Ambientes de Desenvolvimento

Aula 11 – Serviços de Autenticação

Rodolfo Riyoei Goya
rodolfo.goya@faculdadeimpacta.com.br

Referências

- https://aws.amazon.com/pt/cognito/
- https://aws.amazon.com/pt/cognito/pricing/
- https://en.wikipedia.org/wiki/Security_Assertion_Markup_Language
- https://aws.amazon.com/pt/identity/saml/
- https://datatracker.ietf.org/doc/html/rfc7519
- https://developers.facebook.com/

Amazon Cognito

aws academy

Amazon Cognito

Amazon Cognito

O **Amazon Cognito:**

- Permite adicionar cadastramento, login e controle de acesso de usuários a aplicações Web e móveis com rapidez e facilidade.
- Pode ser escalado para milhões de usuários e oferece suporte a login com:
  - Provedores de identidade social como Apple, Facebook, Google e Amazon
  - Provedores de identidade empresariais via SAML 2.0 e OpenID Connect

Amazon Cognito

O **Amazon Cognito:**

Amazon Cognito

- Permite manter o foco na lógica de negócios e delegar os serviços de autenticação
- No nível gratuito, permite 50.000 usuários novos por mês

Integração API Gateway com Cognito

Aplicação Web sem servidor

Serviços do AWS Cognito

- Group pools: Armazenamento de identidades seguro e escalável
- Federação de identidades sociais e empresariais
- Baseado em padrões
- Seguro para aplicativos e usuários
- Controle de acesso para recursos da AWS
- Fácil integração com os aplicativos

Serviços do AWS Cognito

- Group pools: Armazenamento de identidades seguro e escalável
  - Os grupos de usuários (group pools) fornecem um armazenamento de identidades seguro que escala até milhões de usuários
  - Podem ser configurados mais facilmente sem provisionar infraestruturas, e todos os membros do grupo de usuários têm um perfil de diretório que pode ser gerenciado por meio de um Kit de Desenvolvimento de Software (SDK)

Serviços do AWS Cognito

- Federação de identidades sociais e empresariais
  - login usando provedores de identidade social, como Google, Facebook e Amazon
  - provedores de identidade empresarial como SAML e OpenID Connect

Padrões de Mercado

- Baseado em padrões
  - Os grupos de usuários (group pools) do Amazon Cognito são provedores de identidade baseados em padrões de mercado
  - Oferecem suporte a padrões de gerenciamento de identidade e acesso, como OAuth 2.0, SAML 2.0 e OpenID Connect

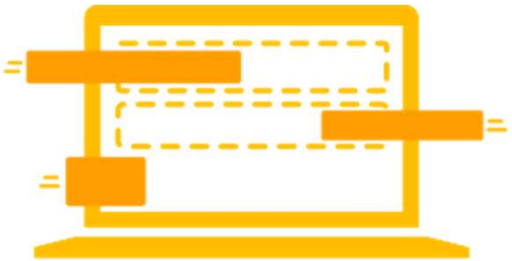
SAML

- Security Assertion Markup Language
  - Baseado em XML
  - Padrão aberto
  - Autenticação e autorização
  - Três participantes:
    - Usuário
    - Identity provider
    - Service provider
- Redução de infraestrutura e mão de obra para suporte e operação
  - Menor base de dados
  - Dispensa procedimentos de recuperação de senhas perdidas

Segurança

- Seguro para aplicativos e usuários
  - O Amazon Cognito oferece suporte à autenticação multifator e à criptografia de dados em repouso e em trânsito
  - O Amazon Cognito está qualificado pela HIPAA e é compatível com as certificações PCI DSS, SOC, ISO/IEC 27001, ISO/IEC 27017, ISO/IEC 27018 e ISO 9001
- Permite, opcionalmente, usar MFA

Fácil integração com os aplicativos

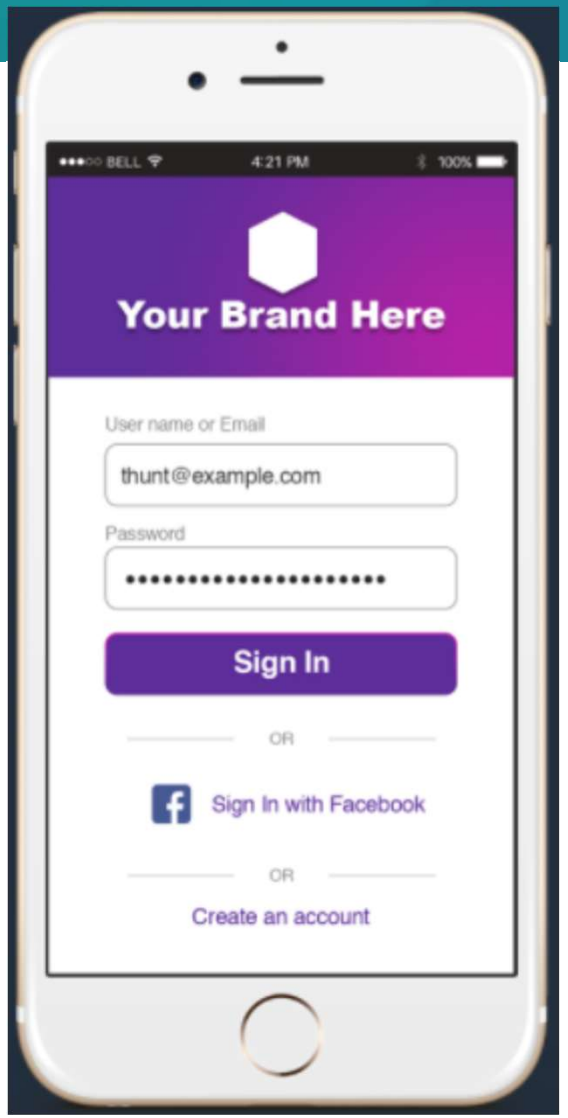


- Interface do usuário incorporada
- Facilidade de configuração para federar provedores de identidade
- Cadastramento, login e controle de acesso aos aplicativos configurado em questão de minutos
- É possível personalizar a interface do usuário destacando a identidade visual da empresa para todas as interações dos usuários

Controle de acesso para recursos da AWS

- O Amazon Cognito oferece soluções para controlar o acesso de um aplicativo a recursos da AWS
- Pode-se definir funções e mapear usuários a funções diferentes para que o aplicativo possa acessar apenas os recursos autorizados para cada usuário
- Opcionalmente, é possível usar atributos de provedores de identidade em políticas de permissão do AWS Identity and Access Management para controle de acesso a recursos por usuários que atendem a condições de atributos específicas

Interface Customizável



Código cliente: Android (java)

```
// 1) -- Create an instance of Auth –

Auth.Builder builder = new Auth.Builder()
.setAppClientId(getString(R.string.cognito_client_id));
.setAppCognitoWebDomain(getString(R.string.cognito_web_domain));
.setApplicationContext(getApplicationContext());

.setAuthHandler(new callback());
.setSignInRedirect(getString(R.string.app_redirect_signin));
.setSignOutRedirect(getString(R.string.app_redirect_signout));
.setScopes(userScopes);

auth = builder.build();

// 2) – Set up url redirect in your app manifest –

<intent-filter> <data
android:host="YOUR_REDIRECT_URI_AUTHORITY"android:scheme="YOUR_REDIRECT_SCHEME"/>

</intent-filter>

// 3) -- Get tokens for your user –

auth.getSession();
```

Código cliente: Aplicativo WEB (js)

```
// Add 'aws-amplify' library into your application
// Configure Auth category with your Amazon Cognito credentials
Amplify.configure({
  Auth: {
    identityPoolId: 'XX-XXXX-X:XXXXXXXX-XXXX', // Cognito Identity Pool ID
    region: 'XX-XXXX-X', // Amazon Cognito Region
} });

// Call Auth.signIn with user credentials
Auth.signIn(username, password)
  .then(user => console.log(user))
  .catch(err => console.log(err));
```

Código cliente: iOS (C)

```
// AppDelegate.swift
func application(_ app: UIApplication, open url: URL,
     options: [UIApplicationOpenURLOptionsKey : Any] = [:]) -> Bool {
  return AWSCognitoAuth.default().application(app, open: url, options: options)
}
// ViewController.swift
let cognitoAuth = AWSCognitoAuth.default()
cognitoAuth.getSession(self) { (session:AWSCognitoAuthUserSession?, error:Error?) in
if(error != nil) {
  print((error! as NSError).userInfo["error"] as? String)
} else {
  //Do something with session }
}
```

Código cliente: React (js)

```
// Add 'aws-amplify' and 'aws-amplify-react-native' libraries into your application
// Configure Auth category with your Amazon Cognito credentials
Amplify.configure({
  Auth: {
    identityPoolId: 'XX-XXXX-X:XXXXXXXX-XXXX', // Amazon Cognito Identity Pool ID
    region: 'XX-XXXX-X', // Amazon Cognito Region
  }
});

// The withAuthenticator component adds Sign Up and Sign In capabilities to your app
export default withAuthenticator(class App extends React.Component {
  // ... your main componente
});
```

Pricing

Níveis de definição de preço (MAUs)	Preço por MAU
Primeiros 50.000	Gratuito
Próximos 50.000	0,00550 USD
Próximos 900.000	0,00460 USD
Próximos 9.000.000	0,00325 USD
Mais de 10.000.000	0,00250 USD

MAU – Monthly Active User:
Contado se, durante o mês, fizer cadastro, login, Alteração de senha ou atributo pelo menos uma vez.

https://aws.amazon.com/pt/cognito/pricing/

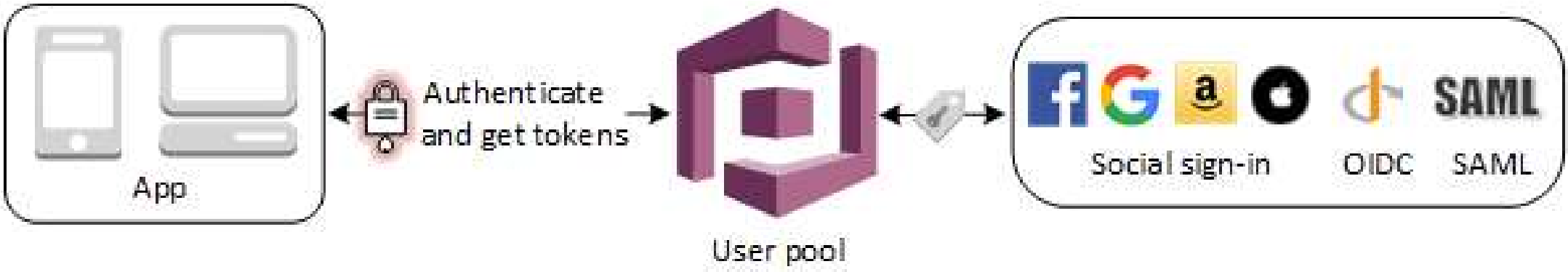
Criação de User pools

- Pré-requisito: Cadastrar-se em uma conta da AWS
- Etapa 1. Criar um grupo de usuários
- Etapa 2. Adicionar um aplicativo para habilitar a interface do usuário da web hospedada

- Opcionais:
  - Etapa 3. Adicionar um login social a um grupo de usuários
  - Etapa 4. Adicionar login com um provedor de identidade SAML a um grupo de usuários

Grupo de usuários – User pool

- Através do serviço de Autenticação no Cognito, dentro do User Pool, o cliente obtém um Token (JSON Web Token - JWT)
  - Criptografado
  - Prazo de validade
- Serviços AWS (S3, Lambda, API Gateway, DynamoDB, etc) podem ser configurados para autorizar acesso apenas mediante a apresentação do token na chamada



JSON Web Token – JWT

- Um JWT é um documento JSON contendo informações sobre um usuário autenticado
- Criado em 2015 na RFC7519 https://datatracker.ietf.org/doc/html/rfc7519
- Formato flexível, contém informações como nome, email, telefone que podem ser usados pela aplicação
- Conteúdo:
  - Cabeçalho: ID da chave e Algoritmo de criptografia
  - Payload: flexível, com informações (nome, email, telefone,...) uteis para aplicação
  - Emissor: URL do cognito
  - Tempo de vida útil (entre 60 min. e 10 anos)
  - Assinatura criptografada

Outras formas de Token

- Token de Atualização
- Token de Grupo
- Revogação de validade de Token

Debugging

- Como verificar a validade de um Token:
  - Estrutura:
    - Cabeçalho
    - Payload
    - Assinatura em Base64

Debugging

- Como verificar a validade de um Token:
  - Validação da assinatura
    - É necessário executar um código para verificar se uma assinatura é válida
    - Há templates em Lambda para validação:

```
var jwt = require('jsonwebtoken’);
var jwkToPem = require('jwk-to-pem’);
var pem = jwkToPem(jwk);
jwt.verify(token, pem, { algorithms: ['RS256'] }, function(err, decodedToken) { } );
```

Debugging

- Como verificar a validade de um Token:
  - Verificação do Payload:
    - Prazo de validade
    - Grupo de Usuários
    - Região

Dúvidas?