs que operam em uma única identidade, como o `RegisterDevice`, é possível consultar a identidade individual pelo seguinte formato de ARN:

```
arn:aws:cognito-sync:REGION:ACCOUNT_ID:identitypool/IDENTITY_POOL_ID/
identity/IDENTITY_ID
```

Para as APIs que operam em conjuntos de dados, como `UpdateRecords` e `ListRecords`, é possível consultar o conjunto de dados específico usando o seguinte formato de 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 do 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

Suporta 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 principal pode executar ações em quais recursos, e em que condições.

O elemento `Condition` (ou bloco `Condition`) permite especificar condições nas quais uma instrução estiver em vigor. O elemento `Condition` é opcional. Você pode criar expressões condicionais que usem operadores de condição, como “igual a” ou “menor que”, para corresponder a condição da política aos valores na solicitação.

Se você especificar vários elementos `Condition` em uma instrução ou várias chaves em um único `Condition` elemento, a AWS os avaliará usando uma operação lógica `AND`. Se você especificar vários valores para uma única chave de condição, AWS avalia a condição usando uma `OR` operação lógica. Todas as condições devem ser atendidas antes que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar condições. Por exemplo, é possível conceder a um usuário do IAM permissão para acessar um atributo somente se ele estiver marcado com seu nome de usuário do IAM. Para obter mais informações, consulte Elementos de Política do IAM: Variáveis e Tags no Guia do Usuário do IAM.

AWS suporta chaves de condição globais e chaves de condição específicas do serviço. 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

Oferece suporte a ACLs	Não

As listas de controle de acesso (ACLs) controlam quais entidades principais (membros, usuários ou funções da conta) têm permissões para acessar um recurso. As ACLs são semelhantes as políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

Controle de acesso por atributo (ABAC) com o Amazon Cognito

Oferece suporte a ABAC (tags em políticas)	Parcial

O controle de acesso baseado em recurso (ABAC) é uma estratégia de autorização que define permissões com base em recursos. Em AWS, esses atributos são chamados de tags. Você pode anexar tags a entidades do IAM (usuários ou funções) e a vários AWS recursos. A marcação de entidades e atributos é a primeira etapa do ABAC. Em seguida, você cria políticas de ABAC para permitir operações quando a tag da entidade principal corresponder à tag do recurso que ela estiver tentando acessar.

O ABAC é útil em ambientes que estão crescendo rapidamente e ajuda em situações em que o gerenciamento de políticas se torna um problema.

Para controlar o acesso baseado em tags, forneça informações sobre a tag no elemento de condição de uma política usando as chaves de condição `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* ou `aws:TagKeys`.

Se um serviço oferecer suporte às três chaves de condição para todo tipo de recurso, o valor será Sim para o serviço. Se um serviço oferecer suporte às três chaves de condição somente para alguns tipos de recursos, o valor será Parcial.

Para obter mais informações sobre o ABAC, consulte O que é ABAC? no Guia do Usuário do IAM. Para visualizar um tutorial com etapas para configurar o ABAC, consulte Usar Controle de Acesso Baseado em Atributos (ABAC) no Guia do Usuário do IAM.

Usar credenciais temporárias com o Amazon Cognito

Oferece suporte a credenciais temporárias	Sim

Alguns Serviços da AWS não funcionam quando você faz login usando credenciais temporárias. Para obter informações adicionais, incluindo quais Serviços da AWS funcionam com credenciais temporárias, consulte Serviços da AWS "Trabalhe com o IAM" no Guia do usuário do IAM.

Você está usando credenciais temporárias se fizer login AWS Management Console usando qualquer método, exceto um nome de usuário e senha. Por exemplo, quando você acessa AWS usando o link de login único (SSO) da sua empresa, esse processo cria automaticamente credenciais temporárias. Você também cria automaticamente credenciais temporárias quando faz login no console como usuário e, em seguida, alterna perfis. Para obter mais informações sobre como alternar funções, consulte Alternar para uma Função (Console) no Guia do Usuário do IAM.

Você pode criar manualmente credenciais temporárias usando a AWS API AWS CLI ou. Em seguida, você pode usar essas credenciais temporárias para acessar AWS. AWS recomenda que você gere credenciais temporárias dinamicamente em vez de usar chaves de acesso de longo prazo. Para mais informações, consulte Credenciais de segurança temporárias no IAM.

Permissões de entidade principal entre serviços para o Amazon Cognito

Suporte para o recurso Encaminhamento de sessões de acesso (FAS)	Não

Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.

Perfis de serviço para o Amazon Cognito

Suporta perfis de serviço	Sim

A função de serviço é uma função do IAM que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para

obter mais informações, consulte Criando 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 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

Oferece suporte a perfis vinculados ao serviço	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 pode editar as permissões para funções vinculadas a 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. Eles também não podem realizar tarefas usando a AWS API AWS Management Console, AWS Command Line Interface (AWS CLI) ou. Para conceder permissões de usuários para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis, e os usuários podem assumir os perfis.

Para saber como criar uma política baseada em identidade do IAM usando esses exemplos de documento de política JSON, consulte Criação de políticas do IAM no Guia do Usuário do IAM.

Para obter detalhes sobre ações e tipos de recurso definidos pelo Amazon Cognito, por exemplo, o formato dos ARNs para cada um dos tipos de recurso, consulte Ações, recursos e chaves de condição do Amazon Cognito Identity na Referência de autorização do serviço.

Tópicos

- Melhores práticas de política
- Como usar o console do Amazon Cognito
- Permitir que usuários visualizem suas próprias permissões
- Como restringir o acesso do console a um grupo específico de identidades
- Como permitir o acesso a um conjunto de dados específico para todas as identidades em um grupo

Melhores práticas 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 seus 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 obter mais informações, consulte Políticas Gerenciadas pela AWS ou AWS Políticas Gerenciadas 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 atributos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para obter mais informações sobre como usar o IAM para aplicar permissões, consulte Políticas e Permissões no IAM no Guia do Usuário do IAM.
- Utilize condições nas políticas do IAM para restringir ainda mais o acesso: você pode adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, você pode gravar 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 AWS

CloudFormation. Para obter mais informações, consulte Condição de Elementos de Política JSON do IAM no Guia do Usuário do IAM.

- Use o IAM Access Analyzer para validar suas políticas do IAM para garantir permissões seguras e funcionais: o IAM Access Analyzer valida as políticas novas e existentes para que elas sigam o idioma de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de 100 verificações de política e ações recomendadas para ajudar você a criar políticas seguras e funcionais. Para obter mais informações, consulte Validação de Política 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 obter mais informações, consulte Configurando Acesso à API Protegido por MFA no Guia do Usuário do IAM.

Para obter mais informações 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 Se você criar uma política baseada em identidade que seja mais restritiva do 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 a 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 mais informações, consulte Adicionando Permissões a um Usuário no Guia do Usuário do IAM.

Permitir que 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

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cognito-identity:ListIdentityPools"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cognito-identity:*"
      ],
      "Resource": "arn:aws:cognito-identity:us-east-1:0123456789:identitypool/us-
east-1:1a1a1a1a-ffff-1111-9999-12345678"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cognito-sync:*"
      ],
      "Resource": "arn:aws:cognito-sync:us-east-1:0123456789: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

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cognito-sync:ListRecords",
        "cognito-sync:UpdateRecords"
      ],
      "Resource": "arn:aws:cognito-sync:us-east-1:0123456789: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 outros usuários acessem o Amazon Cognito, é necessário criar uma entidade do IAM (usuário ou perfil) para a pessoa ou para a aplicação que precisa do acesso. Elas usarão as credenciais dessa entidade para acessar a AWS. Você deve anexar uma política à entidade que concede a elas as permissões corretas no Amazon Cognito.

Para começar a usar imediatamente, consulte Criar os primeiros usuário e grupo delegados pelo IAM no Guia do usuário do IAM.

Quero permitir que pessoas fora da minha AWS conta acessem meus recursos do Amazon Cognito

Você pode criar uma função que os usuários de outras contas ou pessoas fora da sua organização possam usar para acessar seus recursos. Você pode 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 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.
- Saiba como conceder acesso por meio da federação de identidades consultando Concedendo Acesso a Usuários Autenticados Externamente (Federação de Identidades) no Guia do Usuário do IAM.
- Para saber a diferença entre usar perfis e políticas baseadas em recursos para acesso entre contas, consulte Como os perfis do IAM diferem de políticas baseadas em recursos 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 funções vinculadas a serviços, consulte Serviços da AWS compatíveis com o IAM e procure os serviços que contenham Yes (Sim) na coluna Service-Linked Role (Função vinculada a serviço). Escolha um Sim com um link para visualizar a documentação do perfil vinculado a 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

A função vinculada ao serviço `AWSServiceRoleForAmazonCognitoIdpEmailService` confia nos seguintes serviços para aceitar 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 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 uma função vinculada a serviço. Para obter mais informações, consulte Permissões de perfil vinculado a serviços no Guia do usuário do IAM.

AWSServiceRoleForAmazonCognitoIdp

A função AWSServiceRoleForAmazonCognitoIdp 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 AWSServiceRoleForAmazonCognitoIdp

- 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 uma função vinculada a serviço. Quando você configura um grupo de usuários para usar sua configuração do Amazon SES para lidar com a entrega de e-mail na AWS Management Console, 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, você poderá 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 pode editar a descrição da função usando o IAM. Para ter mais informações, 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 recurso 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 AWS Management Console 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 AWS Management Console 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 a função vinculada a 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 obter mais informações, 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

Custos de monitoramento

O Amazon Cognito cobra pelas seguintes dimensões de seu uso.

- Pool de usuários: usuários ativos mensais (MAUs)
- MAUs do grupo de usuários conectados com a federação OIDC ou SAML
- MAUs em um grupo de usuários com recursos avançados de segurança
- Pool de usuários ativos, clientes de aplicativos e volume de solicitações para autorização máquina a máquina (M2M) com concessões de credenciais de cliente
- Uso comprado acima das cotas padrão para algumas categorias de APIs de grupos de usuários

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 os preços do Amazon Cognito.

Visualizando e antecipando custos

Você pode visualizar e relatar seus AWS custos no AWS Billing and Cost Management console. Você pode encontrar suas cobranças mais recentes do Amazon Cognito na seção Faturamento e pagamentos. Em Faturas, Cobranças por serviço, filtre `Cognito` para ver seu uso. Para obter mais informações, consulte Exibição da sua fatura no Guia do usuário do AWS Billing .

Para monitorar as taxas de solicitação de API, revise a métrica de utilização no console Service Quotas. Por exemplo, as solicitações de credenciais do cliente são exibidas como Taxa de

ClientAuthentication solicitações. Na sua fatura, essas solicitações estão associadas ao cliente do aplicativo que as produziu. Com essas informações, você pode alocar custos equitativamente aos locatários em uma arquitetura multilocatária.

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 da

O Amazon Cognito fatura com base na contagem de usuários, no uso de recursos e no volume de solicitações. A seguir estão algumas dicas para gerenciar custos no Amazon Cognito,

Não ative usuários inativos

As operações típicas que tornam um usuário ativo são login, inscrição e redefinição de senha. Para obter uma lista mais completa, consulteUsuários ativos mensalmente. O Amazon Cognito não contabiliza usuários inativos em sua fatura. Evite qualquer operação que deixe um usuário ativo. Em vez da operação AdminGetUserda API, consulte os usuários com a ListUsersoperação. Não realize testes administrativos de alto volume de operações de grupos de usuários com usuários inativos.

Vincular usuários federados

Os usuários que fazem login com um provedor de identidade SAML 2.0 ou OpenID Connect (OIDC) têm um custo maior do que os 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.

Gerenciar taxas de solicitação

Se seu grupo de usuários estiver se aproximando do limite superior de sua cota, considere comprar capacidade adicional para lidar com o volume. Talvez você consiga reduzir o volume de solicitações em seu aplicativo. Para ter mais informações, consulte Otimize 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 do cliente 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 sua fatura. Para otimizar o custo, inclua configurações de expiração e tratamento de tokens no design de seus aplicativos.

- Armazene os tokens de acesso em cache para que, quando seu aplicativo solicitar um novo token, ele receba uma versão em cache de um token emitido anteriormente. Quando você implementa esse método, seu proxy de cache age como uma proteção contra aplicativos que solicitam tokens de acesso sem saber da expiração dos tokens adquiridos anteriormente. O armazenamento em cache de tokens é ideal para microsserviços de curta duração, como funções Lambda e contêineres Docker.
- Implemente mecanismos de tratamento de tokens em seus aplicativos que levem em conta a expiração do token. Não solicite um novo token até que os tokens anteriores tenham expirado. Avalie as necessidades de confidencialidade e disponibilidade de cada aplicativo e configure o cliente do aplicativo do pool de usuários para emitir tokens de acesso com um período de validade apropriado. A duração personalizada do token funciona melhor com APIs e servidores de longa duração que podem gerenciar persistentemente a frequência das solicitações de credenciais.

Excluir credenciais de cliente não utilizadas (clientes de aplicativos)

A autorização M2M é cobrada com base em dois fatores: a taxa de solicitações de token e o número de clientes de aplicativos que concedem credenciais de clientes. Quando os clientes de aplicativos para autorização M2M não estiverem em uso, exclua-os ou remova a autorização para emitir as credenciais do cliente. Para obter mais informações sobre como gerenciar a configuração do cliente do aplicativo, consulteClientes de aplicações de grupos de usuários.

Gerencie a segurança avançada

Quando você configura recursos de segurança avançados em um grupo de usuários, a taxa de cobrança de segurança avançada se aplica a todos os MAUs no grupo de usuários. Se você tiver usuários que não precisam de recursos avançados de segurança, separe-os em outro grupo de usuários.

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 os 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 os 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 CloudWatch observabilidade entre contas.

Tópicos

- Registro em log de atividades adicionais dos grupos de usuários do Amazon Cognito
- Métricas para grupos de usuários do Amazon Cognito
- Dimensões dos grupos de usuários do Amazon Cognito
- Usar o console do Service Quotas para rastrear métricas
- Use o CloudWatch console para monitorar métricas
- Crie um CloudWatch alarme para uma cota

Registro em log de atividades adicionais dos grupos de usuários do Amazon Cognito

Você pode configurar seu grupo de usuários para enviar registros detalhados de alguma atividade adicional para um grupo de CloudWatch registros. Esses registros têm uma granularidade mais fina do que os existentes e podem ser úteis para solucionar problemas em AWS CloudTrail seu 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 atividades 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.

No momento, você só pode entregar logs de notificação de usuário no nível de Erro do grupo de usuários.

Logs detalhados 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 trabalhos de importação de usuários e gatilhos Lambda em Logs. CloudWatch O Amazon Cognito e o Lambda armazenam esses logs em grupos de logs diferentes daqueles que você especifica para logs de atividades detalhados.

Você pode configurar registros de atividades 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.

Você deve autorizar essas solicitações com AWS credenciais que tenham as seguintes permissões.

```
{
    "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"
    }
}
```

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 registros podem afetar a entrega de registros.

CloudWatch As cobranças de registros se aplicam quando a entrega de registros está ativada. Para obter mais informações, consulte Vended Logs nos CloudWatch preços da Amazon.

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.

Métricas para grupos de usuários do Amazon Cognito

A tabela a seguir lista as métricas disponíveis para grupos de usuários do Amazon Cognito. O namespace de métricas Amazon CloudWatch para o Amazon Cognito é `AWS/Cognito`. Para obter mais informações, consulte Namespaces no Guia CloudWatch do usuário da Amazon.

> **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	Descrição
`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

Métrica	Descrição
	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 de registro de usuário bem-sucedidas, 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 solicitações de registro de usuário bem-sucedidas, use a estatística `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`, `UserPoolClient` Unidades: contagem

Métrica	Descrição
`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 solicitaç ão 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`, `UserPoolC lient` Unidades: contagem

Métrica	Descrição
`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 autenticação é emitido para o usuário. Uma autenticação bem-sucedida produz um valor de 1, enquanto uma solicitação malsucedida produz um valor de 0. Uma solicitação com controle de utilizaçã o também é considerada uma solicitação malsucedida e, portanto, também produzirá uma contagem de 0. Para descobrir a porcentagem de solicitaç ões de autenticação de usuário bem-suced idas, use a estatística `Average` nessa métrica. Para contar o número total de solicitações de autenticação de usuário, use a estatística `Sample Count` nessa métrica. Para contar o número total de solicitações de autenticação de usuário bem-sucedidas, use a estatística `Sum` nessa métrica. Para contar o número total de 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.

Métrica	Descrição
	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	Descrição
`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 autenticação 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 estatísti cas de token `Refresh`. Dimensão da métrica: `UserPool`, `UserPoolC lient` Unidades: contagem

Métrica	Descrição
`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 utilizaçã o também é considerada uma solicitação malsucedida e, portanto, também produzirá uma contagem de 0. Para descobrir a porcentagem de solicitaç ões 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-suced idas de atualização de um token do Amazon Cognito, use a estatística `Sum` nessa métrica. Para contar o número total de solicitações 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`.

Métrica	Descrição
	Dimensão da métrica: `UserPool`, `UserPoolClient` Unidades: contagem
`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` Unidades: contagem

Métrica	Descrição
`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 autorizaç ão, mas nenhum token, produzem um valor de 0. Para descobrir a porcentagem de solicitaç ões de federação de identidades bem-suced idas, 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 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`, `UserPoolC lient` , `IdentityProvider` Unidades: contagem

Métrica	Descrição
`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

Métrica	Descrição
`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 utilização, chamadas com falha e chamadas bem-suced idas. Essa métrica está disponível no Usage (Uso) `nameSpace` . 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

Métrica	Descrição
`ThrottleCount`	Fornece o número total de chamadas com controle de utilização relacionadas a uma categoria. Essa métrica está disponível no Usage (Uso) `nameSpace` . 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

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`.
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.

Dimensão	Descrição
Classe	A classe do recurso sob acompanhamento. O Amazon Cognito não usa a dimensão de classe.

Usar o console do Service Quotas para rastrear métricas

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.

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 de 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 de AWS serviço e às APIs. `CallCount` `ThrottleCount` 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

Registro de chamadas da API do Amazon Cognito com 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 sobre 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 CloudWatch para acionar um alarme se a configuração de um grupo de identidades for alterada. Para obter mais informações, consulte Criação de CloudWatch alarmes para CloudTrail eventos: exemplos.

Tópicos

- Informações do Amazon Cognito em CloudTrail
- Compreender os eventos de login do Amazon Cognito
- Análise de CloudTrail eventos do Amazon Cognito com o Amazon Logs Insights CloudWatch

Informações do Amazon Cognito em 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, ao criar uma trilha no console, a mesma é aplicada a todas as Regiões. 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 para um perfil ou usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte o elemento CloudTrail userIdentity.

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_FOR_SECURITY_REASONS` Para ver exemplos de campos que o Amazon Cognito não preenche para eventos, consulte Compreender os eventos de login 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.

Grupos de usuários do Amazon Cognito

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 o OpenID Connect, SAML 2.0 ou a interface de usuário hospedada. 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 de UI hospedada em sua UI hospedada como eventos em. CloudTrail

Operações de interface de usuário hospedadas em CloudTrail

Operation	Descrição
`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.
`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` .

Operation	Descrição
`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 do UserInfo.
`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 recursos de segurança avançada.

> **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.

Banco de identidades do Amazon Cognito

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 se CloudTrail registra, 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.

Compreender os eventos de login do Amazon Cognito

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
- Exemplo de CloudTrail evento para uma solicitação SAML
- Exemplos de CloudTrail eventos para solicitações ao endpoint do token
- Exemplo de CloudTrail evento para CreateIdentityPool

- Exemplo de CloudTrail evento para GetCredentialsForIdentity
- Exemplo de CloudTrail evento para GetId
- Exemplo de CloudTrail evento para GetOpenIdToken
- Exemplo de CloudTrail evento para GetOpenIdTokenForDeveloperIdentity
- Exemplo de CloudTrail evento para UnlinkIdentity

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 inscreve 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.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "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.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "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.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "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.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "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.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "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.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "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.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "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.us-west-2.amazoncognito.com",
        "userPoolId": "us-west-2_aaaaaaaaa"
    },
    "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"
}
```

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
```

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— 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 mais altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de datacenter tradicionais.

Para obter mais informações sobre AWS regiões e zonas de disponibilidade, consulte infraestrutura AWS global.

Tópicos

- Fatores em relação a dados regionais

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 ser compatíveis 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, suporta esses modos.

Além disso, as solicitações devem ser assinadas utilizando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou é possível usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

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 atributos. Esse tipo de atualização afeta todas as identidades (usuários, grupos e perfis) em que a política está anexada.

É mais provável que os serviços atualizem uma política gerenciada pela AWS quando um novo atributo 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.

Várias políticas estão disponíveis por meio do Console do IAM que você pode usar para conceder 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, por isso, mesmo que novas APIs sejam adicionadas, os usuários continuarão a ter 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.

Os grupos de identidades com acesso de convidado não autenticado aplicam uma política AWS gerenciada adicional,`AmazonCognitoUnAuthedIdentitiesSessionPolicy`, como uma 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 ter mais informações, consulte Perfis do IAM.

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
`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` : atualizações em políticas existentes	Foram adicionadas novas permissões para permitir que usuários avançados visualizem e gerenciem associações de ACLs da AWS WAF web com grupos de usuários do Amazon Cognito. Foram adicionadas novas permissões para permitir que usuários somente para leitura visualizem associações de ACLs da AWS WAF web com grupos de usuários do Amazon Cognito.	19 de julho de 2022

Alteração	Descrição	Data
`AmazonCognitoPowerUser` : atualização para uma política existente	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` : atualizar para uma política existente	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.	1º de março de 2021

Como marcar recursos do Amazon Cognito

Uma etiqueta é um rótulo de metadados que você ou a AWS atribui a um recurso da AWS. Cada tag consiste em uma chave e um valor. Para tags atribuídas por você, é possível definir a chave e o valor. Por exemplo, talvez você defina a chave como `stage` e o valor de recurso como `test`.

As tags ajudam você a fazer o seguinte:

- Identificar e organizar seus recursos da AWS. Muitos serviços da AWS oferecem suporte à marcação para que você possa atribuir a mesma etiqueta a diferentes recursos de serviços. 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.
- Monitorar seus custos da AWS. É possível ativar essas etiquetas no painel do AWS Billing and Cost Management. A AWS usa as etiquetas de alocação de custo para categorizar seus custos e entregar um relatório mensal de alocação de custos a 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 AWS Command Line Interface ou a API do Amazon Cognito para adicionar, editar ou excluir etiquetas para grupos de usuários e de 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

- Grupos 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: _ . : / = + - @
- As chaves e os valores diferenciam letras 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 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. Selecione a guia User pool properties (Propriedades do grupo de usuários) e localize Tags (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).

Na página Manage tags (Gerenciar etiquetas), também é possível editar as chaves e os valores das etiquetas existentes. Para remover uma tag, selecione Remove (Remover).

Exemplos do AWS CLI

A AWS CLI disponibiliza comandos que ajudam a gerenciar as etiquetas atribuídas aos seus grupos de usuários e 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 impõe uma taxa máxima de solicitação para operações de API. Para obter mais informações sobre as operações de API que o Amazon Cognito disponibiliza, consulte Referências de API e endpoint do Amazon Cognito. Para 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 do grupo de usuários por categoria, consulteCategorias de operação da API de grupos de usuários do Amazon Cognito e cotas de taxa de solicitação.

No console Service Quotas, você pode monitorar o uso da cota por categorias de grupos de usuários e grupos de identidades. Se a taxa de solicitação dos grupos de usuários do Amazon Cognito exceder ou exceder uma cota, você poderá comprar capacidade adicional. Você pode acompanhar o uso da cota do grupo de usuários por categoria e os aumentos da cota de compra no console 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 `UserAuthentication` categoria inclui quatro operações na API de grupos de usuários do Amazon Cognito:`AdminInitiateAuth`, `InitiateAuthAdminRespondToAuthChallenge`, 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 em seus aplicativos. A cota é suficiente para abranger a grande maioria dos casos

de uso. Depois que seu aplicativo responde até três aos desafios de autenticação, solicitações adicionais contam para a cota da `UserAuthentication` categoria. 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 `UserAuthentication` categoria for 80 RPS, você poderá ligar `RespondToAuthChallenge` ou com uma `AdminRespondToAuthChallenge` taxa de até 240 RPS (3 * 80 RPS). Se seu grupo de usuários solicitar quatro rodadas de desafio por autenticação e 70 usuários entrarem por segundo, o total `RespondToAuthChallenge` será 280 RPS (70 x 4), 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 a cota da categoria definida em 80 RPS por 30 RPS, o Amazon Cognito limita as solicitações do seu aplicativo.

Usuários ativos mensalmente

Quando o Amazon Cognito calcula o faturamento 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. As atividades que tornam um usuário ativo incluem as seguintes:

- Inscrição ou criação administrativa de um usuário
- Fazer login
- Sair
- Confirmação da conta do usuário ou verificação de atributos
- Redefinição de senhas
- Alteração de atributos, associação do grupo ou preferências de MFA
- Consulta de atributos detalhados de um usuário
- Ativação, desativação ou exclusão do usuário

> **Note**
>
> A categoria Atributos detalhados de consulta de um usuário 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

cobranças excessivas, colete dados do usuário com `ListUsers` ou armazene informações do usuário em um banco de dados externo.

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`,, ou `UserCreationAccountRecovery`, talvez seja necessário aumentar as cotas para outras. 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.

Otimize 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 aplicativo 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. Os SDKs da Amazon têm lógica de repetição de tentativa 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.

JWTs em cache

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, consulteArmazenar tokens em cache. 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 da

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 são integrados ao Service Quotas, uma interface de console para exibir e gerenciar o uso da sua 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 algum tempo, você poderá ver um gráfico da utilização de seus recursos.

A coluna Valor da cota aplicada em nível de conta no console de Cotas de Serviços para grupos de usuários e grupos de identidade 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 um determinado período. Com esse conhecimento, você pode calcular como um aumento nos usuários ativos de seus aplicativos afetará suas cotas em seu modelo de uso. Por exemplo, imagine que seus aplicativos combinados no Oeste dos EUA (Oregon) resultaram em 2 milhões de usuários ativos em um mês e sua `UserAuthentication` categoria recebeu erros ocasionais de limitação na cota padrão de 120 solicitações por segundo (RPS). No mês anterior, antes de sua campanha publicitária bem-sucedida, você tinha 1 milhão de 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 seus MAUs

Acesse o AWS Billing console e analise uma fatura recente. Em cobranças por serviço, você pode filtrar no Cognito para ver um detalhamento dos seus MAUs para esse 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 nos grupos de usuários e grupos de identidades em cada um. Região da AWS Você pode comprar um aumento nas cotas de taxa de solicitação da API de grupos de usuários do Amazon Cognito ajustáveis. Verifique sua cota atual e compre um aumento no console Service Quotas ou com as operações da API Service Quotas e. `ListAWSDefaultServiceQuotas` `RequestServiceQuotaIncrease`

- Para comprar um aumento de cota usando o console de Quotas de Serviço, consulte Solicitando um aumento de cota de API no Guia do Usuário de Quotas de Serviço.
- 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 maior capacidade de cota. Para obter preços de aumento de cota, consulte os preços 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 ter 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 um. Conta da AWS

> **Note**
>
> A cota para cada categoria é medida em usuários ativos mensais (MAUs). Contas da AWS com menos de dois milhões de MAUs podem operar dentro da cota padrão. Se você tiver menos de um milhão de MAUs e o Amazon Cognito estiver limitando as solicitações, considere otimizar seu aplicativo. Para ter mais informações, consulte Otimize 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`	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	Descrição	Cota padrão (RPS)	Ajustável
`UserAuthentication` • InitiateAuth • Atualização do token com `InitiateAuth` ou Endpoint de token • RespondToAuthChallenge[1] • AdminInitiateAuth	Operações que autenticam (fazem login) um usuário. Essas operações estão sujeitas a Operações de API de grupos de usuários do Amazon Cognito com processamento especial de taxa de solicitação .	120	Sim

Categoria	Descrição	Cota padrão (RPS)	Ajustável
• AdminRespondToAuthChallenge[1] • Login de interface de usuário hospedada e MFA em concessões implícitas ou código de autorização[2]			
`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
`UserFederation` Operações que federam (autenticam) usuários com um provedor de identidade 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

Categoria	Descrição	Cota padrão (RPS)	Ajustável
`UserAccountRecovery` • ChangePassword • ConfirmForgotPassword • ForgotPassword • AdminResetUserPassword • AdminSetUserPassword • RespondToAuthChallenge[1] • AdminRespondToAuthChallenge[1] • Redefinição de senha da interface de usuário hospedada	Operações que recuperam a conta do usuário ou alteram ou atualizam a senha do usuário.	30	Não
`UserRead` • AdminGetUser • GetUser	Operações que recuperam um usuário dos grupos de usuários.	120	Sim

Categoria	Descrição	Cota padrão (RPS)	Ajustável
`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
`UserToken` • RevokeToken	Operações para gerenciamento de tokens	120	Sim

Categoria	Descrição	Cota padrão (RPS)	Ajustável
`UserResourceRead` • AdminGetDevice • AdminListGroupsForUser • AdminListDevices • GetDevice • ListDevices • GetUserAttributeVerificationCode • ResendConfirmationCode • AdminListUserAuthEvents	As operações que recuperam informações de recursos de usuários do Amazon Cognito, como um dispositivo lembrado ou uma associação de grupo.	50	Sim

Categoria	Descrição	Cota padrão (RPS)	Ajustável
`UserResourceUpdate` • AdminForgetDevice • AdminUpdateAuthEventFeedback • AdminSetUserPreferência de MFAP • AdminSetUserSettings • AdminUpdateDeviceStatus • UpdateDeviceStatus • UpdateAuthEventFeedback • ConfirmDevice • SetUserPreferência de MFAP • 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	Descrição	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	Descrição	Cota padrão (RPS)	Ajustável
`UserPoolResourceRead` • DescribeIdentityProvider • DescribeResourceServer • DescribeUserImportJob • DescribeUserPoolDomain • GetCSVHeader • GetGroup • GetSigningCertificate • GetIdentityProviderByIdentifier • GetUserPoolMfaConfig • ListGroups • ListIdentityProviders • ListResourceServers • ListTagsForResource • ListUserImportJobs • DescribeRiskConfiguration • GetUICustomization	Operações que recuperam informações sobre recursos, como grupos ou servidores de recursos, de um grupo de usuários.[3]	20	Não

Categoria	Descrição	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	Descrição	Cota padrão (RPS)	Ajustável
• SetRiskConfiguration • SetUICustomization • TagResource • UntagResource			
`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
`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] A `RespondToAuthChallenge` ou `AdminRespondToAuthChallenge` resposta com um `ChallengeName` de `NEW_PASSWORD_REQUIRED` conta para a `UserAccountRecovery` categoria. Todas as outras respostas do desafio contam para a `UserAuthentication` categoria.

[2] Cada operação de interface de usuário hospedada 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 proporção da sua cota na categoria. `UserAuthentication`

[3] Qualquer operação individual nessa categoria tem uma restrição que impede que a operação seja chamada a uma taxa superior a 5 RPS para um único grupo de usuários.

Cotas de taxa de solicitação de operação da API de grupos de identidades (identidades federadas) do Amazon Cognito

Operation	Descrição	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.
`GetCredentialsForIdentity`	Recupere AWS credenciais de um grupo de identidades no fluxo de trabalho aprimorado.	200	Sim	Entre em contato com a equipe da conta.
`GetOpenIdTokenForD`	Recupere um token OpenID de um grupo de	50	Sim	Entre em contato com a equipe da conta.

Operation	Descrição	Cota padrão (RPS)[1]	Ajustável	Elegibilidade para aumento de cota
`eveloperIdentity`	identidades no fluxo de trabalho de desenvolvedor.			
`ListIdentities`	Recupere uma lista de IDs de identidade em um banco 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.
`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

Recurso	Cota	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

Recurso	Cota	Ajustável	Cota máxima
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.
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
Caracteres no assunto do e-mail	140	Não	N/D
Caracteres na mensagem de e-mail	20.000	Não	N/D

Recurso	Cota	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
Identificadores por provedor de identidade	50	Não	N/D
Identidades vinculadas a um usuário	5	Não	N/D
URLs de retorno de chamada por cliente da aplicação	100	Não	N/D
URLs de saída por cliente da aplicação	100	Não	N/D
Escopos por servidor de recursos	100	Não	N/D
Escopos por cliente da aplicação	50	Não	N/D
Domínios personalizados por conta	4	Não	N/D
Grupos aos quais cada usuário pode pertencer	100	Não	N/D

Recurso	Cota	Ajustável	Cota máxima
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 declarações existentes e adicionadas, além dos escopos nos tokens de acesso e identidade, 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. Para ter 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

Token	Cota
Token de ID	5 minutos – 1 dia
Token de atualização	1 hora – 3.650 dias
Token de acesso	5 minutos – 1 dia
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

Recurso	Cota
Período de validade do código de confirmação de cadastro	24 horas

Recurso	Cota
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
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 ter mais informações, consulte Comportamento para esquecimento da senha.

Cotas de recursos de trabalho de importação de usuários do grupo de usuários do Amazon Cognito

Recurso	Cota	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
Número máximo de usuários por arquivo CSV	500.000	Não	N/D

Cotas de recursos de grupos de identidades do Amazon Cognito (identidades federadas)

Recurso	Cota	Ajustável	Cota máxima
Grupos 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

Recurso	Cota	Ajustável	Cota máxima
Identidades por grupo de identidades	Ilimitado	Não	N/D
Provedores de identidade para os quais podem ser especificados mapeamentos de função	10	Não	N/D
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

Recurso	Cota	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

Recurso	Cota	Ajustável	Cota máxima
Tempo de espera para uma publicação em massa após uma solicitação bem-sucedida	24 horas	Não	N/D

Referências de API e endpoint do Amazon Cognito

As referências a seguir descrevem os endpoints de serviço para cada recurso do Amazon Cognito. Os grupos de usuários do Amazon Cognito têm as seguintes opções: endpoints de grupos de usuários com um domínio de grupo de usuários e a API de grupos de usuários. Para ter um detalhamento das classes de operações de API com a API de grupos de usuários do Amazon Cognito, consulte Usar a API de grupos de usuários e endpoints de grupo de usuários do Amazon Cognito.

Para ter uma lista dos endpoints de serviço para a API de grupos de usuários por Região da AWS, consulte Endpoints de serviço na Referência geral da AWS.

Tópicos

- Referência da interface do usuário hospedada e endpoints de federação do grupo de usuários
- Referência de API de grupos de usuários do Amazon Cognito
- Referência de API de grupos de identidades (identidades federadas) do Amazon Cognito
- Referência de API do Amazon Cognito Sync

Referência da interface do usuário hospedada e endpoints de federação do grupo de usuários

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. Elas incluem a interface do usuário hospedada, na qual os usuários podem se cadastrar e fazer login (o Endpoint de login) e sair (o Endpoint de logout). Para obter mais informações sobre esses recursos, consulte Configurar e usar a interface de usuário hospedada e endpoints de federação do Amazon Cognito.

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) e OAuth 2.0 de terceiros (). IdPs Para fazer login de um usuário com um provedor de identidades federadas, os usuários devem iniciar uma solicitação para o Endpoint de login da interface do usuário hospedada interativa ou o Autorizar endpoint do OIDC. O endpoint Authorize redireciona seus usuários para sua interface do usuário hospedada 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 da interface de usuário hospedada e dos endpoints de federação, o Amazon Cognito se integra aos SDKs para Android, iOS JavaScript e muito mais. Os SDKs fornecem ferramentas para realizar operações de API do grupo de usuários com os endpoints de serviço da API do 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 é compatível com os certificados do Amazon Cognito são alternadas e se renovam dinamicamente. Quando você fixa seu aplicativo em um certificado intermediário ou intermediário, 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

- Referência de endpoints da interface do usuário hospedada
- Referência de endpoints de federação do OAuth 2.0, do OpenID Connect e do OAuth 2.0
- Concessões do OAuth 2.0
- Usando o PKCE em concessões de código de autorização com grupos de usuários do Amazon Cognito
- Respostas de erro de federação e da interface do usuário hospedada

Referência de endpoints da interface do usuário hospedada

O Amazon Cognito ativa os endpoints da interface do usuário hospedada nesta seção quando você adiciona um domínio ao grupo de usuários. São páginas da Web nas quais seus 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. Para ter mais informações sobre a experiência do usuário na interface do usuário hospedada, consulte Cadastrar-se e fazer login com a UI hospedada.

As páginas da web que compõem a interface de usuário hospedada são uma aplicação web de front-end para sessões interativas de usuários com os clientes. A aplicação deve invocar a interface de usuário hospedada nos navegadores dos usuários. O Amazon Cognito não permite o acesso programático às páginas da web deste capítulo. Esses endpoints de federação no Referência de endpoints de federação do OAuth 2.0, do OpenID Connect e do OAuth 2.0 que retornam uma resposta JSON podem ser consultados diretamente no código da aplicação. Os redirecionamentos de Autorizar endpoint para a interface hospedada ou para uma página de login do IdP e também devem ser abertos nos navegadores dos usuários.

Os tópicos deste guia descrevem detalhadamente os endpoints da interface do usuário hospedada usados com frequência. O Amazon Cognito disponibiliza as páginas da web a seguir quando você atribui um domínio ao grupo de usuários.

Endpoints da interface do usuário hospedada

URL do endpoint	Descrição	Como é acessado
https://*O domínio do grupo de usuários*/login	Faz login no grupo de usuários locais e federados.	Redirecione de endpoints como Autorizar endpoint, `/logout` e `/confirmforgotPassword`. Consulte Endpoint de login.
https://*O domínio do grupo de usuários*/logout	Desconecta os usuários do grupo de usuários.	Link direto. Consulte Endpoint de logout.
https://*o domínio de seu grupo de usuários*/confirmUser	Confirma os usuários que selecionaram um link de e-mail para confirmar a respectiva conta de usuário.	O usuário selecionou um link em uma mensagem de e-mail.
https://*o domínio de seu grupo de usuários*/signup	Inscreve um novo usuário. A página `/login` direciona o usuário para `/signup`	Link direto com os mesmos parâmetros do `/oauth2/authorize`.

URL do endpoint	Descrição	Como é acessado
	quando ele seleciona Sign up (Cadastrar-se).	
https://*o domínio do grupo de usuários*/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://*o domínio do grupo de usuários*/forgotPassword	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 password? (Esqueceu a senha?).	1. Do link Esqueci minha senha em `/login`. 2. Link direto com os mesmos parâmetros do `/oauth2/authorize` .
https://*o domínio do grupo de usuários*/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://*o domínio do grupo de usuários*/resendcode	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`.

Tópicos

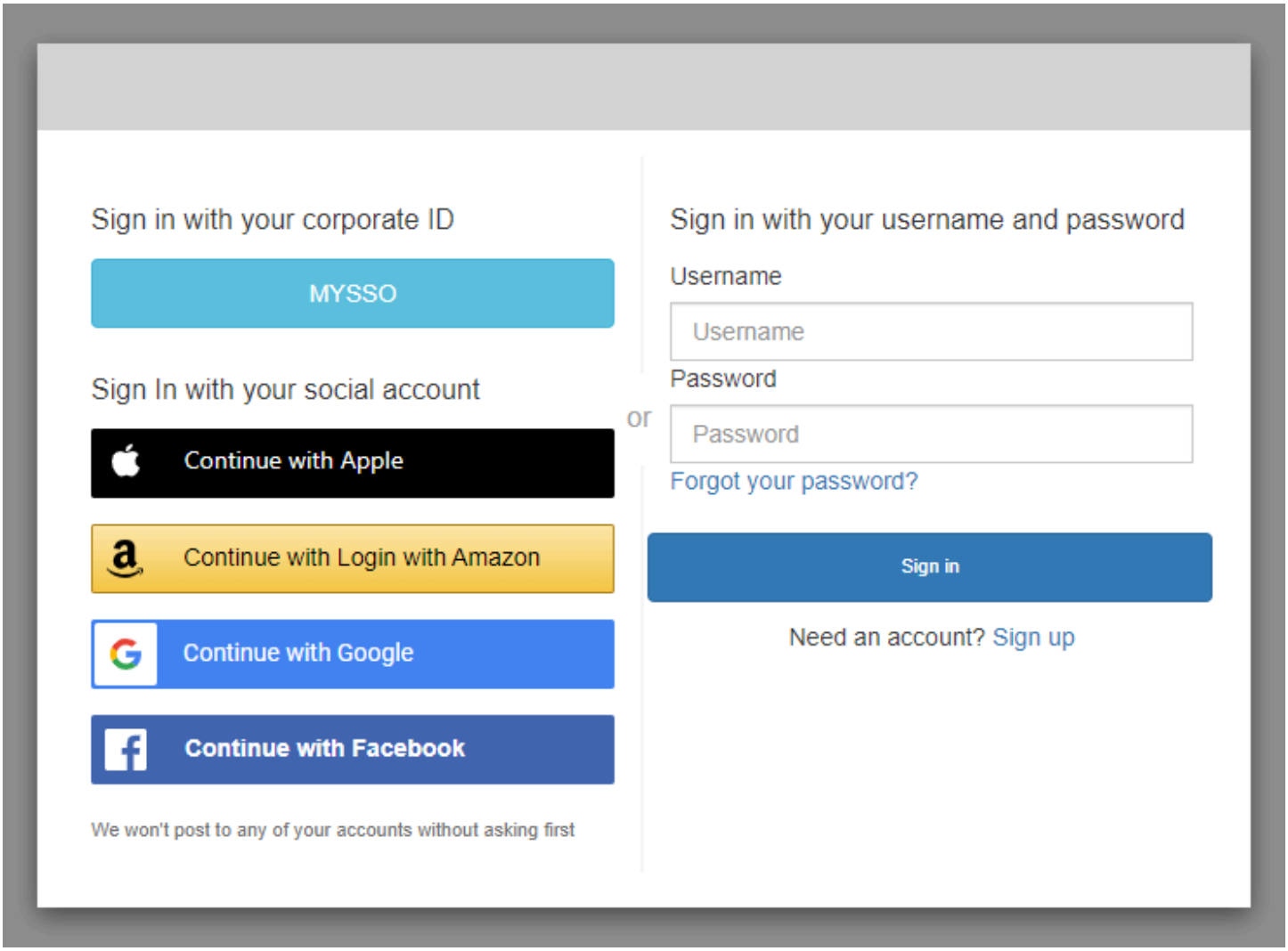
- Endpoint de login

- Endpoint de logout

Endpoint de login

O endpoint de login é um servidor de autenticação e um destino de redirecionamento de Autorizar endpoint. É o ponto de entrada para a interface de usuário hospedada quando você não especifica um provedor de identidades. 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 da interface de usuário hospedada. Na aplicação, invoque a federação e as páginas da interface de usuário hospedada 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 para a `/login` partir doAutorizar endpoint, ele transmite todos os parâmetros que você forneceu na sua 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.

Endpoint de logout

O `/logout` é um endpoint de redirecionamento. Ele desconecta o usuário e redireciona para uma URL de desconexão autorizada do seu cliente de aplicativo ou para o endpoint. `/login` Os parâmetros disponíveis em uma solicitação GET para o endpoint `/logout` são personalizados para casos de uso de interface de usuário hospedada pelo Amazon Cognito.

Para redirecionar o usuário e possibilitar que ela faça login novamente na interface do usuário hospedada, 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`.

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 interface de usuário hospedados nos navegadores dos usuários.

Para redirecionar o usuário para uma página selecionada, adicione URLs de saída permitidos ao cliente da aplicação. Nas solicitações dos usuários para o endpoint `logout`, adicione parâmetros

`logout_uri` e `client_id`. Se o valor de `logout_uri` for um dos URLs de saída permitidos para o cliente da aplicação, o Amazon Cognito redirecionará os usuários para esse 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 seu usuário de volta para, o Amazon `saml2/logout` Cognito responde com mais um redirecionamento para ou de sua solicitação. `redirect_uri` `logout_uri` Para ter mais informações, consulte Fluxo de saída do SAML.

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 ter mais informações, consulte Clientes de aplicações de grupos de usuários.

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 ter mais informações, consulte Clientes de aplicações de grupos de usuários.

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.

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 seu endpoint `/login`.

O Amazon Cognito requer um parâmetro logout_uri ou redirect_uri em sua solicitação para o endpoint `/logout`. Para redirecionar seu usuário ao seu *`/login`* endpoint para reautenticar e passar tokens para seu aplicativo, adicione um parâmetro redirect_uri. Se os parâmetros logout_uri e redirect_uri estiverem incluídos na sua solicitação para o endpoint, o `/logout` Amazon Cognito substituirá o parâmetro redirect_uri e processará exclusivamente o parâmetro logout_uri.

response_type

A resposta OAuth 2.0 que você deseja receber do Amazon Cognito depois que o usuário fizer login. `code` e `token` são os valores válidos para o parâmetro response_type.

Obrigatório quando você usa um parâmetro redirect_uri.

estado

Quando seu aplicativo adiciona um parâmetro de estado a uma solicitação, o Amazon Cognito retorna seu valor ao seu aplicativo quando o `/oauth2/logout` endpoint redireciona seu 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 passar uma string que corresponda a esse formato em um `state` parâmetro, codifique a string em base64 e decodifique-a em seu aplicativo.

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.

Exemplos de solicitações

Exemplo — sair e redirecionar o usuário para o cliente

Com exceção de `logout_uri` e`client_id`, todos os parâmetros de consulta possíveis para esse endpoint são passados para o. Autorizar endpoint 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 — saia e solicite 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 da interface de usuário hospedada. O endpoint de logout anexa os parâmetros em sua solicitação original ao destino do redirecionamento. O parâmetro `redirect_uri` em uma solicitação para o endpoint de logout não é um URL de logout, mas um URL de login que você deseja passar para o endpoint de autorização.

Veja a seguir um exemplo de solicitação que desconecta um usuário, redireciona para a página de login e fornece um código de autorização após o login. `https://www.example.com`

```
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
```

Referência de endpoints de federação do OAuth 2.0, do OpenID Connect e do OAuth 2.0

O Amazon Cognito ativa os endpoints nesta seção quando você adiciona um domínio ao grupo de usuários. Os endpoints da federação não são interativos com o usuário. Eles desempenham uma função de serviço para que seu aplicativo se comunique com provedores de identidade OAuth 2.0, OIDC e SAML 2.0 de terceiros (). IdPs

Os tópicos deste guia descrevem alguns endpoints OAuth 2.0 e OIDC usados com frequência. O Amazon Cognito cria os endpoints a seguir quando você atribui um domínio ao grupo de usuários.

Endpoints de federação do grupo de usuários

URL do endpoint	Descrição	Como é acessado
https://*0 domínio do grupo de usuários*/oauth2/authorize	Redireciona um usuário para a interface do usuário hospedada ou para fazer login com seu IdP.	Invocado no navegador do cliente para iniciar a autenticação do usuário. Consulte Autorizar endpoint.
https://*0 domínio do grupo de usuários*/oauth2/token	Retorna tokens com base em um código de autorização ou solicitação de credenciais do cliente.	Solicitado pelo aplicativo para recuperar tokens. Consulte Endpoint de token.
https://*0 domínio do grupo de usuários*/oauth2/userInfo	Retorna atributos do usuário com base nos escopos do OAuth 2.0 e na identidade do usuário em um token de acesso.	Solicitado pelo aplicativo para recuperar o perfil do usuário. Consulte Endpoint do UserInfo.

URL do endpoint	Descrição	Como é acessado
https://*O domínio do grupo de usuários*/oauth2/revoke	Revoga um token de atualização e os tokens de acesso associados.	Solicitado pelo aplicativo para revogar um token. Consulte Revogar endpoint.
https://cognito-idp.*Região*.amazonaws.com/*O ID do grupo de usuários*/.well-known/openid-configuration	Um diretório da arquitetura OIDC de seu grupo de usuários.	Solicitado pelo aplicativo para localizar metadados do emissor do grupo de usuários.
https://cognito-idp.*Region*.amazonaws.com/*O ID de seu grupo de usuários*/.well-known/openid-configuration	Chaves públicas que você pode usar para validar tokens do Amazon Cognito.	Solicitado pelo aplicativo para verificar os JWTs.
https://*O domínio de seu grupo de usuários*/oauth2/idpresponse	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.
https://*O domínio do grupo de usuários*/saml2/idpresponse	O URL do Assertion Consumer Response (ACS) para integração com provedores de identidade SAML 2.0.	Redirecionado do SAML 2.0 IdP como o URL do ACS ou o ponto de origem para o login iniciado pelo IdP. [1]
https://*Seu domínio do grupo de usuários* /saml2/logout	O URL de logout único (SLO) para integração com provedores de identidade SAML 2.0.	Redirecionado do SAML 2.0 IdP como URL de logout único (SLO). Aceita somente a vinculação POST.

[1] Para obter mais informações sobre o login SAML iniciado pelo IdP, consulte. Usando o login SAML iniciado pelo IdP

Para ter mais informações sobre os padrões OpenID Connect e OAuth, consulte OpenID Connect 1.0 e OAuth 2.0.

Tópicos

- Autorizar endpoint
- Endpoint de token
- Endpoint do UserInfo
- Revogar endpoint
- endpoint saml2/idpresponse

Autorizar endpoint

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 a interface de usuário hospedada 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 do OAuth 2.0 que você deseja solicitar no token de acesso do 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) A URL em que 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 comporta URLs de retorno de chamada da aplicação, como `myapp://example`.

`state`

(Opcional, recomendado) Quando seu aplicativo adiciona um parâmetro de estado a uma solicitação, o Amazon Cognito retorna seu valor ao seu aplicativo quando o `/oauth2/authorize` endpoint redireciona seu 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 passar uma string que corresponda a esse formato em um `state` parâmetro, codifique a string em base64 e decodifique-a no seu aplicativo.

`identity_provider`

(Opcional) Adicione esse parâmetro para ignorar a interface hospedada e redirecionar seu usuário para a página de login de um 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` `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 identity_provider. Você pode inserir identificadores para seu SAML 2.0 e OIDC na guia Experiência de login IdPs do console do Amazon Cognito.

`scope`

(Opcional) Pode ser uma combinação de qualquer escopo reservado pelo sistema ou escopo personalizado associado 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 usou 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 que você gerou a partir do`code_verifier`.

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).

Exemplos de solicitações com respostas positivas

Os exemplos a seguir ilustram o formato das solicitações HTTP para o `/oauth2/authorize` endpoint.

Concessão de código de autorização

Este é um exemplo de solicitação de concessão de código de autorização.

Exemplo — solicitação GET

A solicitação a seguir inicia uma sessão para recuperar um código de autorização que seu usuário passa para seu aplicativo no `redirect_uri` destino. 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
```

Exemplo — resposta

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
```

Concessão de código de autorização com PKCE

Este é um exemplo de solicitação de concessão de código de autorização com o PKCE.

Exemplo — solicitação GET

A solicitação a seguir adiciona um `code_challenge` parâmetro à solicitação anterior. Para concluir a troca de um código por um token, você deve incluir o `code_verifier` parâmetro em sua solicitação para o `/oauth2/token` endpoint.

```
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...
```

Exemplo — resposta

O servidor de autenticação redireciona de volta para seu aplicativo com o código e o estado da autorização. O código e o estado devem ser retornados nos parâmetros da string de consulta e não no fragmento:

```
HTTP/1.1 302 Found
Location: https://www.example.com?code=a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111&state=abcdefg
```

Concessão de token sem escopo **`openid`**

Esse é um exemplo de solicitação que gera uma concessão implícita e retorna JWTs diretamente para a sessão do usuário.

Exemplo — solicitação GET

A solicitação a seguir é para uma concessão implícita do seu servidor de autorização. O token de acesso do Amazon Cognito autoriza operações de API 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
```

Exemplo — resposta

O servidor de autorização do Amazon Cognito faz o redirecionamento de volta à aplicação com 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. O Amazon Cognito retorna o token de acesso e o estado no fragmento e não na string de consulta:

```
HTTP/1.1 302 Found
Location: https://YOUR_APP/
redirect_uri#access_token=ACCESS_TOKEN&token_type=bearer&expires_in=3600&state=STATE
```

Concessão de token com escopo **`openid`**

Esse é um exemplo de solicitação que gera uma concessão implícita e retorna JWTs diretamente para a sessão do usuário.

Exemplo — solicitação GET

A solicitação a seguir é para uma concessão implícita do seu servidor de autorização. O token de acesso do Amazon Cognito autoriza o acesso aos atributos do usuário e às operações de API 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
```

Exemplo — resposta

O servidor de autorização redireciona de volta para seu aplicativo com o token de acesso e o token de ID (porque o `openid` escopo 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 do cliente `redirect_uri` e anexará uma mensagem de erro em um parâmetro de URL. Veja a seguir exemplos de formatação incorreta.
  - A solicitação não inclui um `response_type` parâmetro.

- A solicitação de autorização forneceu um `code_challenge` parâmetro, mas não um `code_challenge_method` parâmetro.
- O valor do `code_challenge_method` parâmetro não é`S256`.

Veja a seguir a resposta a um exemplo de solicitação com formatação incorreta.

```
HTTP 1.1 302 Found Location: https://client_redirect_uri?error=invalid_request
```

- Se o cliente solicitar `code` ou entrar `tokenresponse_type`, mas não tiver permissão para essas solicitações, o servidor de autorização do Amazon Cognito retornará `unauthorized_client` ao do clienteredirect_uri, 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 houver algum erro inesperado no servidor, o servidor de autenticação retornará `server_error` ao servidor do clienteredirect_uri. Como o erro HTTP 500 não é enviado ao cliente, o erro não é exibido no navegador do usuário. O servidor de autorização retorna o seguinte erro.

```
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 ao chamar o `jwks_uri` endpoint para validação do token de ID, o servidor de autenticação redirecionará com um erro para o cliente da seguinte forma: `redirect_uri`

```
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 ser devido a 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 ao se conectar a um IdP externo, o servidor de autenticação redireciona com um erro para o cliente com uma das seguintes mensagens: `redirect_uri`
  - ```
    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
    ```

Endpoint de token

O endpoint do token do OAuth 2.0 em `/oauth2/token` emite tokens JSON da web (JWTs).

Seu servidor de autorização OAuth 2.0 do grupo de usuários 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. Sessões M achine-to-machine (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 na interface hospedada ou por meio da federação sempre podem atualizar seus tokens a partir do endpoint do 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 lembrados não estão ativos em seu grupo de usuários. Se os dispositivos lembrados estiverem ativos, atualize os tokens com as solicitações `AuthFlow` `REFRESH_TOKEN_AUTH` de entrada `InitiateAuth` ou de `AdminInitiateAuth` API.

O endpoint do token fica disponível ao público quando você adiciona um domínio ao grupo de usuários. Ele aceita solicitações HTTP POST. Para segurança do aplicativo, use o PKCE com seus 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 o PKCE, consulte IETF RFC 7636.

Você pode aprender mais sobre os clientes da aplicação do grupo de usuários e seus tipos de concessão, segredos de clientes, escopos autorizados e IDs de clientes em Clientes de aplicações de grupos de usuários. Você pode aprender mais sobre autorização M2M, concessões de credenciais de clientes e autorização com escopos de token de acesso em. Escopos, M2M e autorização de API com servidores de recursos

Para recuperar informações sobre um usuário a partir do token de acesso, passe-as para você Endpoint do UserInfo ou para uma solicitação de GetUserAPI.

POST /oauth2/token

O endpoint `/oauth2/token` só é compatível com `HTTPS POST`. Sua aplicação faz solicitações para esse endpoint diretamente, e não por meio do navegador do usuário.

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 OpenID Connect, consulte Autenticação do cliente. Para obter mais informações sobre o endpoint de token na especificação do OpenID Connect, consulte Endpoint de token.

Parâmetros de solicitação no cabeçalho

`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 do aplicativo `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

`grant_type`

(Obrigatório) O tipo de subsídio 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 ativou o escopo solicitado na configuração do seu cliente de aplicativo.
- Você configurou seu cliente de aplicativo com um segredo de cliente.
- Você ativa a concessão de credenciais de cliente em seu cliente de aplicativo.

`client_id`

(Opcional) O ID de um cliente de aplicativo em seu grupo de usuários. Especifique o mesmo cliente de aplicativo que autenticou seu usuário.

Você deve fornecer esse parâmetro se o cliente for público e não tiver um segredo ou se não tiver `client_secret_post` autorização. `client_secret`

`client_secret`

(Opcional) O segredo do cliente do aplicativo que autenticou seu usuário. Obrigatório se o cliente de aplicação tiver um segredo de cliente e você não tiver enviado uma cabeçalho de `Authorization`.

`scope`

(Opcional) Pode ser uma combinação de qualquer escopo personalizado associado a um cliente de aplicativo. Qualquer escopo que você solicitar deve ser ativado para o cliente do aplicativo. Caso contrário, o Amazon Cognito o ignorará. Se o cliente não solicitar nenhum escopo, o servidor de autenticação atribuirá todos os escopos personalizados que você autorizou na configuração do seu cliente de aplicativo.

Usado somente se o `grant_type` for `client_credentials`.

`redirect_uri`

(Opcional) Deve ser o mesmo `redirect_uri` que foi usado para `authorization_code` entrar`/oauth2/authorize`.

Você deve fornecer esse parâmetro se `grant_type` estiver`authorization_code`.

`refresh_token`

(Opcional) Para gerar novos tokens de acesso e ID para a sessão de um usuário, defina o valor de um `refresh_token` parâmetro em sua `/oauth2/token` solicitação como um token de atualização emitido anteriormente pelo mesmo cliente do aplicativo.

`code`

(Opcional) O código de autorização de uma concessão de código de autorização. Você deve fornecer esse parâmetro se sua solicitação de autorização incluir um `grant_type` de`authorization_code`.

`code_verifier`

(Opcional) O valor arbitrário que você usou para calcular o `code_challenge` em uma solicitação de concessão de código de autorização com o PKCE.

Exemplos de solicitações com respostas positivas

Como trocar um código de autorização por tokens

Exemplo — solicitação POST

```
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
```

Exemplo — resposta

```
HTTP/1.1 200 OK

                           Content-Type: application/json

                           {
                            "access_token":"eyJra1example",
                            "id_token":"eyJra2example",
                            "refresh_token":"eyJj3example",
                            "token_type":"Bearer",
                            "expires_in":3600
                           }
```

> **Note**
>
> O endpoint de token retorna `refresh_token` somente quando o `grant_type` é `authorization_code`.

Trocar credenciais de cliente para um token de acesso: segredo de cliente no cabeçalho de autorização

Exemplo — solicitação POST

```
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/scope1 resourceServerIdentifier2/scope2
```

Exemplo — resposta

```
HTTP/1.1 200 OK
                            Content-Type: application/json

                            {
                            "access_token":"eyJra1example",
                            "token_type":"Bearer",
                            "expires_in":3600
                            }
```

Trocar credenciais de cliente para um token de acesso: segredo de cliente no corpo da solicitação

Exemplo — solicitação POST

```
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%2F
```

Exemplo — resposta

```
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"
}
```

Como trocar uma concessão de código de autorização com PKCE por tokens

Exemplo — solicitação POST

```
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
```

Exemplo — resposta

```
HTTP/1.1 200 OK
                           Content-Type: application/json

                           {
                           "access_token":"eyJra1example",
                           "id_token":"eyJra2example",
                           "refresh_token":"eyJj3example",
                           "token_type":"Bearer",
                           "expires_in":3600
```

```
                        }
```

> **Note**
>
> O endpoint de token retorna `refresh_token` somente quando o `grant_type` é `authorization_code`.

Como trocar um token de atualização por tokens

Exemplo — solicitação POST

```
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
```

Exemplo — resposta

```
HTTP/1.1 200 OK
                        Content-Type: application/json

                        {
                        "access_token":"eyJra1example",
                        "id_token":"eyJra2example",
                        "token_type":"Bearer",
                        "expires_in":3600
                              }
```

> **Note**
>
> O endpoint de token retorna `refresh_token` somente quando o `grant_type` é `authorization_code`.

Exemplos de respostas negativas

Exemplo — resposta de 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`.

Endpoint do UserInfo

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 de token 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.

O provedor de identidades (IdP) externo do OAuth 2.0 também hospeda um endpoint userInfo. Quando seu usuário se autentica com esse IdP, o Amazon Cognito troca silenciosamente um código de autorização com o endpoint do IdP. `token` Seu grupo de usuários passa o token de acesso do IdP para autorizar a recuperação das informações do usuário do endpoint do IdP. `userInfo`

GET /oauth2/userInfo

A aplicação faz solicitações para esse endpoint diretamente e 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 de respostas negativas

Parâmetros de solicitação no cabeçalho

`Authorization: Bearer <access_token>`

Passe o token de acesso no campo do cabeçalho de 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 de 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 tem um parâmetro obrigatório, inclui um valor de parâmetro não suportado ou está mal formada.

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 expirou, foi revogado, está mal formado ou é inválido.

Revogar endpoint

O `oauth2/revoke` endpoint/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 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 revogados para acessar APIs autenticadas pelos tokens do Amazon 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 do seu aplicativo tiver um segredo de cliente, o aplicativo deverá passar seu `client_id` e `client_secret` no cabeçalho de 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 deseja 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 do aplicativo para o token que você deseja revogar.

Obrigatório se o cliente for público e não tiver um segredo.

Exemplos de solicitação de revogação

Exemplo 1: Revogar um token para um cliente da aplicação sem um segredo do cliente

```
POST /oauth2/revoke HTTP/1.1
    Host: https://mydomain.auth.us-east-1.amazoncognito.com
    Accept: application/json
    Content-Type: application/x-www-form-urlencoded
    token=2YotnFZFEjr1zCsicMWpAA&
    client_id=djc98u3jiedmi283eu928
```

Exemplo 2: Revogar um token para um cliente da aplicação com um segredo do cliente

```
POST /oauth2/revoke HTTP/1.1
    Host: https://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.

endpoint saml2/idpresponse

O `/saml2/idpresponse` recebe afirmações de SAML. No login service-provider-initiated (iniciado pelo SP), seu provedor de identidade (IdP) do SAML 2.0 redireciona o usuário para esse endpoint com a resposta do SAML. No login iniciado pelo SP, seu aplicativo não interage com esse endpoint. Configure seu IdP com o caminho para sua URL `saml2/idpresponse` como assertion consumer service (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, seus usuários podem fazer login com seu IdP por meio de seu próprio processo e enviar uma declaração de SAML no corpo de uma solicitação por HTTPS. `HTTP POST` O corpo da sua `POST` solicitação deve ser um `SAMLResponse` parâmetro e um `Relaystate` parâmetro. Para ter mais informações, consulte [Usando o login SAML iniciado pelo IdP](#).

POSTAR **`/saml2/idpresponse`**

Para usar o `/saml2/idpresponse` endpoint em um login iniciado pelo IdP, gere uma solicitação POST com parâmetros que forneçam ao grupo de usuários informações sobre a sessão do usuário.

- O cliente do aplicativo no qual eles querem fazer login.
- O URL de retorno de chamada em que eles querem terminar.
- Os escopos do 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

Resposta SAML

Uma declaração SAML codificada em Base64 de um IdP associado a um cliente de aplicativo válido e a uma configuração de IdP em seu grupo de usuários.

RelayState

Um `RelayState` parâmetro contém os parâmetros de solicitação que, de outra forma, você passaria para o `oauth2/authorize` endpoint. Para obter informações detalhadas sobre esses parâmetros, consulteAutorizar endpoint.

response_type

O tipo de concessão do 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 identidade para o qual você deseja redirecionar seu usuário.

idp_identifier

O identificador do provedor de identidade para o qual você deseja redirecionar seu usuário.

scope

Os escopos do 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 do aplicativo. `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 do OAuth 2.0, e. `openid 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

A seguir está a resposta à 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]
```

Concessões do OAuth 2.0

O servidor de autorização do OAuth 2.0 do grupo de usuários do Amazon Cognito emite tokens em resposta a três tipos de concessão de autorização do 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 Concessão de código 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 Concessão de token 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 escopos do OAuth 2.0. Normalmente, o token contém declarações de escopo personalizado que autorizam operações HTTP a acessar APIs protegidas. Para ter mais informações, consulte Escopos, M2M e autorização de API com servidores de recursos.

As concessões de credenciais do cliente adicionam custos à sua AWS fatura. Para mais informações, consulte Preços do Amazon Cognito.

Usando o PKCE em concessões de código de autorização com grupos de usuários do Amazon Cognito

O Amazon Cognito oferece suporte à autenticação Proof Key for Code Exchange (PKCE) em concessões de códigos de autorização. O PKCE é uma extensão da concessão do código de autorização OAuth 2.0 para clientes públicos. O PKCE se protege contra o resgate de códigos de autorização interceptados.

Como o Amazon Cognito usa o PKCE

Para iniciar a autenticação com o PKCE, seu aplicativo 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 hash SHA256 à string do verificador de código e codificar o resultado em base64. Passe a string com hash para o Autorizar endpoint como um `code_challenge` parâmetro no corpo da solicitação. Quando seu aplicativo troca o código de autorização por tokens, ele deve incluir a string do verificador de código em texto simples como um `code_verifier` parâmetro 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 na mesma contestação 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, você será solicitado a configurar um cliente de aplicativo e configurar a interface de usuário hospedada durante o assistente.

   a. Se você criar um novo grupo de usuários, configure um cliente de aplicativo e configure a interface hospedada durante a configuração guiada.

   b. Se você configurar um grupo de usuários existente, adicione um domínio e um cliente de aplicativo 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 `code_verifier` parâmetro que você enviará em sua solicitação para Endpoint de token o.
5. Faça o hash da `code_verifier` string com o algoritmo SHA256. Codifique o resultado da operação de hashing para base64. Essa string é o valor do `code_challenge` parâmetro que você enviará em sua solicitação para Autorizar endpoint o.

   O Python exemplo a seguir gera um `code_verifier` e calcula o`code_challenge`:

```
#!/usr/bin/env python3

import random
from base64 import urlsafe_b64encode
from hashlib import sha256
from string import ascii_letters
from string import digits

# use a cryptographically strong random number generator source
rand = random.SystemRandom()

code_verifier = ''.join(rand.choices(ascii_letters + digits, k=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 a seguir um exemplo de saída do Python script:

```
code challenge: Eh0mg-OZv7BAyo-tdv_vYamx1boOYDulDklyXoMDtLg
```

```
code verifier: 9D-aW_iygXrgQcWJd0y0tNVMPSXSChIc2xceDhvYVdGLCBk-
JWFTmBNjvKSdOrjTTYazOFbUmrFERrjWx6oKtK2b6z_x4_gHBDlr4K1mRFGyE8yA-05-_v7Dxf3EIYJH
```

6. Conclua o login na interface de usuário hospedada com uma solicitação de concessão de código de autorização junto ao PKCE. Veja a seguir um exemplo de URL:

```
https://mydomain.us-east-1.amazoncognito.com/oauth2/authorize?
response_type=code&client_id=1example23456789&redirect_uri=https://
www.example.com&code_challenge=Eh0mg-0Zv7BAyo-
tdv_vYamx1boOYDulDklyXoMDtLg&code_challenge_method=S256
```

7. Colete a autorização `code` e troque-a por tokens com o endpoint do token. Veja a seguir uma solicitação de exemplo:

```
POST /oauth2/token HTTP/1.1
Host: mydomain.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. Analise a resposta. Ele conterá tokens de ID, acesso e atualização. Para obter mais informações sobre o uso de tokens do grupo de usuários do Amazon Cognito, consulte. Como usar tokens com grupos de usuários

Respostas de erro de federação e da interface do usuário hospedada

Um processo de login na interface do usuário hospedada 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 na interface do usuário hospedada.

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 IdP OIDC ou OAuth 2.0 retorna um erro que está de acordo com os padrões, o Amazon Cognito redireciona o usuário para o URL de retorno de chamada e adiciona a resposta de erro do provedor aos parâmetros da solicitação com 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 mensagens de erro do IdP OAuth e OIDC dos endpoints a seguir.

`/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 tokens web JSON (JWTs) do usuário, o Amazon Cognito descobre as chaves web JSON (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.

Referência de API de grupos de usuários do Amazon Cognito

Com os grupos de usuários do Amazon Cognito, é possível inscrever e fazer login de usuários com a aplicação Web e móvel. Você pode alterar senhas para usuários autenticados e iniciar fluxos de senha esquecida para usuários não autenticados. Para obter mais informações, consulte Como usar tokens com grupos de usuários e Fluxo de autenticação de grupo de usuários.

A API de grupos de usuários do Amazon Cognito inclui operações para visualizar e modificar usuários e grupos de usuários para realizar autenticação e autorização de usuários. Para obter uma descrição das classes das operações de API que se combinam com a API dos grupos de usuários do Amazon Cognito, consulte Usar a API de grupos de usuários e endpoints de grupo de usuários do Amazon Cognito.

Para obter uma lista detalhada das operações e sintaxe da API de grupos de usuários do Amazon Cognito, consulte a Referência de API de grupos de usuários do Amazon Cognito. Cada página na referência de API dos grupos de usuários do Amazon Cognito está vinculada ao material de referência com sintaxe e exemplos para diversos AWS SDKs.

Referência de API de grupos de identidades (identidades federadas) do Amazon Cognito

Com um grupo de identidades do Amazon Cognito, seus usuários de aplicações Web e móveis podem obter credenciais da AWS com privilégio temporário e limitado, habilitando-os a acessar outros serviços da AWS.

Para obter uma referência de API completa dos grupos de identidades (identidades federadas), consulte Referência da API do Amazon Cognito.

Referência de API do Amazon Cognito Sync

O Amazon Cognito Sync é um serviço da AWS e uma biblioteca de clientes que permite a sincronização entre dispositivos de dados de usuário relacionados a aplicações.

Para obter mais informações sobre a Referência da API do Amazon Cognito Sync, consulte Referência da API do Amazon Cognito Sync.

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
Foi adicionado suporte para objetos complexos no gatilho Lambda pré-token	Agora você pode adicionar matrizes e objetos JSON às declarações de ID e token de acesso.	30 de maio de 2024
Informações atualizadas sobre permissões verificadas 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 alguns Regiões da AWS sem o Amazon SES, os grupos de usuários do Amazon Cognito equilibram a carga de e-mails entre duas regiões remotas.	10 de maio de 2024
Foram adicionadas informações sobre autorização M2M e 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 da Europa	15 de abril de 2024

	(Espanha) e Ásia-Pacífico (Hyderabad).	
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 Android no Flutter para grupos de usuários do Amazon Cognito.	Você pode criar um aplicativ o móvel inicial para o Amazon Cognito a partir de um exemplo de aplicativo Flutter ativado. GitHub	4 de abril de 2024
Novo conteúdo para começar	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 você pode criar recursos do Amazon Cognito na região da Europa (Zurique).	14 de março de 2024
O Amazon Cognito agora está disponível no Oriente Médio (EAU). Região da AWS	Agora você pode criar recursos do Amazon Cognito na região do 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 SAML SSO iniciado pelo IdP.	1 de fevereiro de 2024

Aumentos de cota disponíveis.	Agora você pode comprar capacidade adicional para as cotas de taxa de solicitação do Amazon Cognito.	25 de janeiro de 2024
Os grupos de identidade do Amazon Cognito suportam taxas de solicitação em Cotas de Serviço.	Agora você pode monitorar cotas requests-per-second (RPS) para grupos de identidade do Amazon Cognito e solicitar aumento no console Service Quotas.	19 de dezembro de 2023
Foi adicionado um novo recurso para personalização do conteúdo dos tokens de acesso.	Agora é possível adicionar, modificar e remover declarações e escopos nos tokens de acesso do grupo de usuários.	12 de dezembro de 2023
Conteúdo aprimorado sobre clientes de aplicativos e escopos do OAuth.	Edições e correções de Clientes de aplicações de grupos de usuários e Escopos, M2M e autorização de API com servidores de recursos para aumentar a clareza. Instruções do console herdado removidas.	14 de novembro de 2023
Conteúdo aprimorado 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

AWS Management Console 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
Acesso direto sem ênfase 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 Ásia-Pacífico (Osaka) e Israel (Tel Aviv).	30 de agosto de 2023
Introduziu informações sobre autorização para o Amazon Cognito com permissões verificadas pela Amazon.	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 do conteúdo e novas instruções do console para grupos de identidade do Amazon Cognito.	Adição de novas orientaçõ es do console para refletir a nova experiência do console, detalhes aprimorad os 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 informaçõ es sobre verificação de tokens foram adicionadas.	16 de fevereiro de 2023
Agora você pode registrar eventos de dados de grupos de identidade do Amazon Cognito em. 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 gatilhos do Lambda atualizad os.	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
Exemplos de código adicionad os.	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 informaçõ es sobre a proteção contra exclusão do grupo de usuários.	Quando você cria um novo grupo de usuários com o AWS Management Console, ele agora está protegido contra exclusão por padrão.	20 de outubro de 2022
Foi adicionado um guia do usuário para a interface hospedada e informações sobre TOTP MFA 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 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 atualizada da federação. Novo recurso de propagação de endereços 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
Faça login com usuários federados sem interação com a interface hospedada	Foi adicionada uma nova página sobre como marcar aplicativos como favoritos para que o Amazon Cognito direcione silenciosamente os usuários para o login federado.	29 de maio de 2022

Mensagens de SMS e e-mail 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 as cotas de recursos e taxas de solicitação.	10 de janeiro de 2022
Nova experiência de 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 para vários locatários	Práticas recomendadas adicionadas para aplicativos multilocatários.	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
Remetente de SMS personalizado Lambda Trigger e remetente de e-mail personalizado Lambda Trigger	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

Quotas do Amazon Cognito Service	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 a GovCloud dos EUA AWS	O Amazon Cognito agora tem suporte 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
Os grupos de identidade do Amazon Cognito agora oferecem suporte ao 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 controle de versão da API do Facebook	Seleção de versão adicionada à API do Facebook	3 de abril de 2020
Atualização de insensibilidade entre maiúsculas e minús	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 usando SDKs e bibliotecas. Foram removidas informações sobre o uso de SDKs do Amazon Cognito anteriores ao AWS Amplify.	22 de novembro de 2019
Novo atributo para acionador es de grupos 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 para marcação	Adicionadas informações sobre marcação de recursos do Amazon Cognito.	26 de março de 2019
Alterar o certificado de 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 verificação de 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 a interface de 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

Grupos de usuários do Amazon Cognito | Provedor de identidade OIDC	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 de migração do Amazon Cognito Lambda	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
Beta de segurança avançada 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 com o 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 interface de usuário de aplicativos incorporados 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. Adição de uma interface do usuário de aplicativo interno e suporte a OAuth 2.0 com solicitações personalizadas.	10 de agosto de 2017
Alterações de recursos relacionadas à conformidade com HIPAA e PCI	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 baseados em funções	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 códigos 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
Recurso de criação 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

Suporte 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
Stream 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 push	Habilita o suporte para a sincronização por push silenciosa.	6 de novembro de 2014
Suporte de identidades autenticadas pelo desenvolvedor adicionado	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.