HITACHI
Inspire the Next

※　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

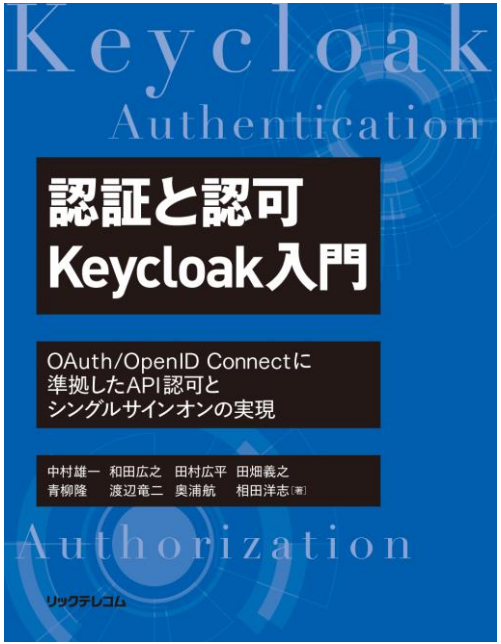
日立製作所　OSSソリューションセンタ
中村　雄一

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。**

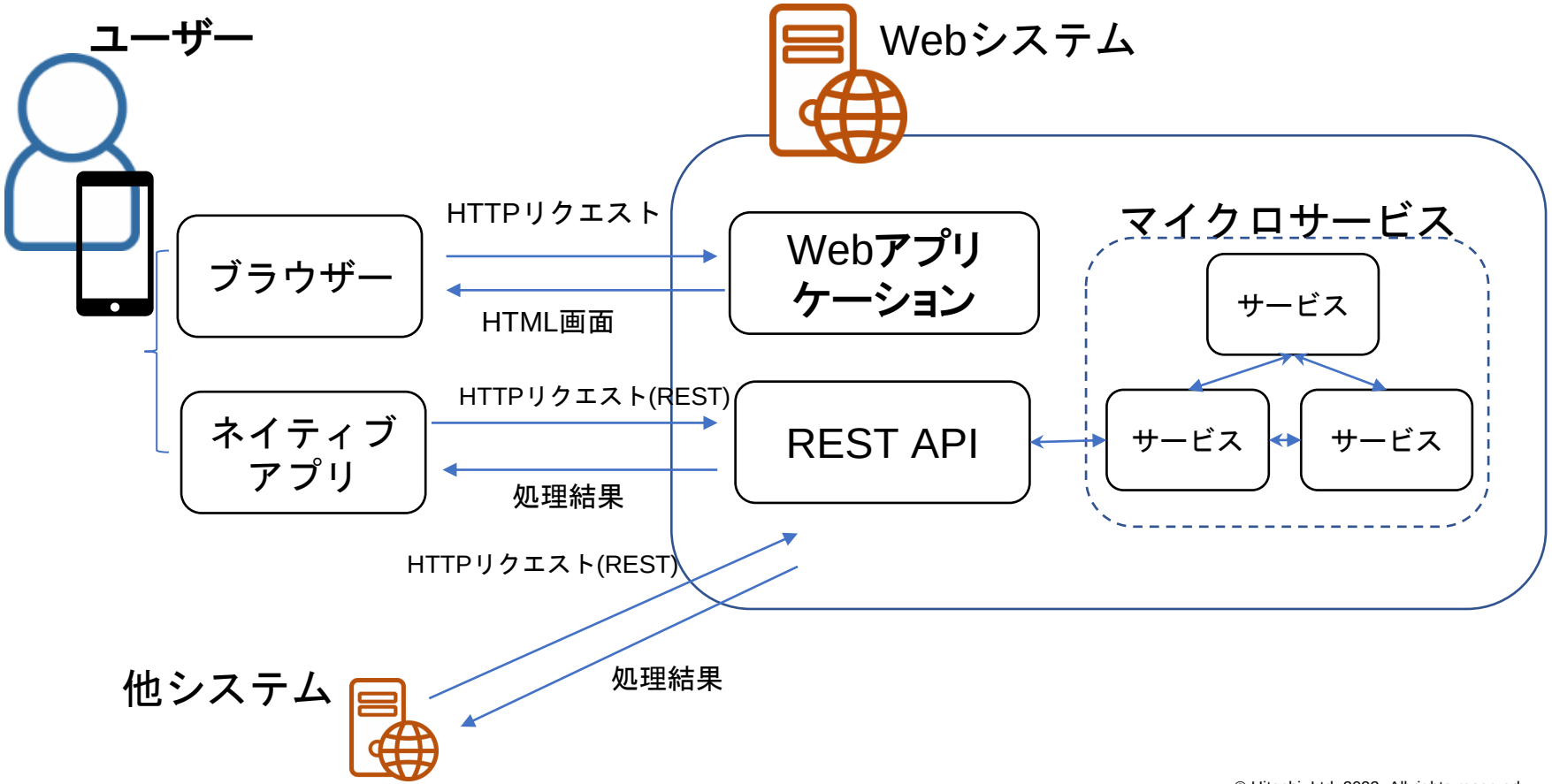
https://ric.co.jp/book/new-publication/detail/2081
https://www.amazon.co.jp/dp/4865943226/



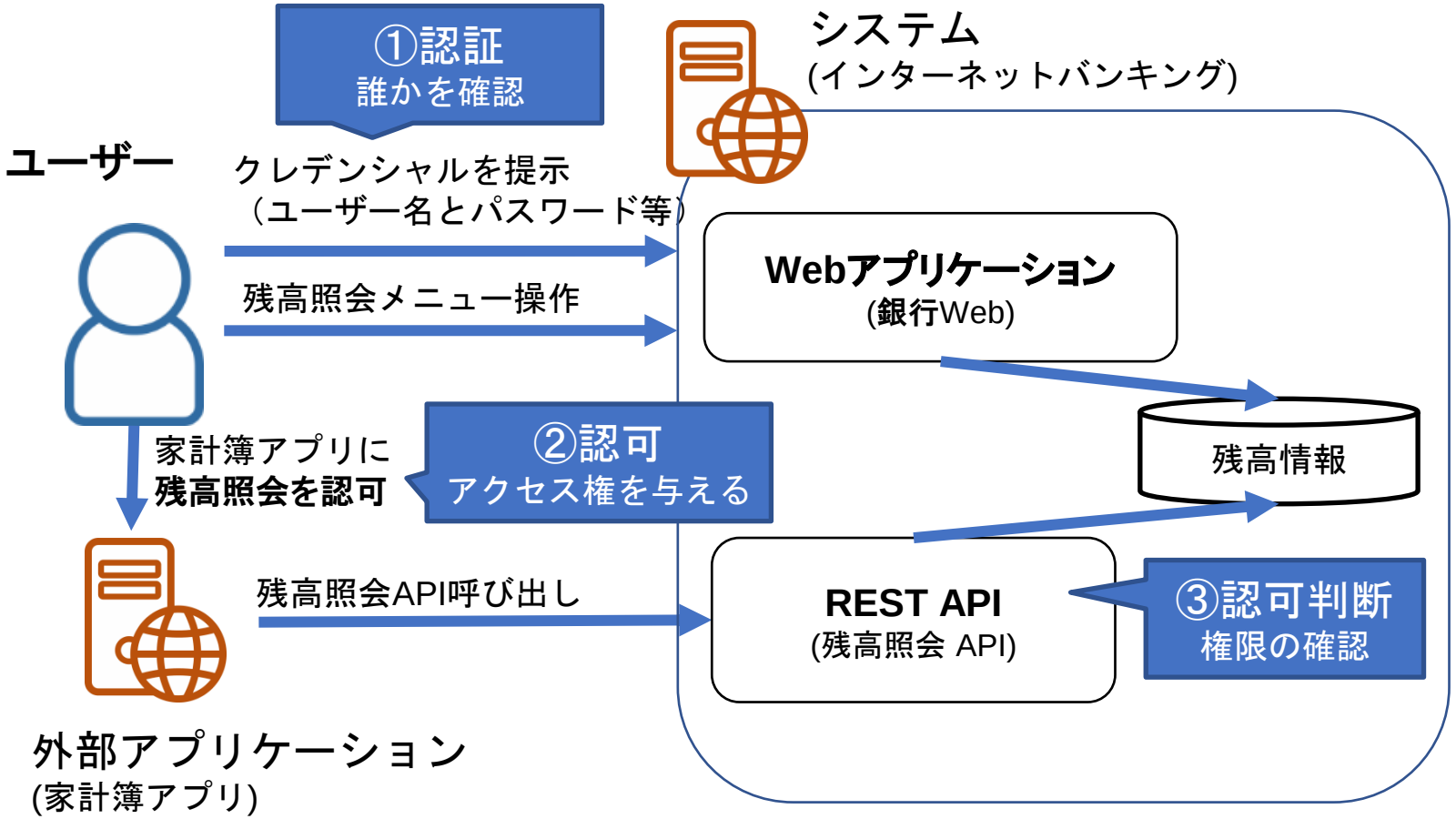
HITACHI
Inspire the Next

Contents

Webベースのシステム



認証と認可

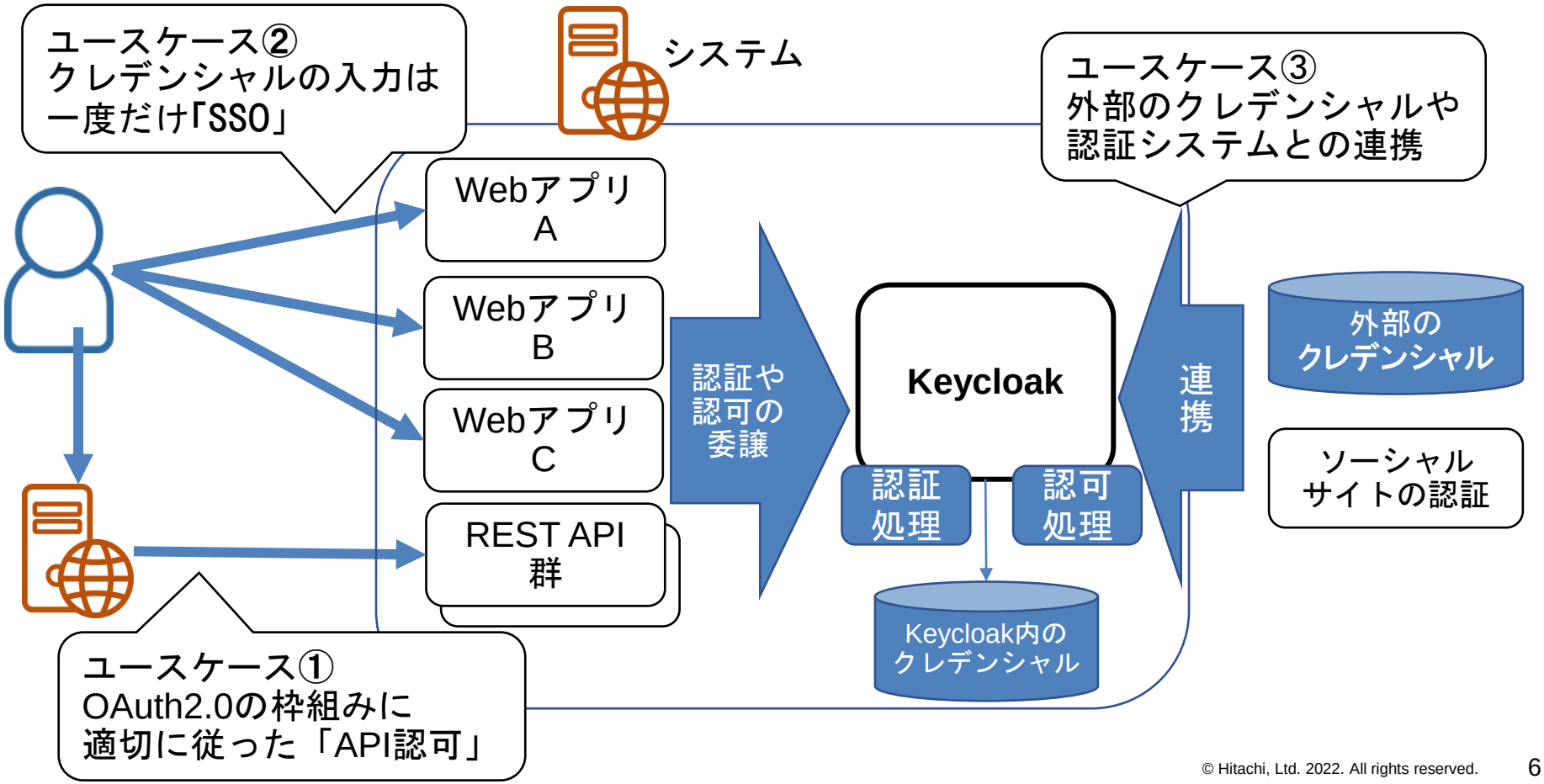


Keycloakとは

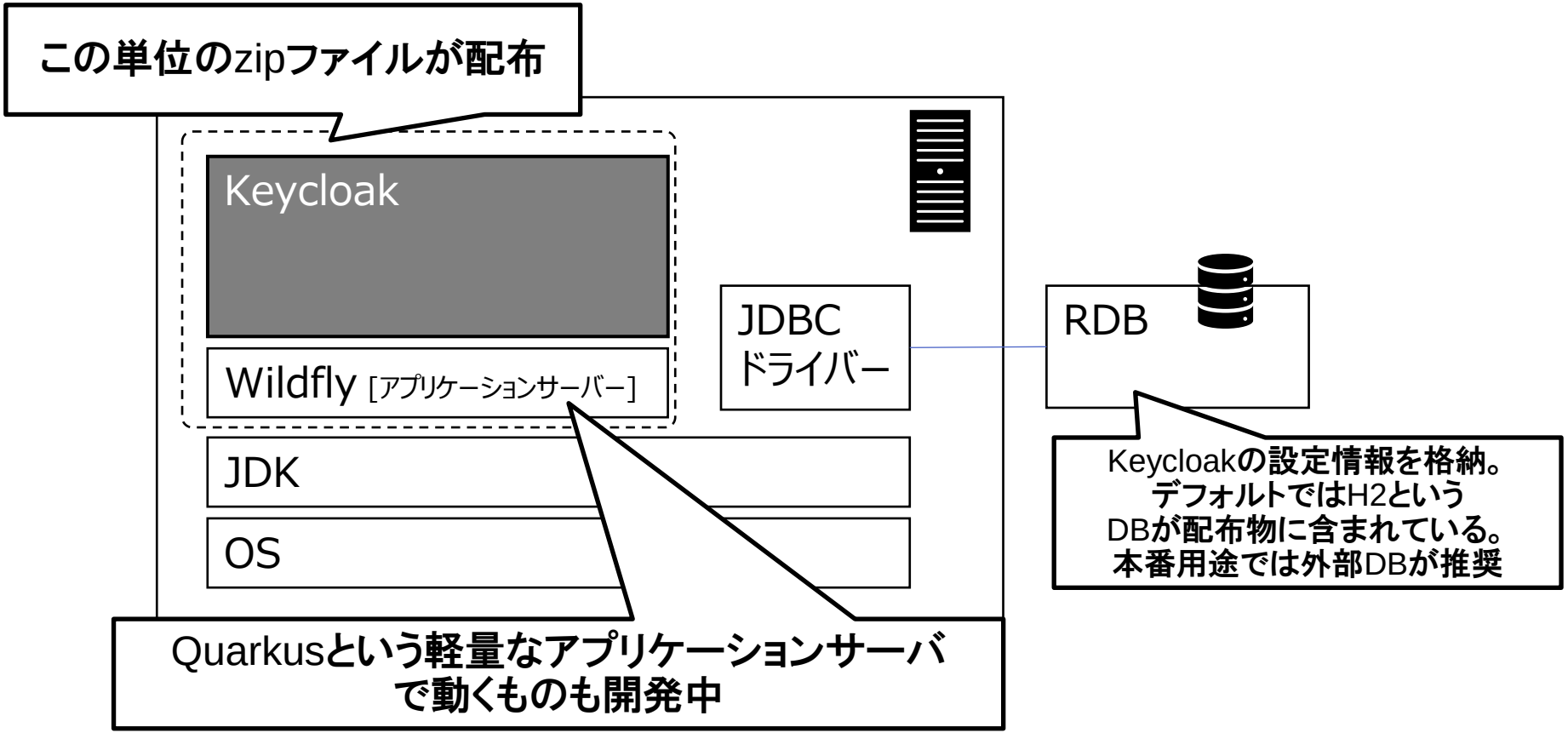
- OSSの「IAM(Identity and Access Management)」のソフトウェア
  https://www.keycloak.org/
- コミュニティが活発・オープンで、商用利用も広く進んでいる
  - 日本人のメンテナも！（乗松さん@日立）
  - 商用サポート版がRed Hat社から（Red Hat Single Sign-On)
- 最新標準にも対応。v15.0では、OpenID FoundationのFAPI, FAPI-CIBA, Open Banking Brasil という仕様にも正式対応。
  https://openid.net/certification/
- 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・認可連携プロトコル	OAuth 2.0 認可サーバ	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
カスタマイズポイント	SPI	様々な拡張を作りこむためのJavaのインタフェース
	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

Keycloakでできること



Keycloakの構成



Keycloakのインストール

HITACHI
Inspire the Next

- 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

- 公式サイトより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

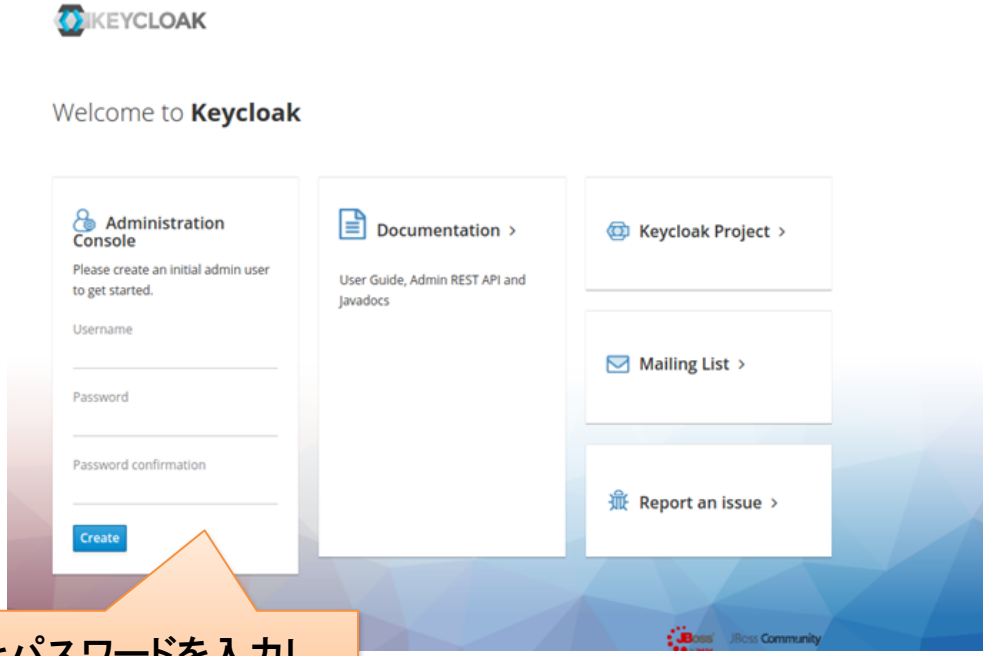
- インストール＆起動（Windowsの場合）
  zipを展開し、コマンドプロンプトで展開したディレクトリに移動し、

  .¥bin¥standalone.bat

  とするだけ。

管理者ユーザーを作成

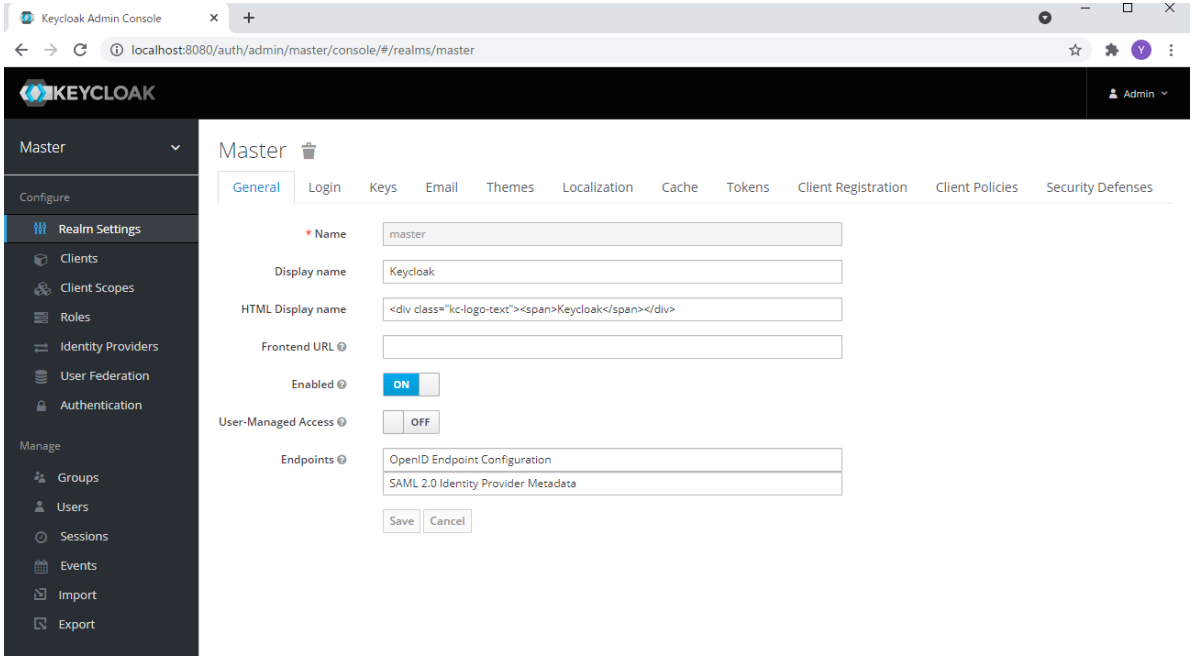
ウェルカムページ（http://localhost:8080/）をブラウザで開き、管理者ユーザーを作成



ユーザー名とパスワードを入力し
管理者ユーザーを作成

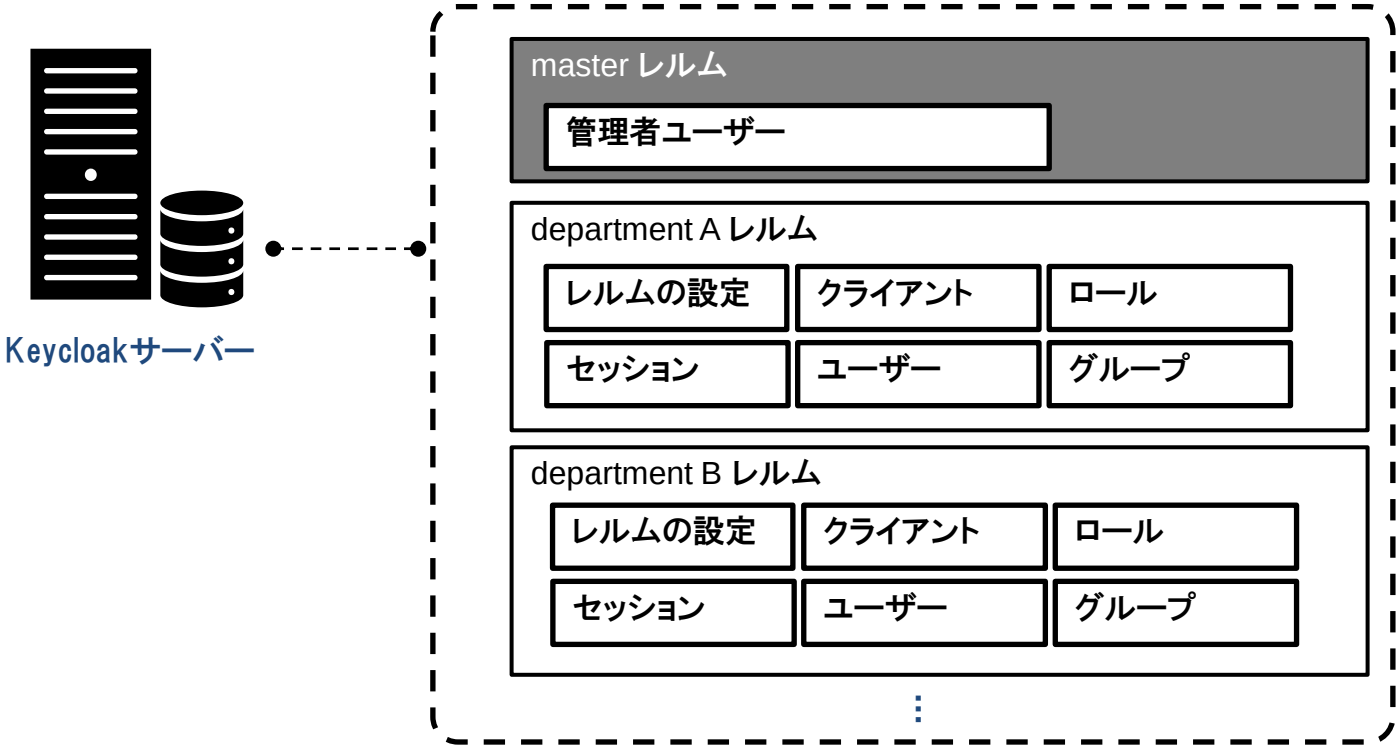
管理コンソールにログイン

- 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



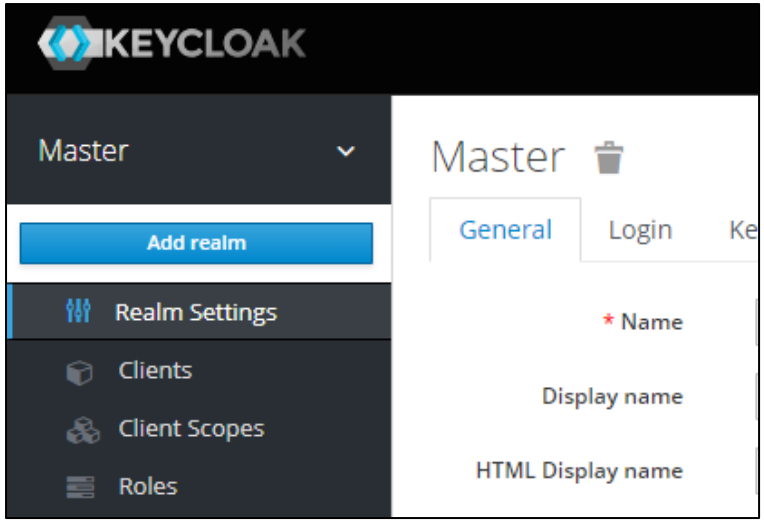
レルム作成： レルムとは

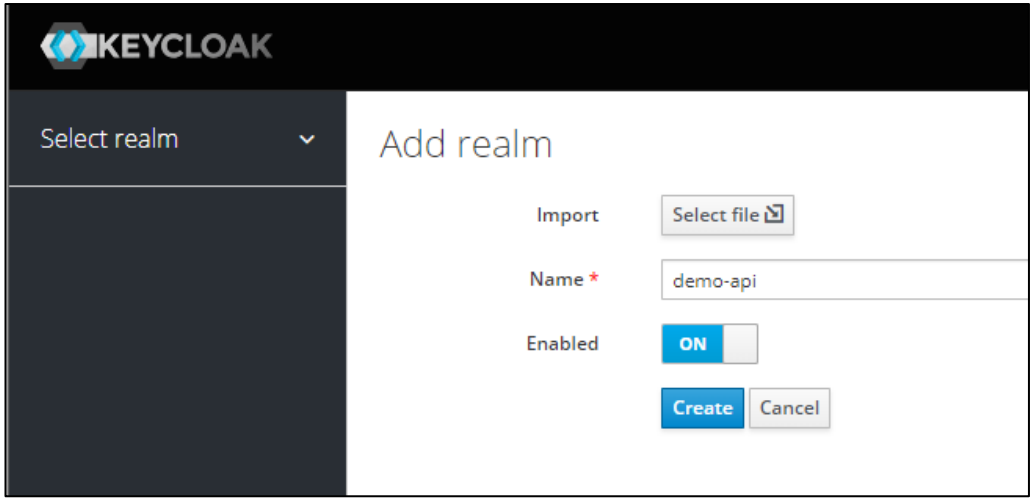
「レルム」とはKeycloak の管理単位のことで、ユーザーなどの情報はレルム単位で管理。
「master」レルムは、すべてのレルムを管理できる特別なレルム（先ほどは「master」レルム）
任意にレルムを作成して管理ができ（部署単位、用途単位等）、マルチテナントのような使い方もできる。



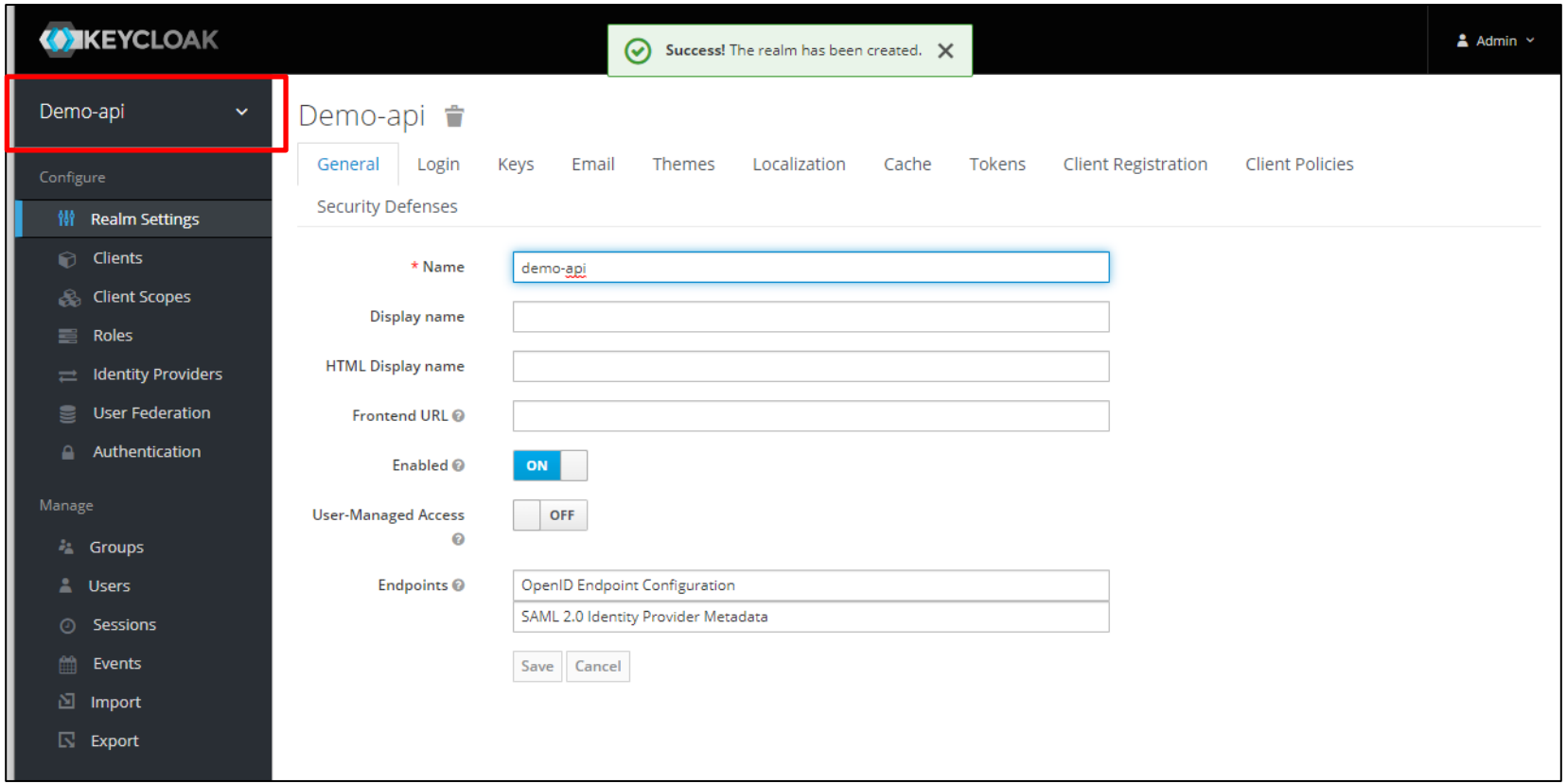
レルムの作成

API認可の動作確認用に、「demo-api」レルムを作成





レルムを確認



HITACHI
Inspire the Next

Contents

API認可とは

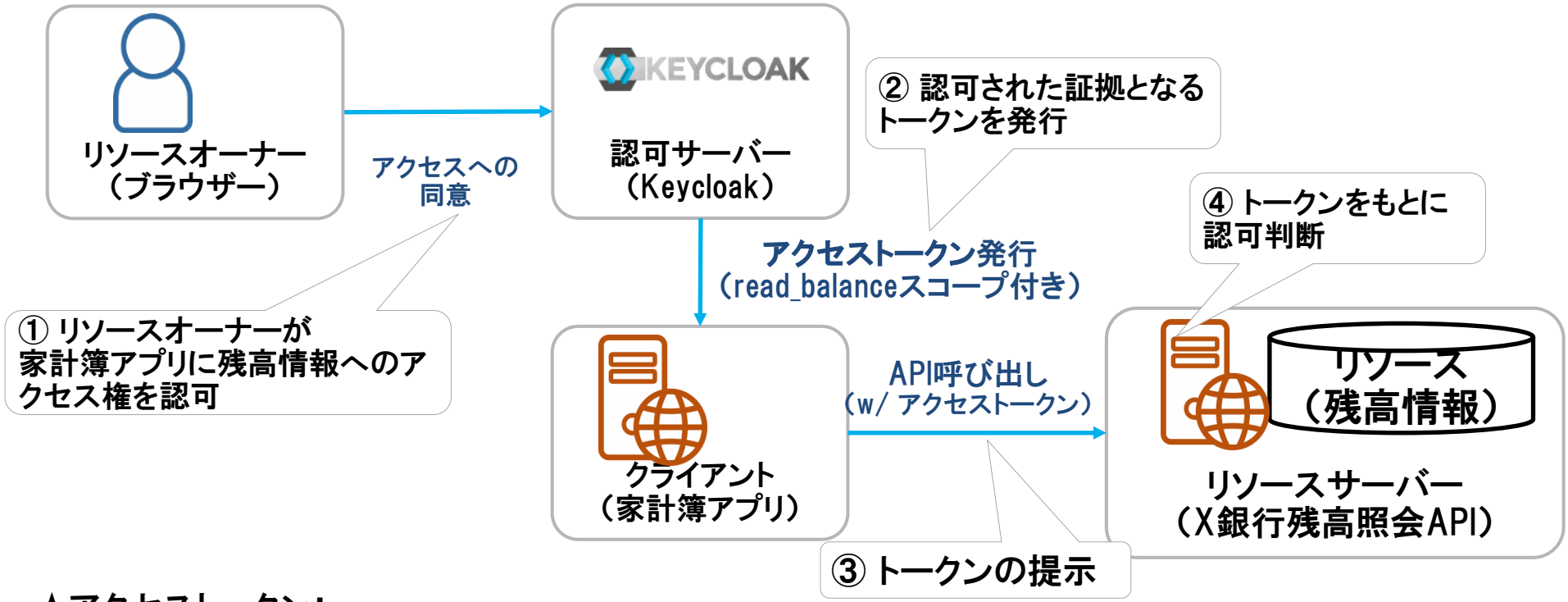
HITACHI
Inspire the Next

- 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(3rd party application)に認可するためのプロトコル
  RFC6749で規定

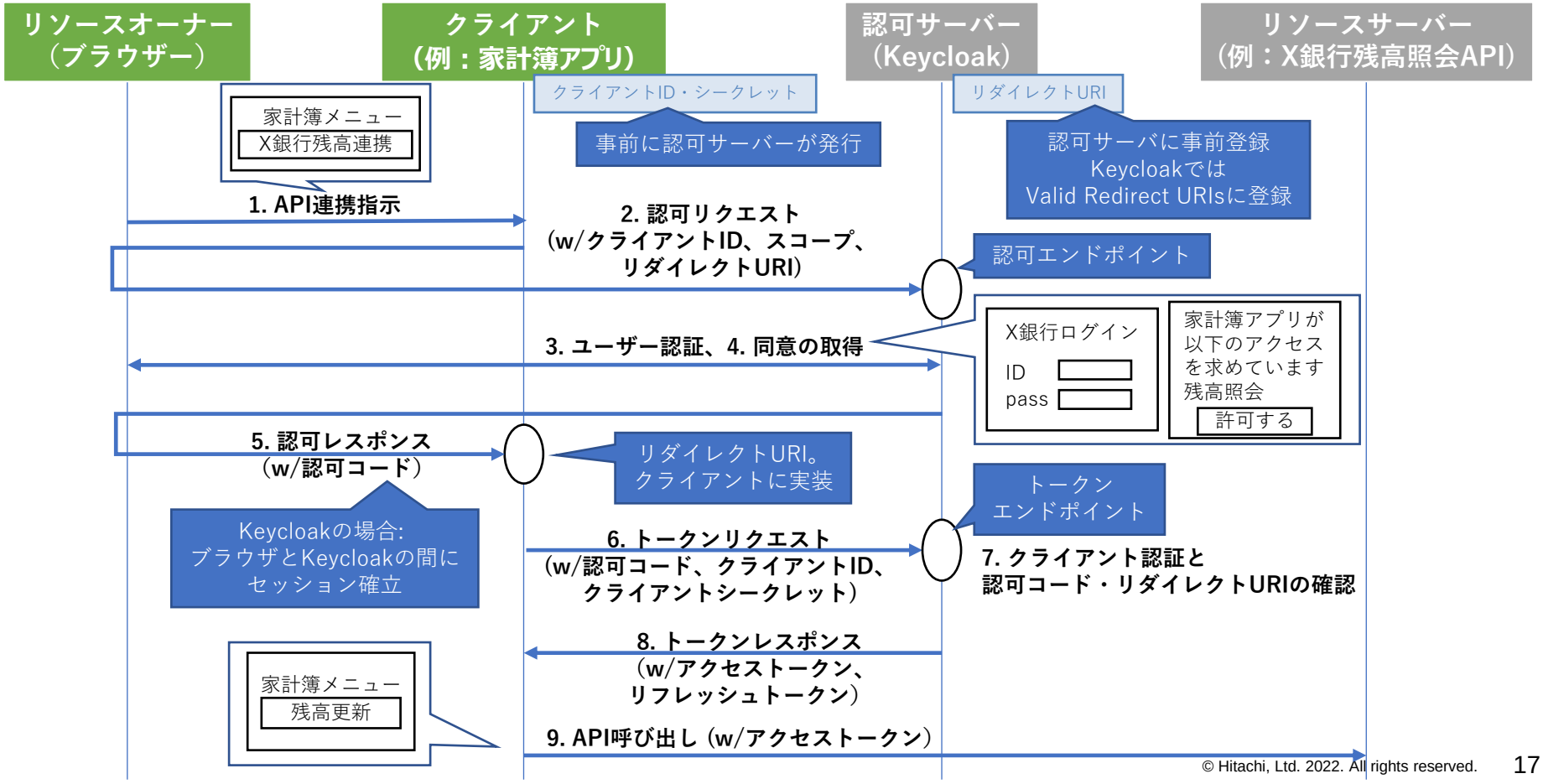
OAuthの登場人物とAPI認可



★アクセストークン：
API を呼び出すときに使う「認可された証拠」
リソースアクセスがクライアントに認可されたかを示す 「スコープ（scope）」が関連付けられている。
OAuthではアクセストークン発行のためのフローをいくつか規定

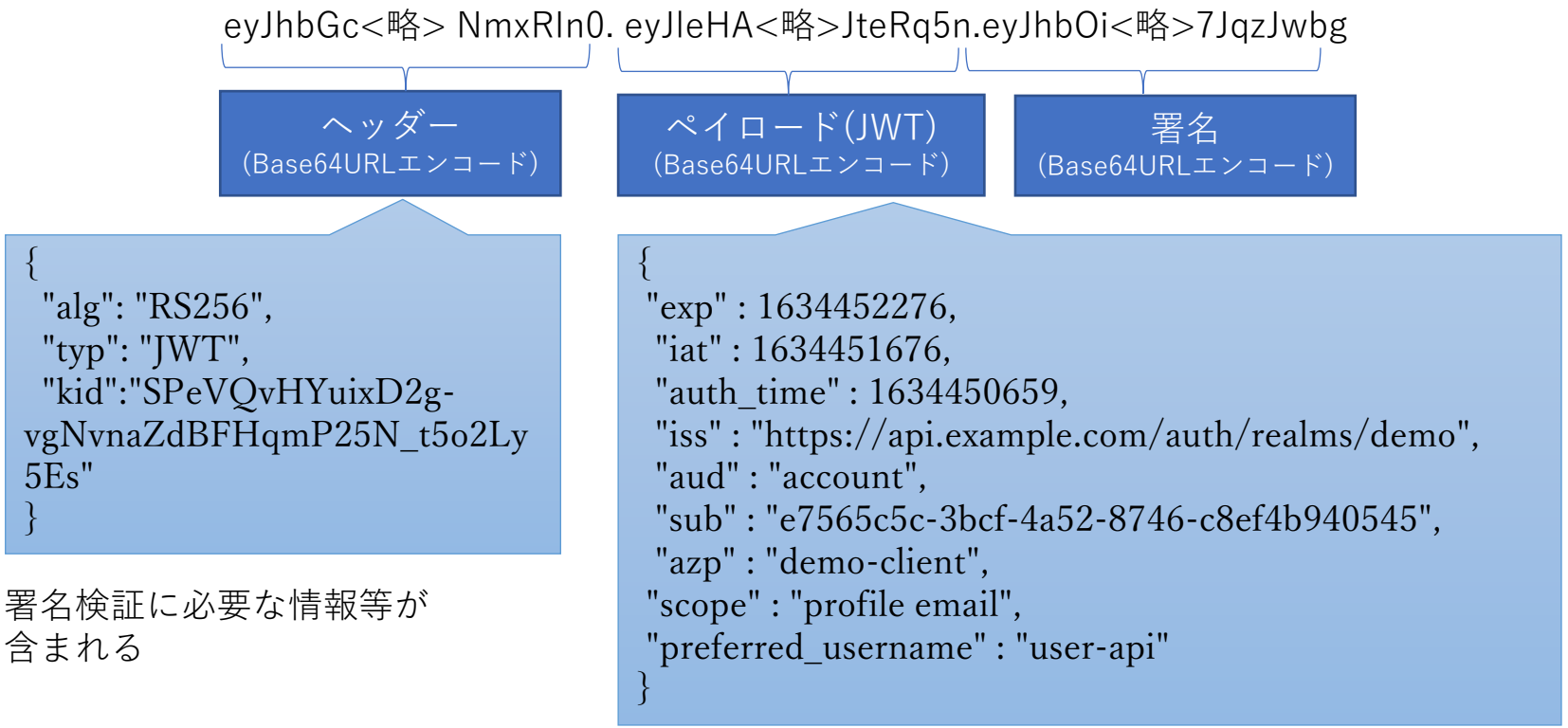
OAuthの典型的なフロー ： 認可コードフロー

HITACHI
Inspire the Next



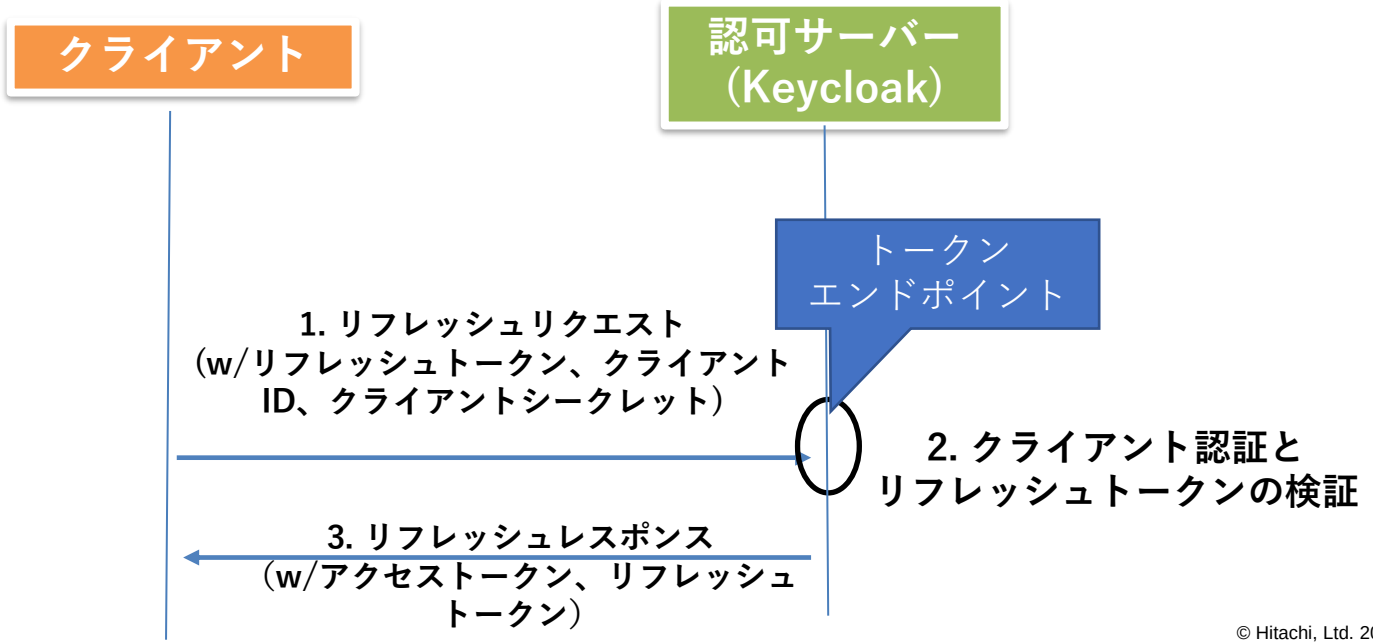
Keycloakにおけるアクセストークン

アクセストークンの形式は任意だがKeycloakではJWSを採用



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

HITACHI
Inspire the Next

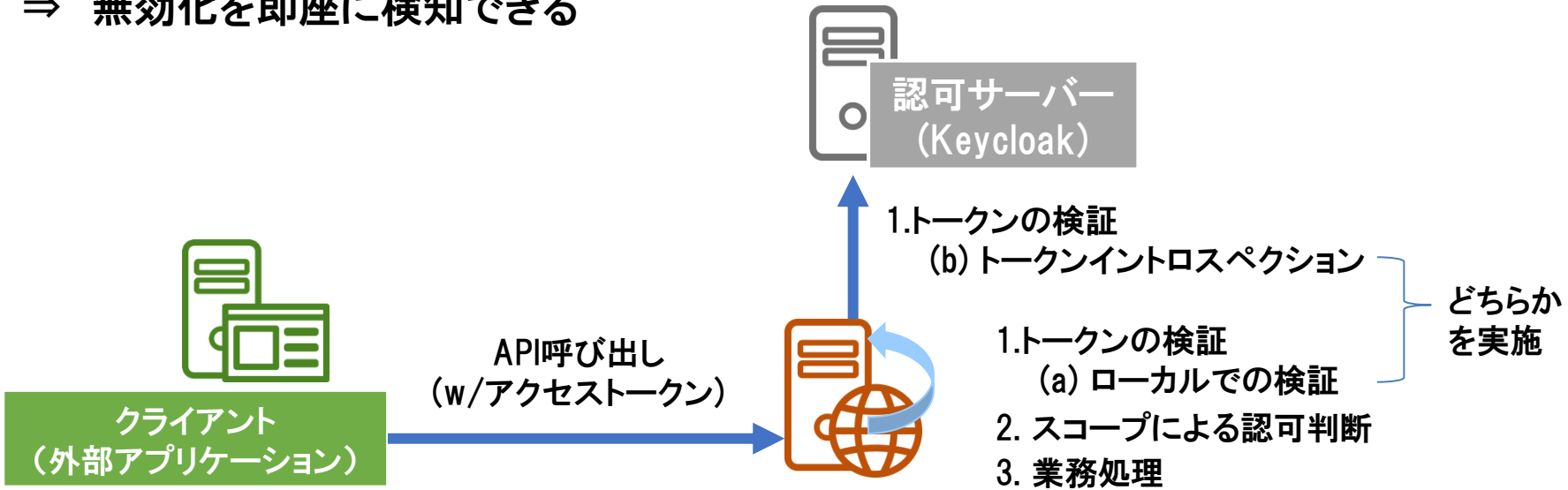
- 有効期限内であっても、アクセストークンやリフレッシュトークンは
  無効にすることができる。
  （セキュリティ上の理由や、リソースオーナーがAPI連携解除したい場合）

- 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

- ローカルでの検証（署名のチェックなど）
  ⇒ 無効化を即座に検知できない
- トークンイントロスペクション （RFC7662で規定）
  ⇒ 無効化を即座に検知できる

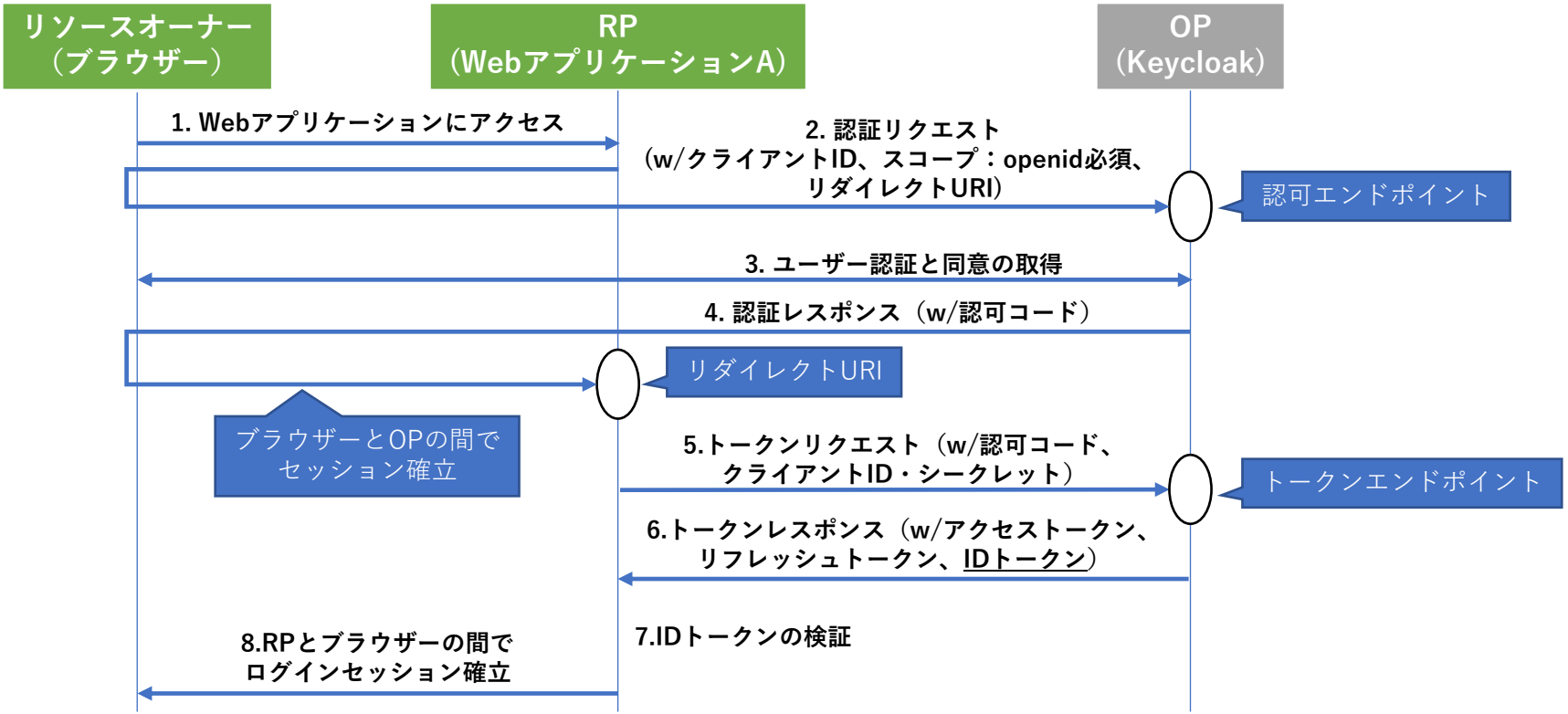


トークンの検証などの共通処理は
APIゲートウェイで代行することもできる

OIDC (OpenID Connect)

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。



HITACHI
Inspire the Next

Contents

Keycloakの基本概念

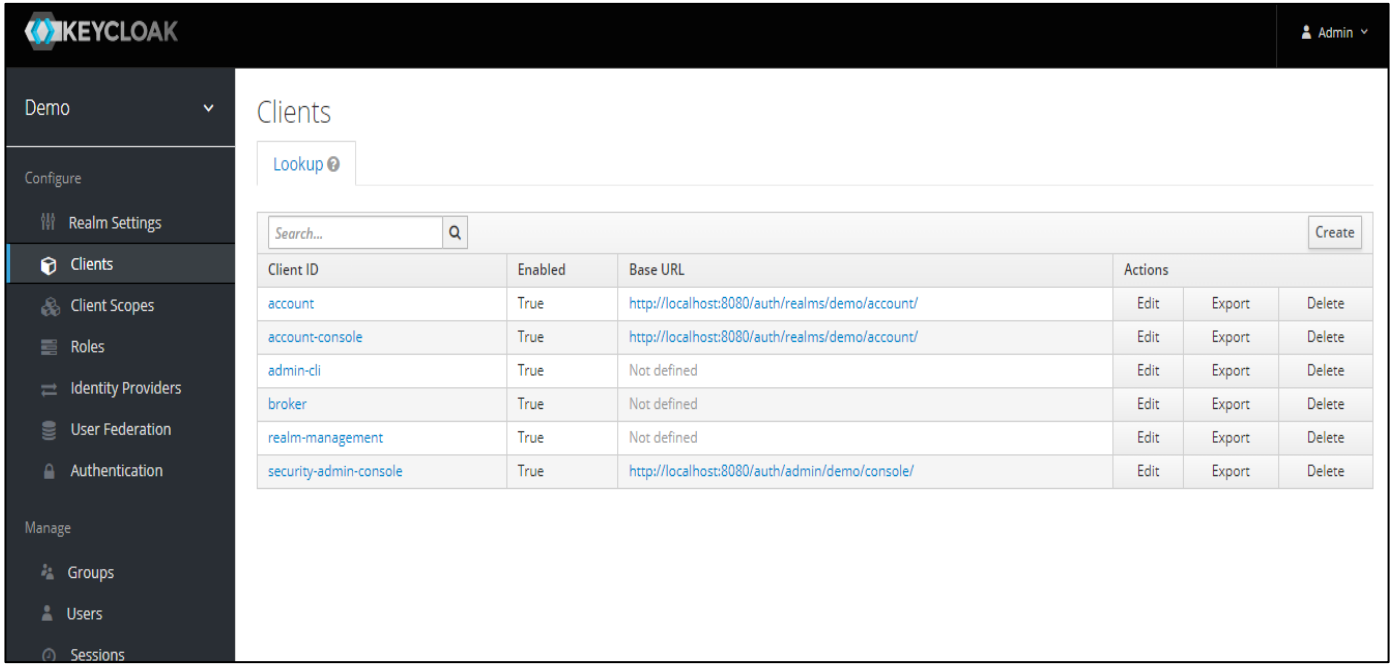
Keycloakの設定を行うためには、特有の用語の理解が必要。

「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。

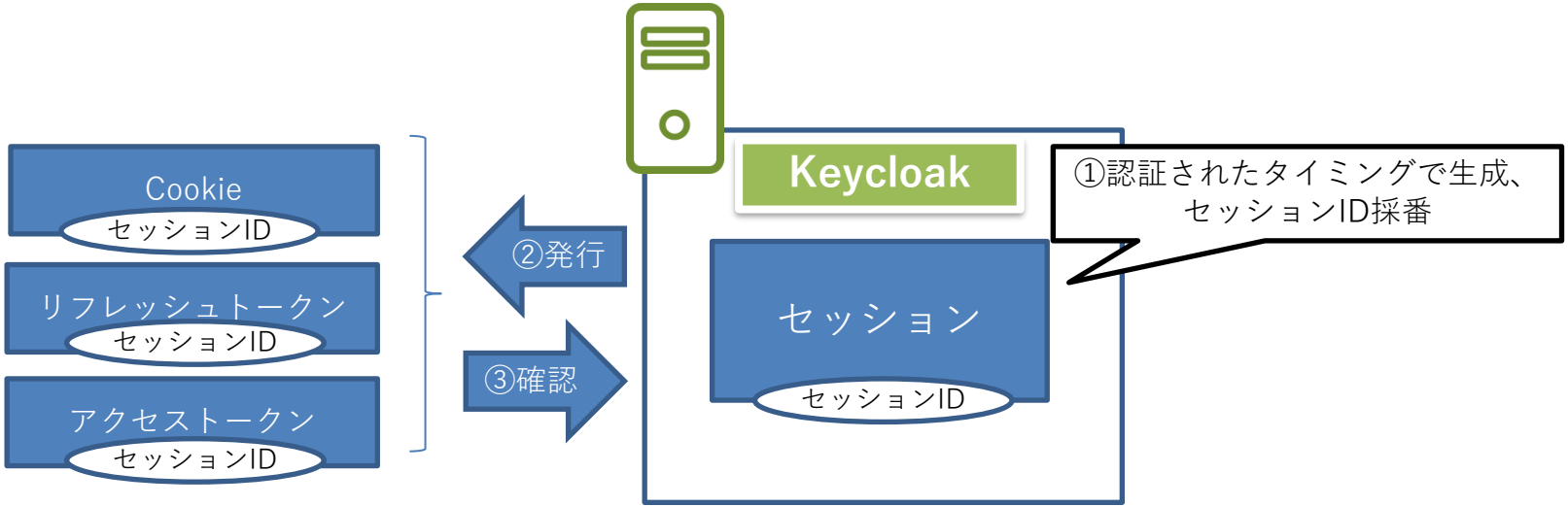
クライアント

- 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。



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。



HITACHI
Inspire the Next

Contents

HITACHI
Inspire the Next

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

HITACHI
Inspire the Next

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

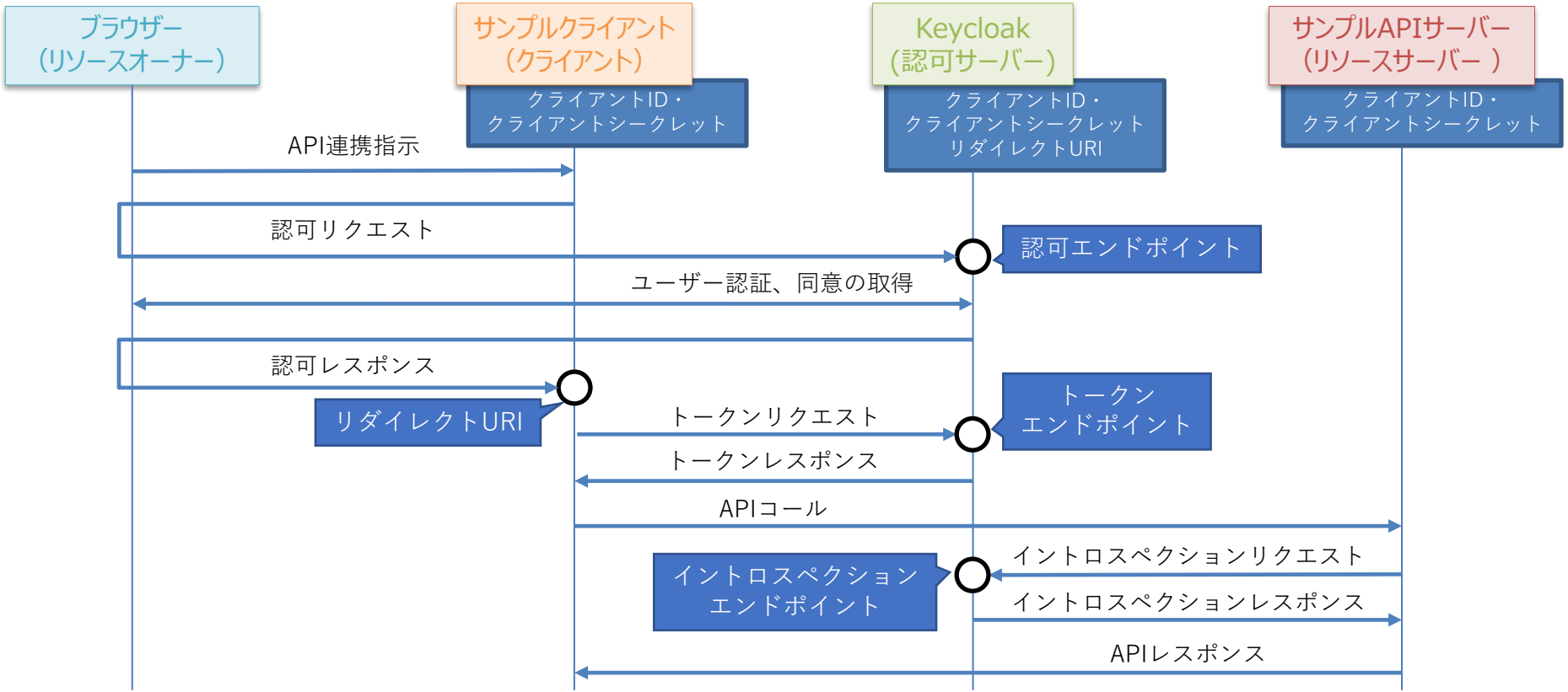
③ 検証環境を動かしてみよう

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

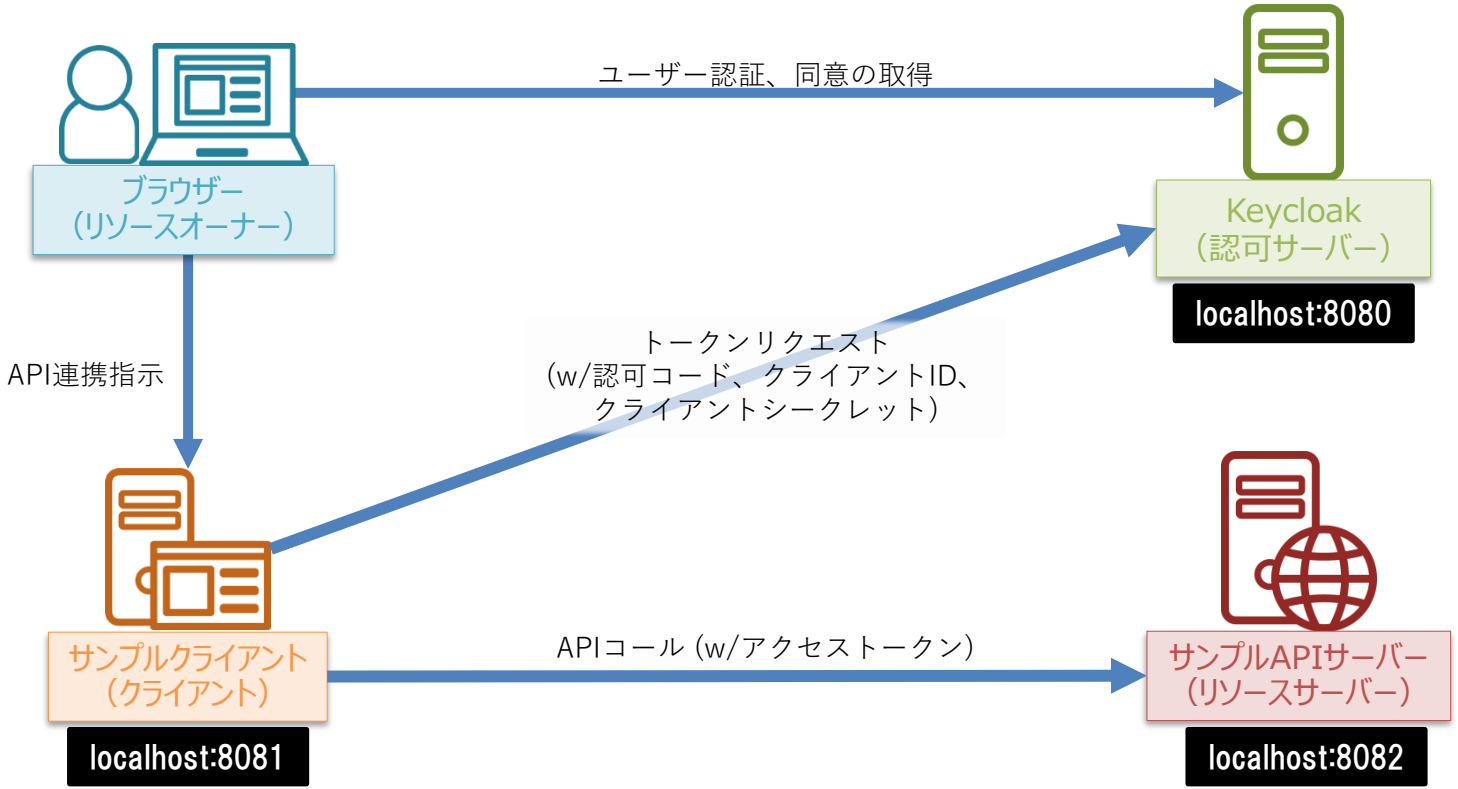
①検証環境を構築する

- 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



HITACHI
Inspire the Next

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

②検証に必要な設定をする

HITACHI
Inspire the Next

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

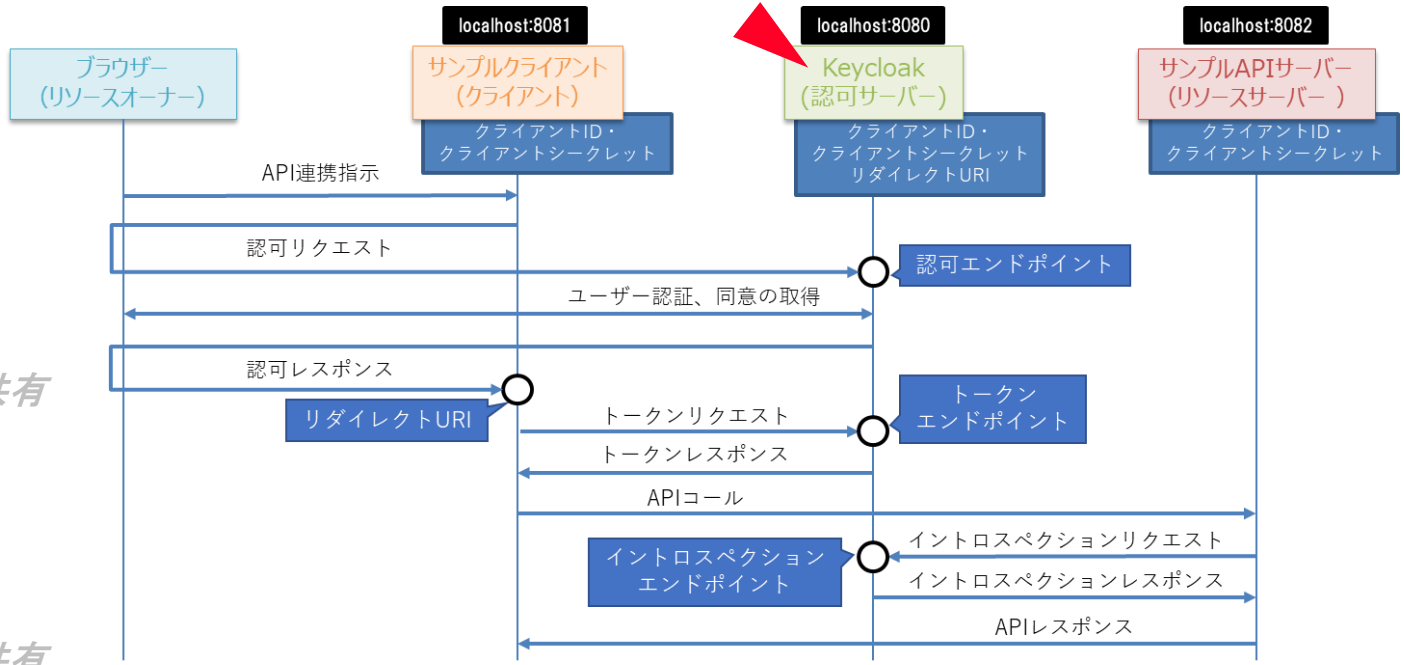
⑤ クライアントID・
クライアントシークレット共有

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

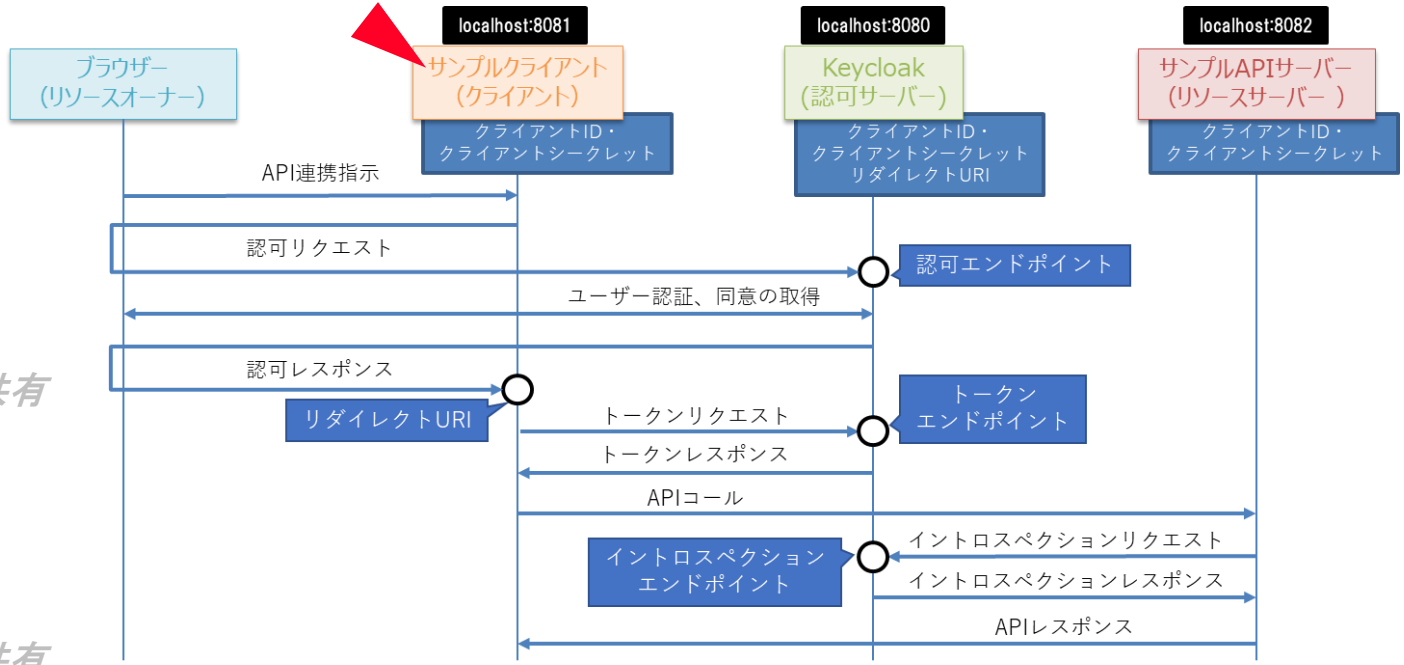
⑨ スコープ作成・設定



②検証に必要な設定をする

HITACHI
Inspire the Next

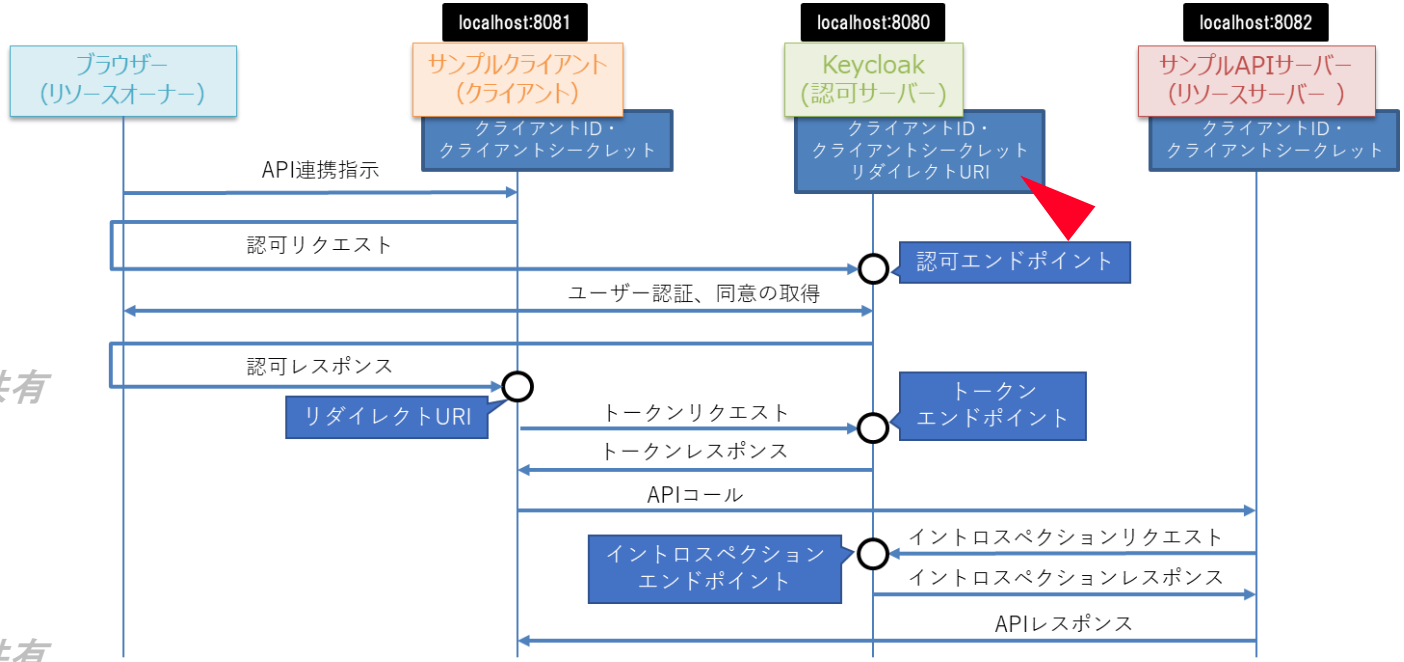
① レルムの追加
② ユーザーの追加
③ エンドポイント指定
④ クライアント作成・設定
⑤ クライアントID・
クライアントシークレット共有
⑥ エンドポイント指定
⑦ クライアント作成・設定
⑧ クライアントID・
クライアントシークレット共有
⑨ スコープ作成・設定



②検証に必要な設定をする

HITACHI
Inspire the Next

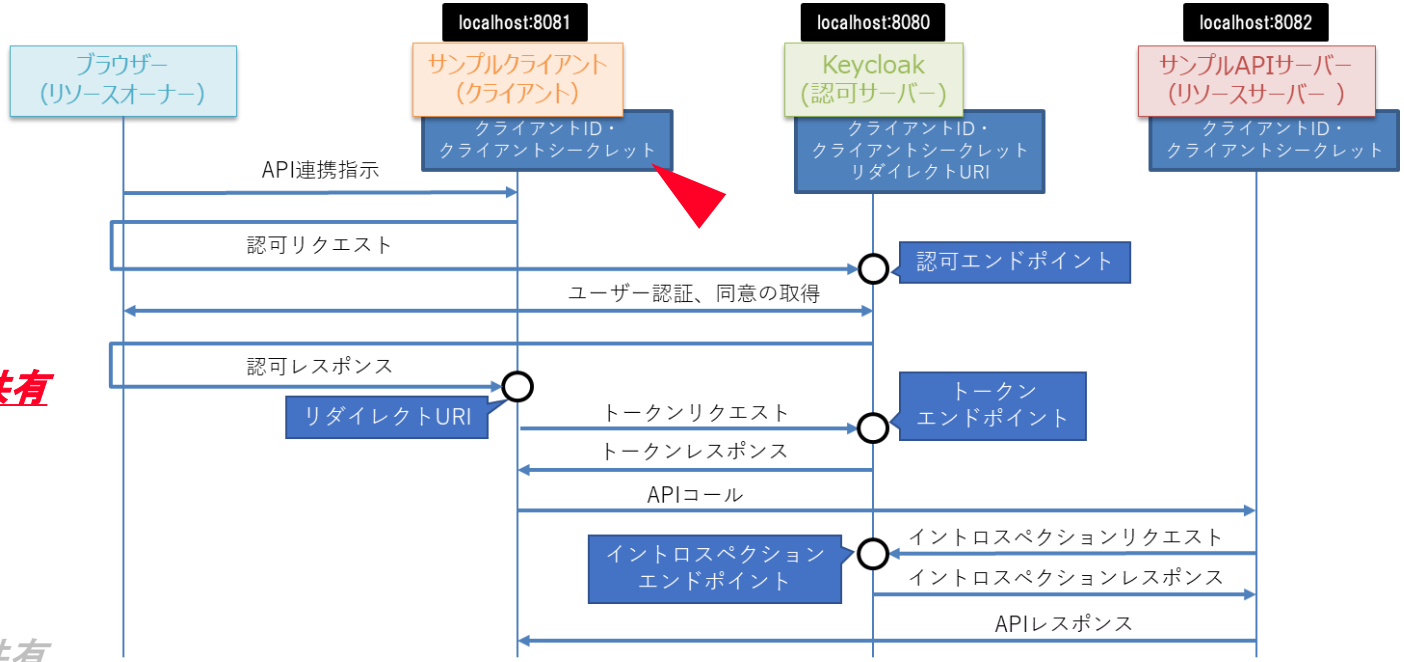
① レルムの追加
② ユーザーの追加
③ エンドポイント指定
④ クライアント作成・設定
⑤ クライアントID・クライアントシークレット共有
⑥ エンドポイント指定
⑦ クライアント作成・設定
⑧ クライアントID・クライアントシークレット共有
⑨ スコープ作成・設定



②検証に必要な設定をする

HITACHI
Inspire the Next

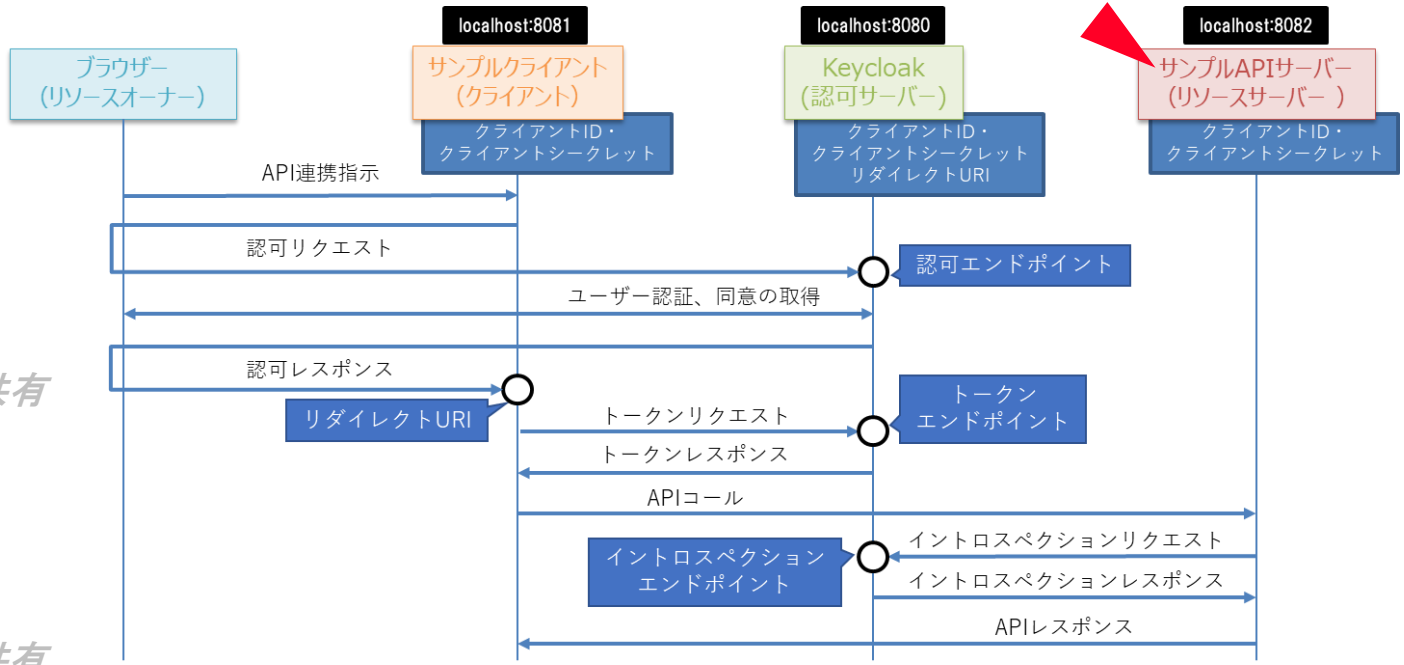
① レルムの追加
② ユーザーの追加
③ エンドポイント指定
④ クライアント作成・設定
⑤ **クライアントID・クライアントシークレット共有**
⑥ エンドポイント指定
⑦ クライアント作成・設定
⑧ クライアントID・クライアントシークレット共有
⑨ スコープ作成・設定



②検証に必要な設定をする

HITACHI
Inspire the Next

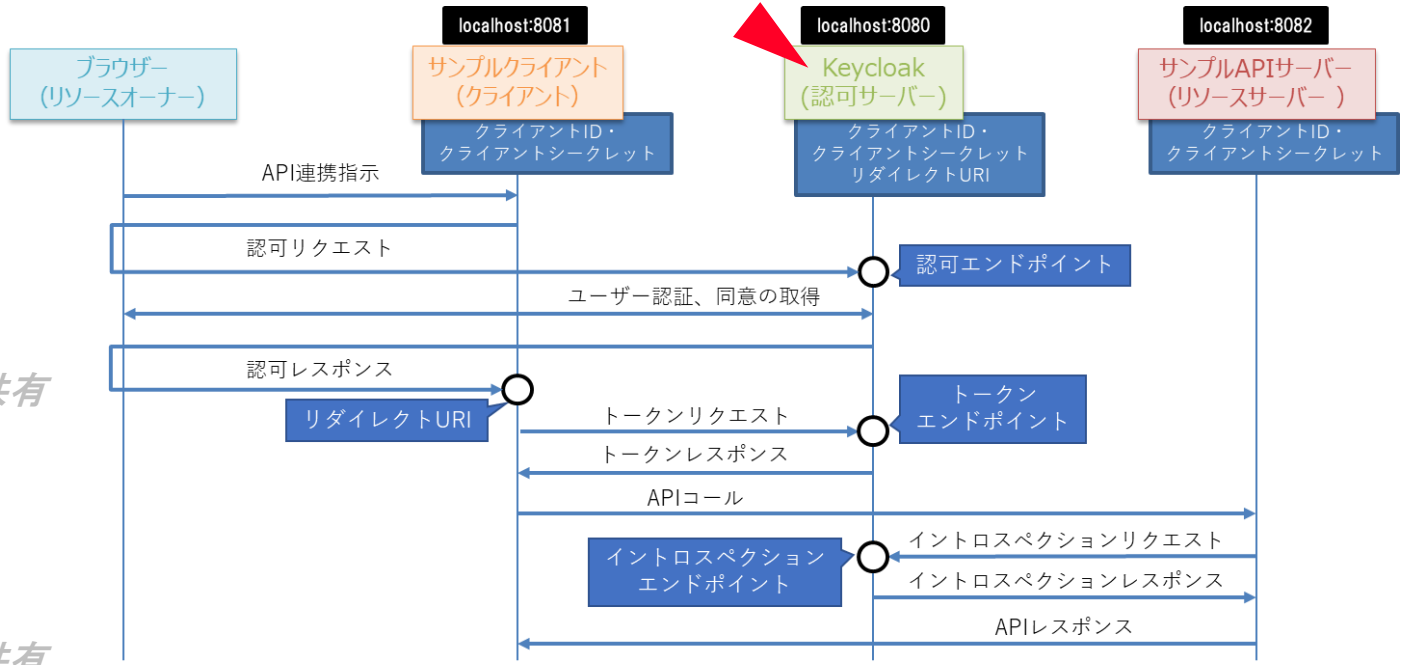
① レルムの追加
② ユーザーの追加
③ エンドポイント指定
④ クライアント作成・設定
⑤ クライアントID・
クライアントシークレット共有
⑥ エンドポイント指定
⑦ クライアント作成・設定
⑧ クライアントID・
クライアントシークレット共有
⑨ スコープ作成・設定



②検証に必要な設定をする

HITACHI
Inspire the Next

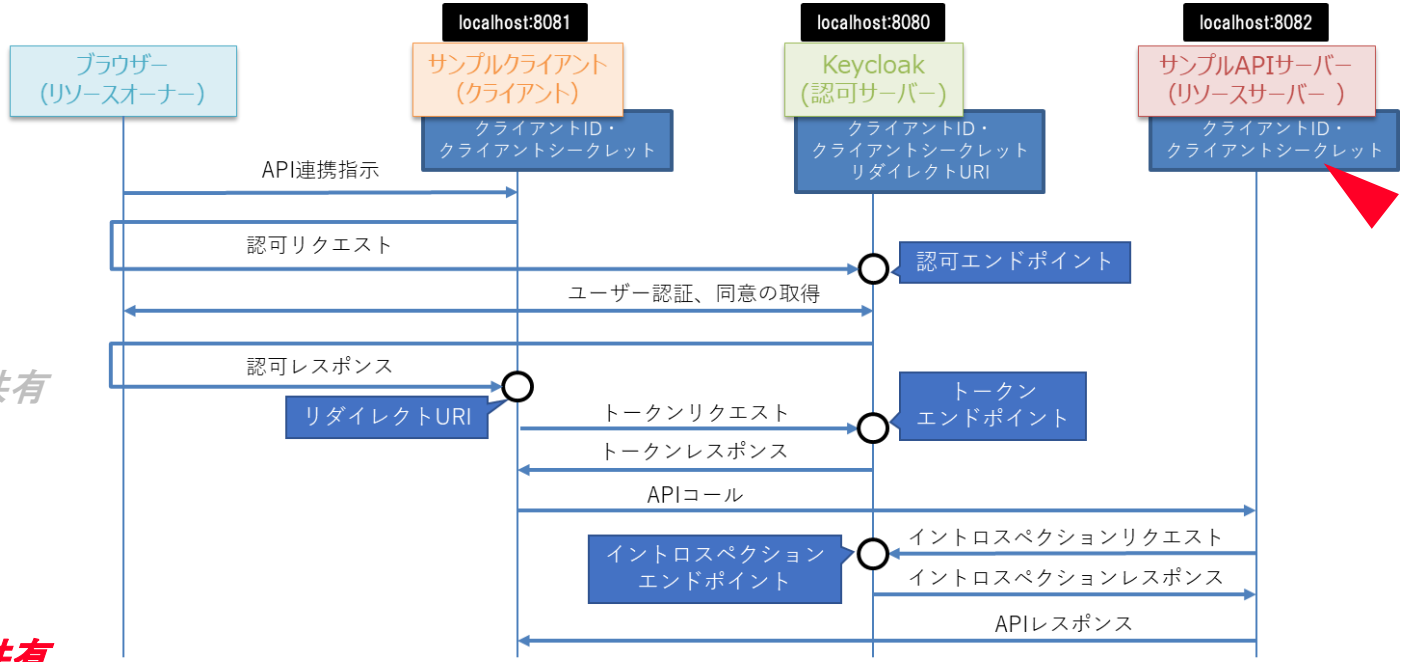
① レルムの追加
② ユーザーの追加
③ エンドポイント指定
④ クライアント作成・設定
⑤ クライアントID・
クライアントシークレット共有
⑥ エンドポイント指定
⑦ クライアント作成・設定
⑧ クライアントID・
クライアントシークレット共有
⑨ スコープ作成・設定



②検証に必要な設定をする

HITACHI
Inspire the Next

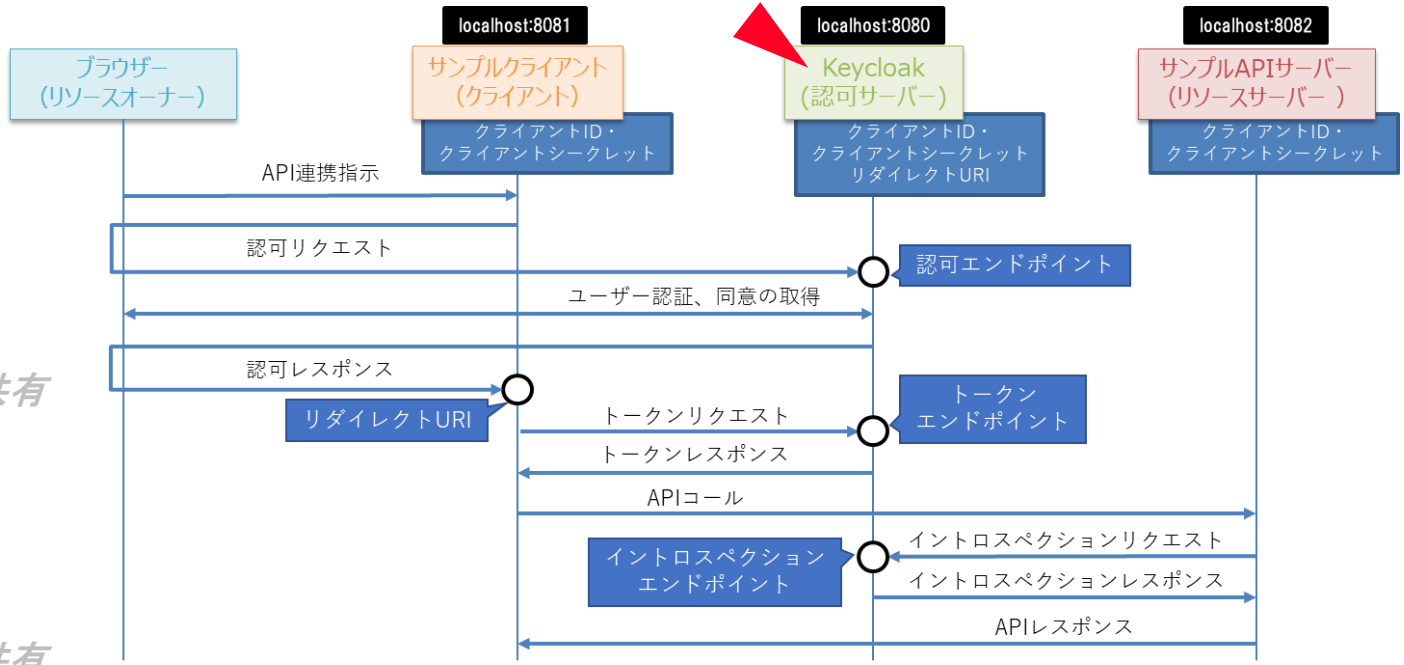
① レルムの追加
② ユーザーの追加
③ エンドポイント指定
④ クライアント作成・設定
⑤ クライアントID・
クライアントシークレット共有
⑥ エンドポイント指定
⑦ クライアント作成・設定
⑧ クライアントID・
クライアントシークレット共有
⑨ スコープ作成・設定



②検証に必要な設定をする

HITACHI
Inspire the Next

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



HITACHI
Inspire the Next

4. API認可をKeycloakで試す

① 検証環境を構築する

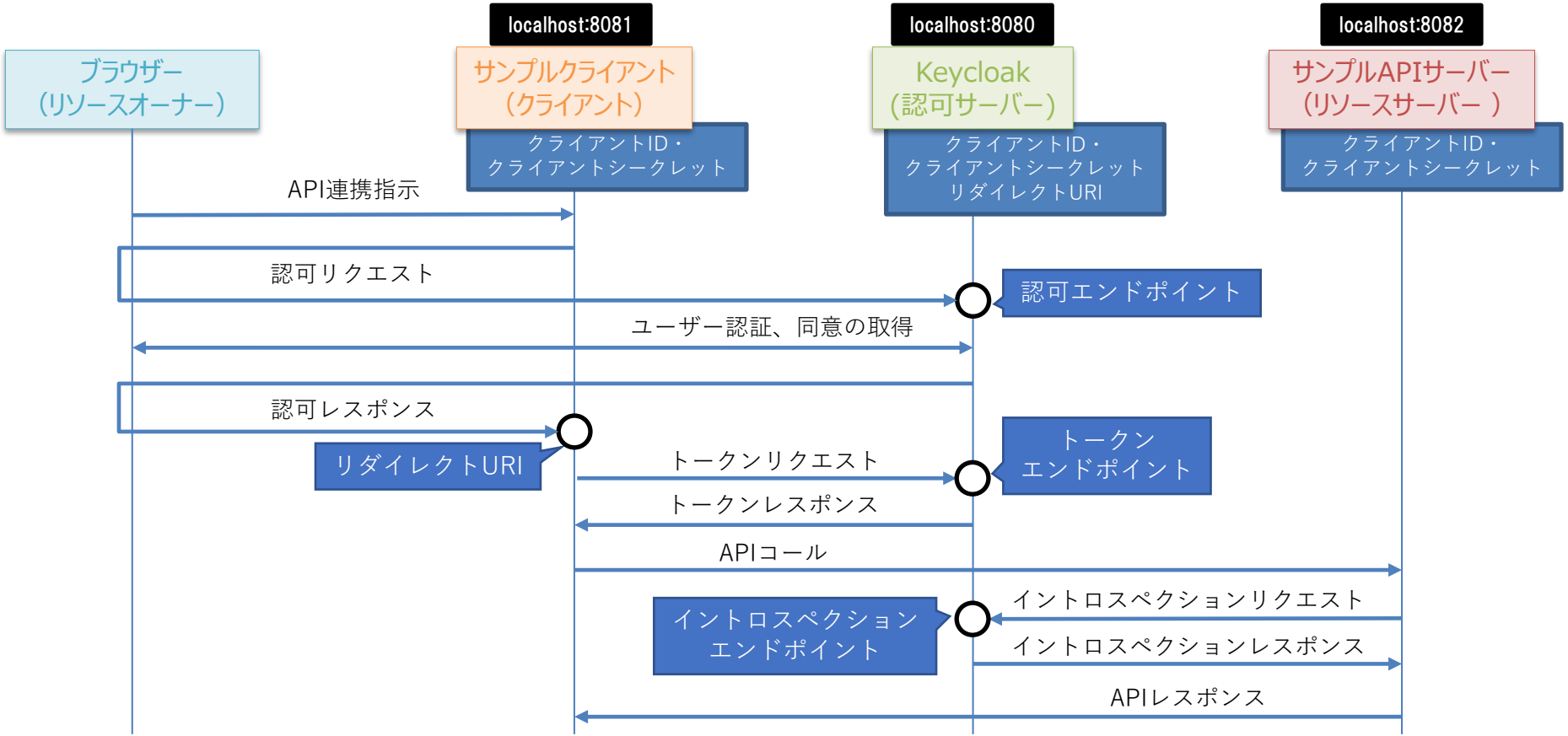
② 検証に必要な設定をする

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



HITACHI
Inspire the Next

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

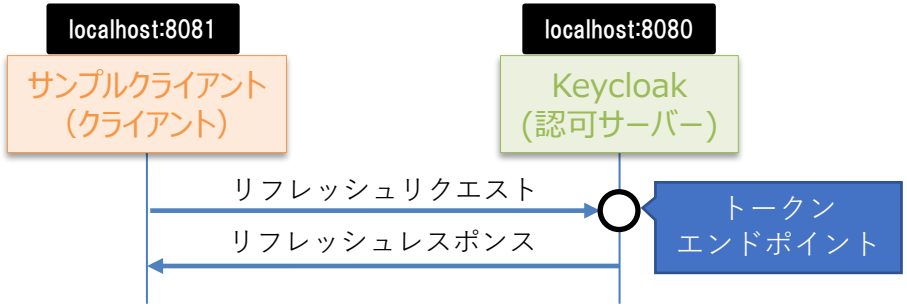
④ トークンリフレッシュとトークン無効化を試してみる

⑤ セキュリティを向上する

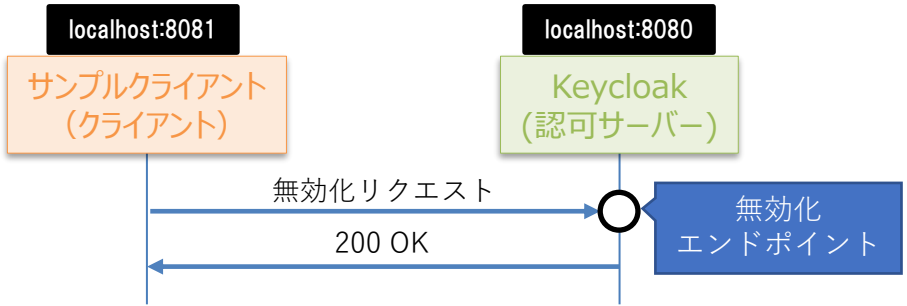
④トークンリフレッシュとトークン無効化を試してみる

HITACHI
Inspire the Next

- **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。



参考

- 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
    https://thinkit.co.jp/series/10313
    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
    https://www.keycloak.org/documentation
    日本語訳： https://keycloak-documentation.openstandia.jp/
  - コミュニティのGitHub
    - https://github.com/keycloak/
  - 今後予期される変更
    - https://www.keycloak.org/2021/10/keycloak-x-update.html

Trademarks

HITACHI
Inspire the Next

- 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.

HITACHI

Inspire the Next