※ 本セッションの内容は 「OSSセキュリティ技術の会第10回勉強会」 とほぼ同じですのでご了承ください

## KeycloakでAPI認可に入門する

2022/03/12

#### 日立製作所 OSSソリューションセンタ 中村 雄一

© Hitachi, Ltd. 2022. All rights reserved.

元ネタの書籍

- 認証と認可Keycloak入門
   今回の内容の詳細、SSO、様々な認証
   (多要素認証、外部ストレージ等)
   HA構成やカスタマイズの基本までカバー
- 飛ばされがちな前提知識(OAuth/OIDC/SAML、SSOの構成)から解説
- ・「入門書」です。

https://ric.co.jp/book/new-publication/detail/2081 https://www.amazon.co.jp/dp/4865943226/





### Contents

- 1. <u>Keycloakとは</u>
- 2. API認可の基礎
- 3. Keycloakの基本概念
- 4. API認可をKeycloakで試す



認証と認可



### Keycloakとは



- OSSの「IAM(Identity and Access Management)」のソフトウェア
   <a href="https://www.keycloak.org/">https://www.keycloak.org/</a>
- コミュニティが活発・オープンで、商用利用も広く進んでいる
  - 日本人のメンテナも!(乗松さん@日立) - 商用サポート版がRed Hat社から(Red Hat Single Sign-On)
- 最新標準にも対応。v15.0では、OpenID FoundationのFAPI, FAPI-CIBA, Open Banking Brasil という仕様にも正式対応。 <a href="https://openid.net/certification/">https://openid.net/certification/</a>
- 豊富な機能

分類	機能	概要
認証	認証情報管理	認証に必要なユーザ名・属性情報を管理
	パスワード認証	ユーザ名・パスワードに基づき認証
	多要素認証	OTP(FreeOTPベース)や、WebAuthnに基づいた多要素認証を提供
	外部連携	外部の認証情報と連携して認証する
認可	権限管理	ユーザやアプリに紐づく権限を管理
	認可サービス	権限とリソースの紐づけを管理し、認可判断を実行する
認証・認可連携	OAuth 2.0	APIの認可サーバとして必要なエンドポイントを提供(認可エンドポイ
プロトコル	認可サーバ	ント、トークンエンドポイント、トークン失効エンドポイント等)
	OpenID Connect OP	OpenID ConnectのOPサーバとしての機能を提供
	SAML IdP	SAMLのIdPサーバとしての機能を提供
カスタマイズ	SPI	様々な拡張を作りこむためのJavaのインタフェース
ポイント	クライアントポリシー	クライアント毎に異なった振る舞いをKeycloakに持たせるための拡張 フレームワーク
アプリ用 ライブラリ	クライアントアダプタ	アプリにOpenID Connect RP,SAML SPの機能を持たせたり、認可 サービスのクライアント機能を持たせるアプリ用ライブラリ

5

Keycloakでできること



#### Keycloakの構成







- ・ 前提環境: Java(JDK 8以上)。最低でも512MのRAMと1Gのディスク容量
- <u>公式サイト</u>よりzipまたはtar.gzをダウンロード。
   書籍で使っているバージョンは15.0.2。本勉強会時点の最新版は16.1.1
- インストール&起動(Windowsの場合)
   zipを展開し、コマンドプロンプトで展開したディレクトリに移動し、

.¥bin¥standalone.bat

とするだけ。



#### ウェルカムページ(http://localhost:8080/)をブラウザで開き、管理者ユーザーを作成

Welcome to <b>K</b>	eycloak	
Administration Console Please create an initial a	on Documentation >	Keycloak Project >
to get started. Username	Javadocs	Mailing List >
Password Password confirmation		
Create		∰ Report an issue >
ユーザー名とパスワードを 管理者ユーザーを作り	入力し成	Boss Community

管理コンソールにログイン



- Keycloakの設定は管理コンソールから行う。
  ウェルカムページから「Administration Console」を選択し、管理者ユーザーでログイン

🚳 Keycloak Admin Console	× +					• - • ×
$\leftrightarrow$ $\rightarrow$ C (i) localhost:80	080/auth/admin/master/consol	le/#/realms/master				🖈 😕 E
						💄 Admin 🗸
Master ~	Master 👕					
Configure	General Login	Keys Email Themes	Localization Cache	Tokens Client	Registration Client Pol	icies Security Defenses
🚻 Realm Settings	* Name	master				
😭 Clients 🚓 Client Scopes	Display name	Keycloak				
Roles	HTML Display name	HTML Display name <pre><div class="kc-logo-text"><span>Keycloak</span></div></pre>				
	Frontend URL 🕼					
User Federation	Enabled 😡	ON				
	User-Managed Access 🖗	OFF				
Manage	Endpoints 🔞	OpenID Endpoint Configurat	ion			
Groups		SAML 2.0 Identity Provider M	letadata			
<ul> <li>Osers</li> <li>Sessions</li> </ul>		Save Cancel				
🛗 Events						
🖾 Import						
Export						

「レルム」とはKeycloak の管理単位のことで、ユーザーなどの情報はレルム単位で管理。 「master」レルムは、すべてのレルムを管理できる特別なレルム(先ほどは「master」レルム) 任意にレルムを作成して管理ができ(部署単位、用途単位等)、マルチテナントのような使い方もできる。

	master レルム 管理者ユーザー	master レルム 管理者ユーザー				
	department A レル	department A レルム				
	レルムの設定	クライアント	ロール			
(eycloakサーバー	セッション	ユーザー	グループ			
	department B レル	Ъ				
	レルムの設定	クライアント	ロール			
	セッション	ユーザー	グループ			





#### API認可の動作確認用に、「demo-api」レルムを作成

		<b>WIKEYCLOAK</b>
Master 🗸	Master 👕	Select realm Y Add realm
Add realm	General Login Ke	e Import Select file 🖸
🚻 Realm Settings	* Name	Name * demo-api
- Clients		Enabled ON
🚕 Client Scopes	Display name	Create Cancel
Roles	HTML Display name	



		Success! The realm has been created. X	🛔 Admin 🖌
Demo-api 🗸 🗸	Demo-api 👕		
Configure	General Login	Keys Email Themes Localization Cache Tokens Client Registration Client Policies	
🚻 Realm Settings	Security Defenses		
😭 Clients	* Name	demo-api	
🙈 Client Scopes	Display pame		
Roles	Display name		
⇒ Identity Providers	HTML Display name		
User Federation	Frontend URL 😡		
Authentication	Enabled 🚱	ON	
Manage	User-Managed Access	OFF	
ka Groups	Ø		
💄 Users	Endpoints 🕖	OpenID Endpoint Configuration	
② Sessions		SAML 2.0 Identity Provider Metadata	
🛗 Events		Save Cancel	
🖾 Import			
Export			



### Contents

- 1. Keycloakとは
- 2. <u>API認可の基礎</u>
- 3. Keycloakの基本概念
- 4. API認可をKeycloakで試す



• API 認可: API 呼び出しを行う際の認可

• API認可では、「OAuth 2.0」が広く使われている。

 OAuth 2.0 (以下OAuth)
 ユーザーのリソース(protected resource)へのアクセスを
 外部アプリケーション(3<sup>rd</sup> party application)に認可するためのプロトコル RFC6749で規定

#### OAuthの登場人物とAPI認可



#### OAuthの典型的なフロー : 認可コードフロー



Keycloakにおけるアクセストークン





#### リフレッシュトークン

- アクセストークンには有効期限があり、短くすることが推奨(数分、数十分)
   期限が切れるたびに認可コードフローで認証や認可をしていたら不便。
- リフレッシュトークンはアクセストークンよりも長い期限(数時間や数日など)。
   アクセストークンと一緒に発行。
   クライアントはリフレッシュトークンを使えばアクセストークンを再取得できる。



トークンの無効化

- 有効期限内であっても、アクセストークンやリフレッシュトークンは 無効にすることができる。
   (セキュリティ上の理由や、リソースオーナーがAPI連携解除したい場合)
- Keycloakがサポートする無効化の手段

無効化の方法	誰が無効化するか
トークン無効化エンドポイントによる無効化 (RFC7009)	クライアント
アカウント管理コンソールによる無効化	リソースオーナー
管理コンソールによる無効化	管理者ユーザー

リソースサーバーでのアクセストークンの扱い



HITACHI

Inspire the Next

OAuth2.0を拡張した「認証」のためのプロトコル。用語が異なったり、追加概念(IDトークン等)がある。 中身はOAuth2.0であるため、API認可にも使うことが可能。





### Contents

- 1. Keycloakとは
- 2. API認可の基礎
- 3. <u>Keycloakの基本概念</u>
- 4. API認可をKeycloakで試す



### Keycloakの設定を行うためには、特有の用語の理解が必要。

### 「ユーザー」「クライアント」「セッション」「ロール」「グループ」 「レルム」「クライアントアダプター」「プロトコルマッパー」 「クライアントスコープ」など

今回は、最も基本的な「クライアント」と「セッション」を紹介。

クライアント



- 認証・認可サーバーであるKeycloakのサービスを利用するアプリケーションのこと。 OAuthのクライアント、OIDCのRP、SAMLのSP。
- ・ 管理コンソールの「Clients」画面より作成、管理できる。

					💄 Admin 🗸
Demo 🗸	Clients				
Configure	Lookup 🔞				
👫 Realm Settings	Search	Q			Create
🍞 Clients	Client ID	Enabled	Base URL	Actions	
🚷 Client Scopes	account	True	http://localhost:8080/auth/realms/demo/account/	Edit Export	Delete
Roles	account-console	True	http://localhost:8080/auth/realms/demo/account/	Edit Export	Delete
→ Identity Providers	admin-cli	True	Not defined	Edit Export	Delete
	broker	True	Not defined	Edit Export	Delete
User Federation	realm-management	True	Not defined	Edit Export	Delete
Authentication	security-admin-console	True	http://localhost:8080/auth/admin/demo/console/	Edit Export	Delete
Manage Line Groups Users Sessions					

セッション



- Keycloakではログインが成功するとセッションが生成され、メモリー 上で管理。
- アクセストークンやリフレッシュトークンもセッションと関連付けられて管理されている。セッションが無効になるとトークンも無効に。





### Contents

- 1. Keycloakとは
- 2. API認可の基礎
- 3. Keycloakの基本概念
- 4. <u>API認可をKeycloakで試す</u>



## 4. API認可をKeycloakで試す

- ① 検証環境を構築する
- 2 検証に必要な設定をする
- ③ 検証環境を動かしてみよう
- ④ トークンリフレッシュとトークン無効化を試してみる
- ⑤ セキュリティを向上する



## 4. API認可をKeycloakで試す

### ① <u>検証環境を構築する</u>

- 2 検証に必要な設定をする
- ③ 検証環境を動かしてみよう
- ④ トークンリフレッシュとトークン無効化を試してみる
- ⑤ セキュリティを向上する

#### ①検証環境を構築する

・ 今回実現するもの → 認可コードフローでアクセストークンを取得しAPIを呼び出すための検証環境



#### ①検証環境を構築する

・ 今回実現するもの → 認可コードフローでアクセストークンを取得しAPIを呼び出すための検証環境
 https://github.com/keycloak-book-jp から検証環境のソースコードを取得可能





## 4. API認可をKeycloakで試す

- ① 検証環境を構築する
- ② 検証に必要な設定をする
- ③ 検証環境を動かしてみよう
- ④ トークンリフレッシュとトークン無効化を試してみる
- ⑤ セキュリティを向上する

①レルムの追加 ② ユーザーの追加 (3) エンドポイント指定 ④ クライアント作成・設定 (5) クライアントID· クライアントシークレット共有 6 エンドポイント指定 ⑦ クライアント作成・設定 (8) クライアントID\*

クライアントシークレット共有



②検証に必要な設定をする

① レルムの追加

2 ユーザーの追加

③ エンドポイント指定
 ④ クライアント作成・設定
 ⑤ クライアントID・

クライアントシークレット共有

⑥ エンドポイント指定

⑦ クライアント作成・設定

8 クライアントID・

クライアントシークレット共有



レルムの追加
 ユーザーの追加

③ エンドポイント指定

 ④ <u>クライアント作成・設定</u>
 ⑤ クライアントID・ クライアントシークレット共有
 ⑥ エンドポイント指定
 ⑦ クライアント作成・設定

8 クライアントID・

クライアントシークレット共有



②検証に必要な設定をする

①レルムの追加

2 ユーザーの追加

③ エンドポイント指定

④ クライアント作成・設定

⑤ <u>クライアントID・</u>

<u>クライアントシークレット共有</u>

⑥ エンドポイント指定

⑦ クライアント作成・設定

8 クライアントID・

クライアントシークレット共有



①レルムの追加

2 ユーザーの追加

③ エンドポイント指定

④ クライアント作成・設定
 ⑤ クライアントID・

クライアントシークレット共有

⑥ <u>エンドポイント指定</u>

⑦ クライアント作成・設定

8 クライアントID・

クライアントシークレット共有



②検証に必要な設定をする

① レルムの追加

2 ユーザーの追加

③ エンドポイント指定

④ クライアント作成・設定

5 クライアントID・

クライアントシークレット共有

⑥ エンドポイント指定

#### ⑦ クライアント作成・設定

8 クライアントID・

クライアントシークレット共有

#### 9 スコープ作成·設定



レルムの追加
 ユーザーの追加

③ エンドポイント指定

④ クライアント作成・設定
 ⑤ クライアントID・

クライアントシークレット共有

⑥ エンドポイント指定

⑦ クライアント作成・設定

⑧ <u>クライアントID・</u>

<u>クライアントシークレット共有</u>



②検証に必要な設定をする

1 レルムの追加
 2 ユーザーの追加
 3 エンドポイント指定

④ クライアント作成・設定
 ⑤ クライアントID・

クライアントシークレット共有

⑥ エンドポイント指定

⑦ クライアント作成・設定

8 クライアントID・

クライアントシークレット共有

⑨ スコープ作成・設定





## 4. API認可をKeycloakで試す

- ① 検証環境を構築する
- 2 検証に必要な設定をする
- ③ 検証環境を動かしてみよう
- ④ トークンリフレッシュとトークン無効化を試してみる
- ⑤ セキュリティを向上する

③検証環境を動かしてみよう





## 4. API認可をKeycloakで試す

- ① 検証環境を構築する
- 2 検証に必要な設定をする
- ③ 検証環境を動かしてみよう

#### ④ トークンリフレッシュとトークン無効化を試してみる

⑤ セキュリティを向上する

④トークンリフレッシュとトークン無効化を試してみる



トークンリフレッシュ: アクセストークンの有効期限が切れたときに、ユーザの再認証無しでアクセストークンを再取得する。



トークン無効化: アクセストークンの有効期限が切れる前に、アクセストークンを無効化する。







- 書籍でカバーできてない内容について
  - ThinkIT: KeycloakのFAPI 1.0対応で実現する高度なAPIセキュリティ <u>https://thinkit.co.jp/series/10313</u> Financial-grade API (FAPI)や、ベースとなっているClient Policies等を解説
  - 公式ドキュメント <u>https://www.keycloak.org/documentation</u> 日本語訳: <u>https://keycloak-documentation.openstandia.jp/</u>
  - **コミュニティの**GitHub
    - <u>https://github.com/keycloak/</u>
  - 今後予期される変更
    - <u>https://www.keycloak.org/2021/10/keycloak-x-update.html</u>

#### **Trademarks**

- OpenID is a trademark or registered trademark of OpenID Foundation in the United States and other countries.
- GitHub is a trademark or registered trademark of GitHub, Inc. in the United States and other countries.
- Red Hat is trademark of Red Hat, Inc., registered in the United States and other countries.
- Other brand names and product names used in this material are trademarks, registered trademarks, or trade names of their respective holders.