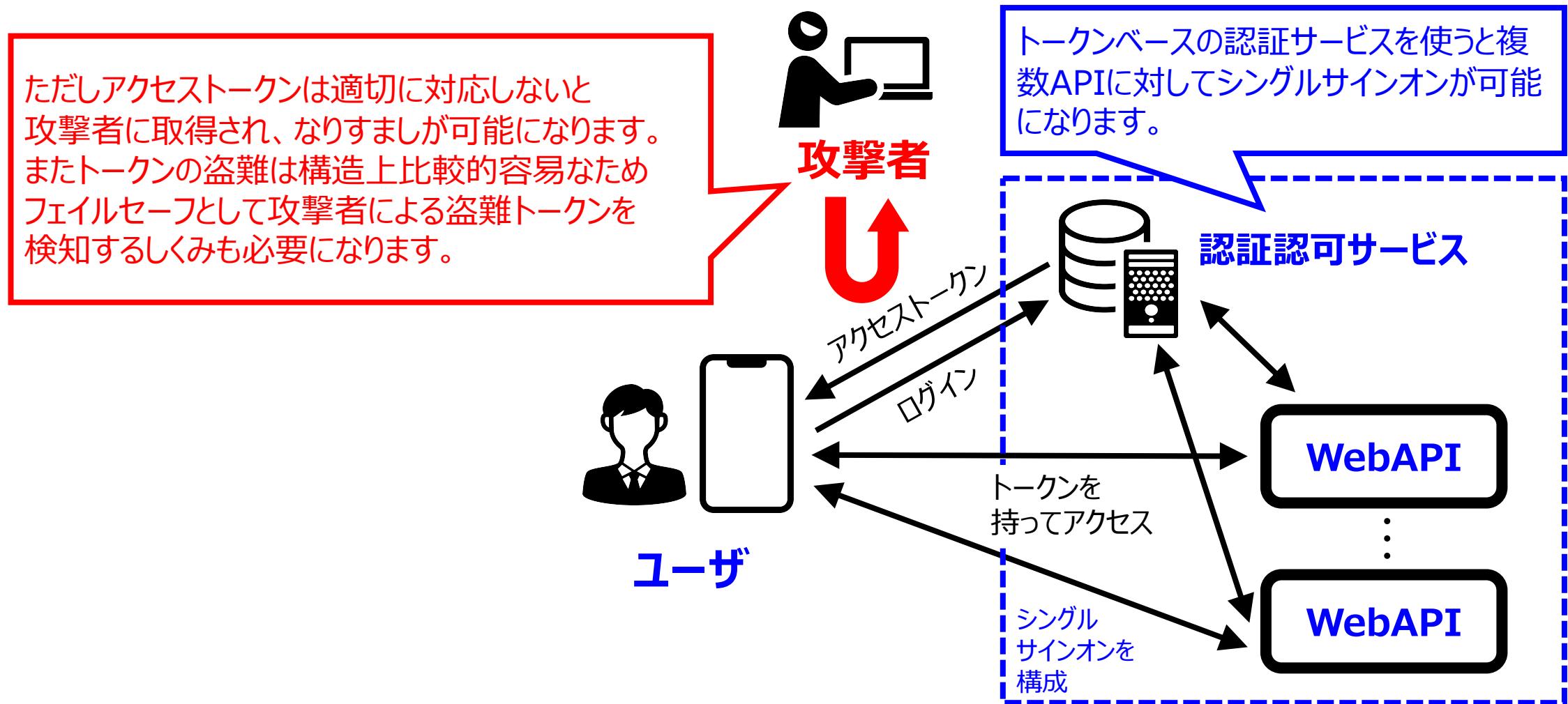


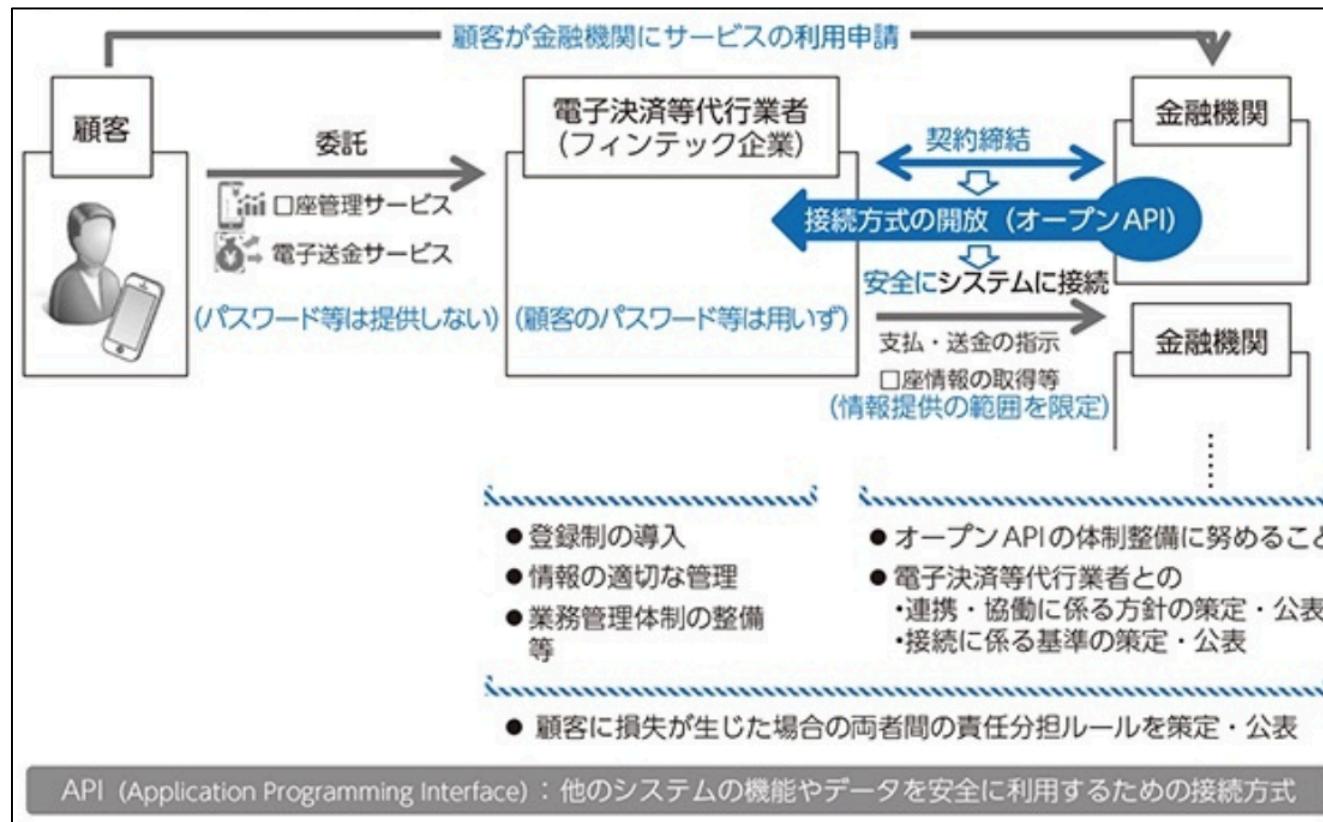
# WebAPIとトークンベース認証

WebAPIではステートレスな(状態を持たない)実装が推奨されるので、セッションIDをもつクッキーとサーバに保持されたセッション情報による認証ではなく、認証に成功したという情報を持つトークンを使った**トークンベース認証**と相性がよいです。ただし、攻撃者によるトークンの盗難や盗難トークンの利用を検知してはじくフェイルセーフのしくみの採用が不可欠です。



# オープンAPIとは？

- 2017年5月に改正銀行法が成立し、「銀行等はオープンAPIに係る体制整備に努めること」との努力義務が課された。また、本法ではこのAPIを利用し銀行システムの決済機能を利用するアプリを提供する事業者を「電子決済等代行業」と称し、銀行との契約締結と財務局への事業者登録が義務付けられた。
- この金融機関におけるAPIのことを、金融機関とフィンテック企業とのオープン・イノベーションを進める鍵となる技術ということで「オープンAPI」とよばれている。



※「金融分野におけるオープンAPIに関する取り組み」より抜粋

# オープンAPIの実現に必要なこと

オープンAPIの実現には、「ガイドラインに従ったAPIの実装」「高度なセキュリティの実現」および「法令の遵守」といった対応が必要です。

## ガイドラインに従ったAPIの実装

- APIを実装するためには、まず**提供者と利用者の間のインターフェースを定義し、各銀行の勘定系システムにあわせて実装**していきます。このインターフェースの電文仕様についてガイドラインを記述した「銀行分野のオープンAPIに係る電文仕様標準について」を一般社団法人全国銀行協会が公開しています。  
<https://www.zenginkyo.or.jp/news/2018/n10918/>

- また当ガイドラインでは、「電文仕様標準」のほか、関係者がAPIを開発するに当たって留意すべき「開発原則」、推奨されるAPIの基本的な仕様を定める「開発標準」の三点が定められています。
- プログラムが処理できる形でAPIの仕様をより詳細に記述するためは、OpenAPI（旧Swagger）という仕様記述のための標準フォーマットを利用するとよく、これによって、提供者と利用者の間で取り交わすデータ構造や使われる通信プロトコルやHTTPメソッド、接続先やセキュリティ標準を細かく規定し、APIの提供者と利用者との間の齟齬を取り除くことができます。

## 高度なセキュリティの実現

### 本資料の対象

- 銀行の預金口座の読み書きを扱うAPIの運用のためには、それらを狙ったハイレベルな攻撃に耐えうる高度なセキュリティの実現が必要です。
- 複数の内部APIを集約して一元管理するAPI Gatewayでは、そのセキュリティ担保のために統合認証のしくみが必要であるが、そのための標準仕様としてOpenID Connectがある。さらにこれを金融機関向けに拡張した標準仕様にFAPIがあり、このレベルの正しい実装が理想的です。

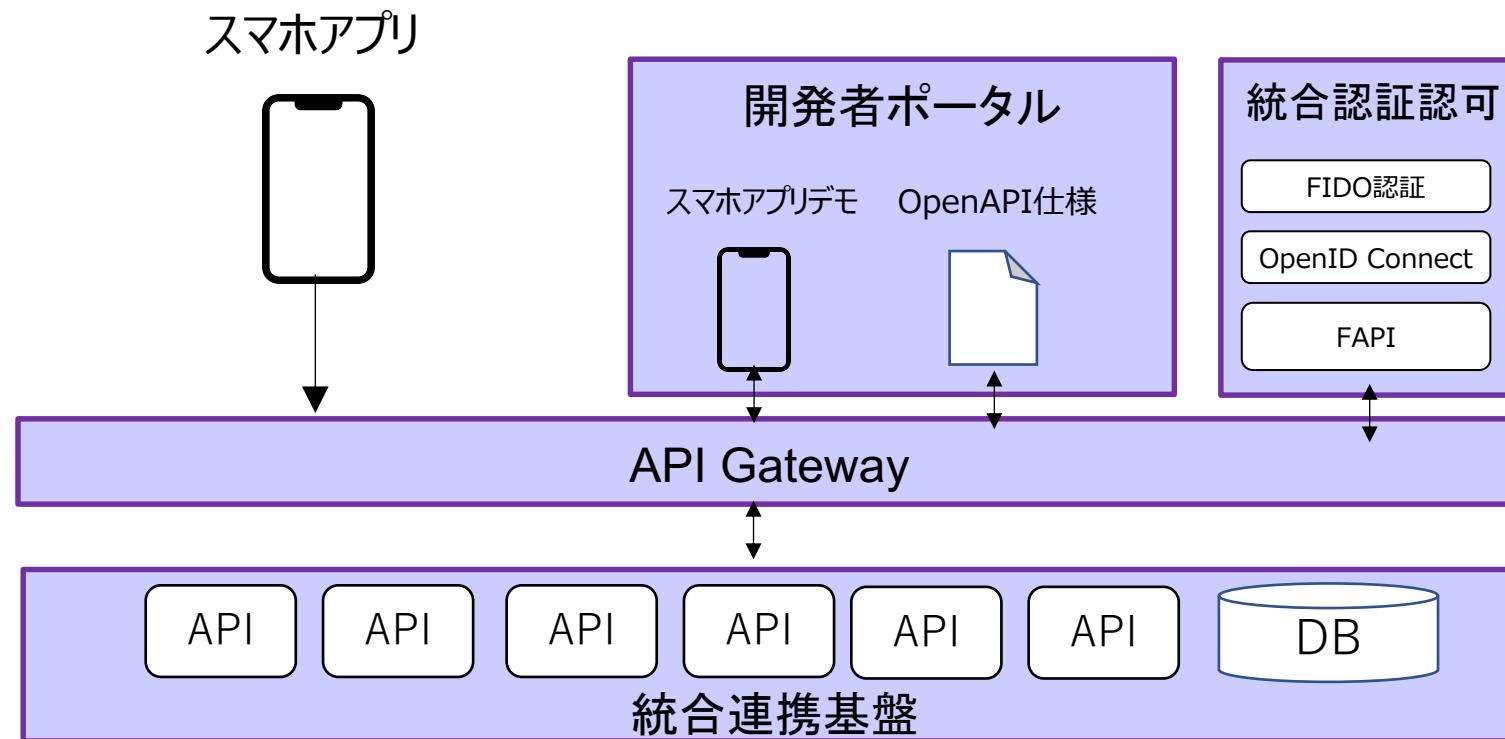
## 法令の遵守

- オープンAPIは、その提供者は「銀行」、利用者は改正銀行法に則って「電子決済等代行業」と呼ばれ、銀行との契約締結と財務局への事業者登録が義務付けられています。
- ここでの契約書は、一般社団法人全国銀行協会が公開している「銀行法に基づくAPI利用契約の条文例」を参考に作成されます。

<https://www.zenginkyo.or.jp/news/2018/n10918/>

# APIプラットフォーム

- API Gatewayと連携する統合認証とユーザフェデレーションを実現する統合認証認可サーバを提供し、生体認証によるパスワードレス認証やOpenID Connect、金融業界レベルのセキュリティを担保するFinancial Grade APIを実現します。
- 統合連携基盤が提供するAPIを、統合認証認可サーバと連携したAPI Gatewayを通して、第三者のアプリケーションに安全に提供します。Flutterをベースとして、FIDOに対応したパスワードレス認証が可能なスマホアプリを簡単に作成することが可能です。ユースケースを考えて統合連携基盤のAPIを利用するスマホアプリを作ってもよいかもしれません。



# 参考) 金融機関におけるKongの採用



2020年7月29日プレスリリース

Xinja Banks on Kong Enterprise to Power Digital Banking Platform  
(オーストラリアのモバイル銀行であるシンシャ銀行がKong Enterpriseを採用)

“Our lending origination partner needed us to expose our APIs in a way that was secure and compliant with OIDC standards,” said Rohan Sharp, co-founder and head of architecture at Xinja Bank. “This was easy with Kong; not only could we comply with industry standards, but we could protect our APIs and internal landscape quickly and with low risk – while minimizing impacts on Xinja’s existing infrastructure.”

シンシャ銀行の共同創設者でチーフアーキテクトのローハン・シャープ氏

「我々の銀行の融資パートナー企業は、OpenID Connect(OIDC)標準にのっとり、安全な方法で銀行APIを提供することを求めていました。Kongは業界標準に従うだけでなく、既存インフラへの影響を最小限にして、APIや内部システムを迅速に低リスクで攻撃から守ることができます。」

<https://konghq.com/press-release/xinja-banks-kong-enterprise-power-digital-banking-platform/>

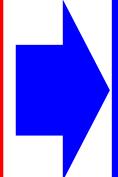
# FIDOが求められる背景

ネットバンクなどのスマホアプリやWebアプリのユーザ認証では、セキュリティ上の課題や使い勝手の課題が存在し、それらを公開鍵暗号方式や高度なハードウェアを使って解決する方法をFIDO2は業界標準として規定しています。

## ユーザログインにおける課題

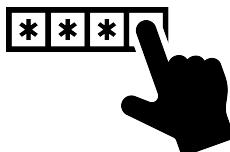
### ①セキュリティ上の課題

パスワードログインをベースとしており、マルウェア感染やフィッシング攻撃などによるパスワード漏洩が絶えない。ワンタイムパスワードによる二段階認証もフィッシング攻撃をうける。



### ②使い勝手の課題

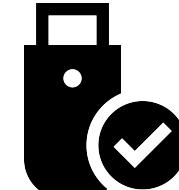
ログインのたびにパスワード入力を求められたり、送金等の重要処理のたび二段階認証が必要で、利用者をイライラさせたり、利用者がパスワードを忘れてしまうことがある。



## FIDO2における解決方法

### 認証機によるキーペア生成と保管

USBドングル型やスマホ内蔵型で提供される認証機は、公開鍵暗号方式で生成されたキーペアを使い認証を行う。認証機は、署名は可能だが、秘密鍵はいかなる方法でも取り出せない特徴があり、これにより認証情報の漏洩を防ぐ。



### 生体認証ログイン

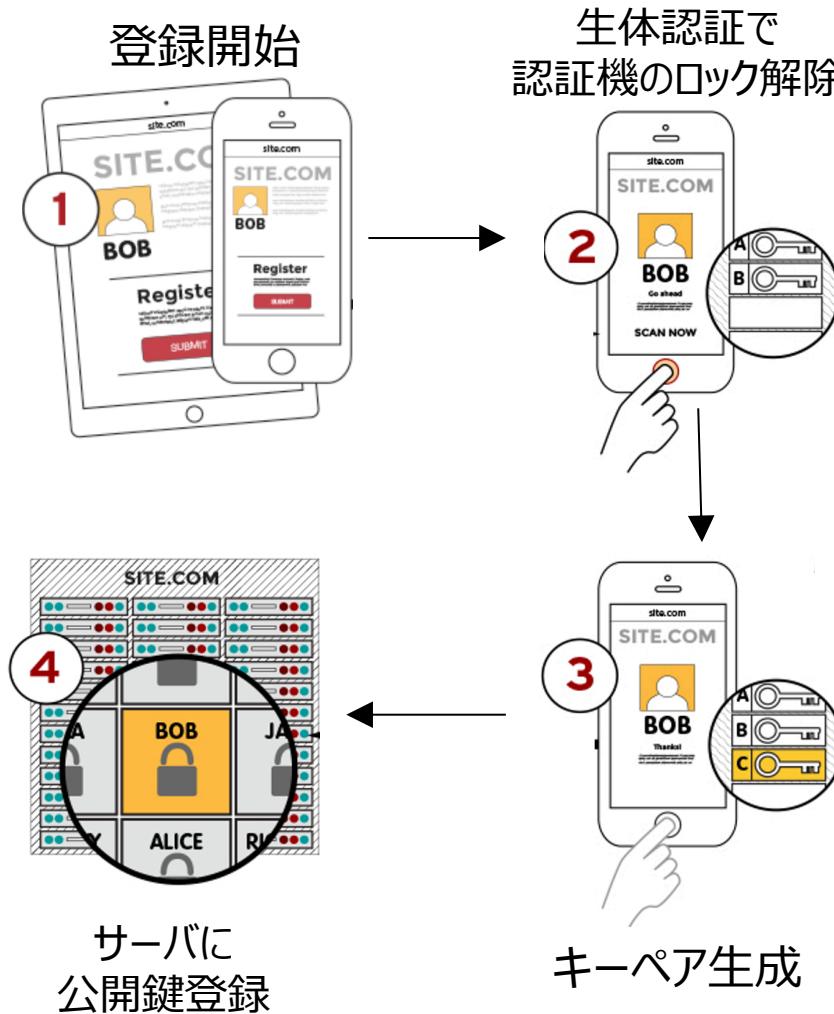


指紋や顔認証といった生体情報を使い認証機のキーペア生成や署名を行うことにより、キーボードを使ったパスワード入力より簡単にすばやくログインを行うことができ、ユーザのストレスを低減する。また、パスワードを覚える必要もない。

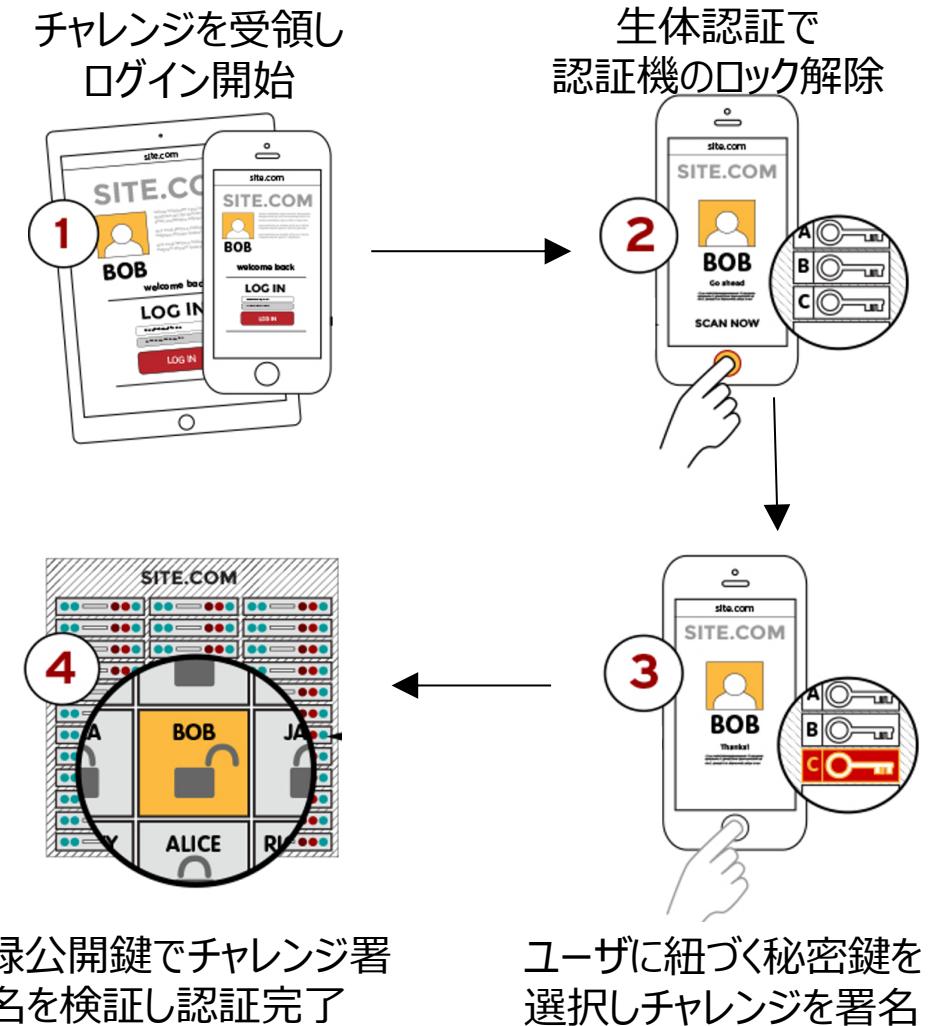
# FIDOのしくみ

一般的にWebサービスの認証プロセスは「ユーザ登録」と「ユーザログイン」の2フェーズに分かれFIDO認証もこれと同じく「FIDO登録」と「FIDOログイン」からなります。

## FIDO登録



## FIDOログイン



# FIDOの普及について

- ・「FIDOが求められる背景」によるとメリットばかりで、FIDOを使わない原因が見当たりませんが、**2020年現在FIDOが普及していない理由**には以下のものが挙げられます。
  - ① ユーザがログインを行う端末であるパソコンやスマホの古いものには**FIDO認証機が内蔵されていない**ため、FIDOを利用するには**別途FIDO認証機（USB型とBLE型などがある）**を数千円で購入し、端末とつないで利用しなければならなかった。
  - ② パソコンやスマホ提供事業者の代表格であるMS社（Surface PC）、Google社（Chromebook PC、Android端末）、Apple社（Mac、iPhone）のうち、**Apple社だけがFIDOアライアンスに参画していなかった**。
- ・しかし、**2020年2月にApple社はFIDOアライアンスに加盟し、同年9月のiPhone12の出荷と同時にリリースが予定されているiOS14からSafariブラウザでのFIDO2対応が決定**している。（※2020/7/10にApple社はiOS14のパブリックベータを配信開始したため希望者は無料で利用が開始できる）
- ・そのため**生体認証を搭載するFIDO対応端末がひとつおりそろうため、2020年後半からFIDOが急速に普及する**だろうと予想できます。
  - ① MS社：生体認証機能をもつWindows10を搭載したPCやタブレットはFIDO対応済
  - ② Google社：生体認証機能をもつAndroidスマホやChromebookはFIDO対応済
  - ③ Apple社：iOS14/macOS 11.0 Big Surが2020年9月に無事リリースされると過去の生体認証搭載スマホやmac PCを含めたすべてがFIDO対応になる。

# OpenID Connectとは？

「OpenID Connect」は、アクセス権の連携仕様である「OAuth2.0」とID情報の連携仕様である「OpenID2.0」を組み合わせた統合認証認可を実現する業界仕様です。

アクセス権連携

ID連携

あるサービスへのアクセス権を  
別のサービスに安全に与える

ID情報（認証結果と属性情報）  
を安全にサービス間でやりとり

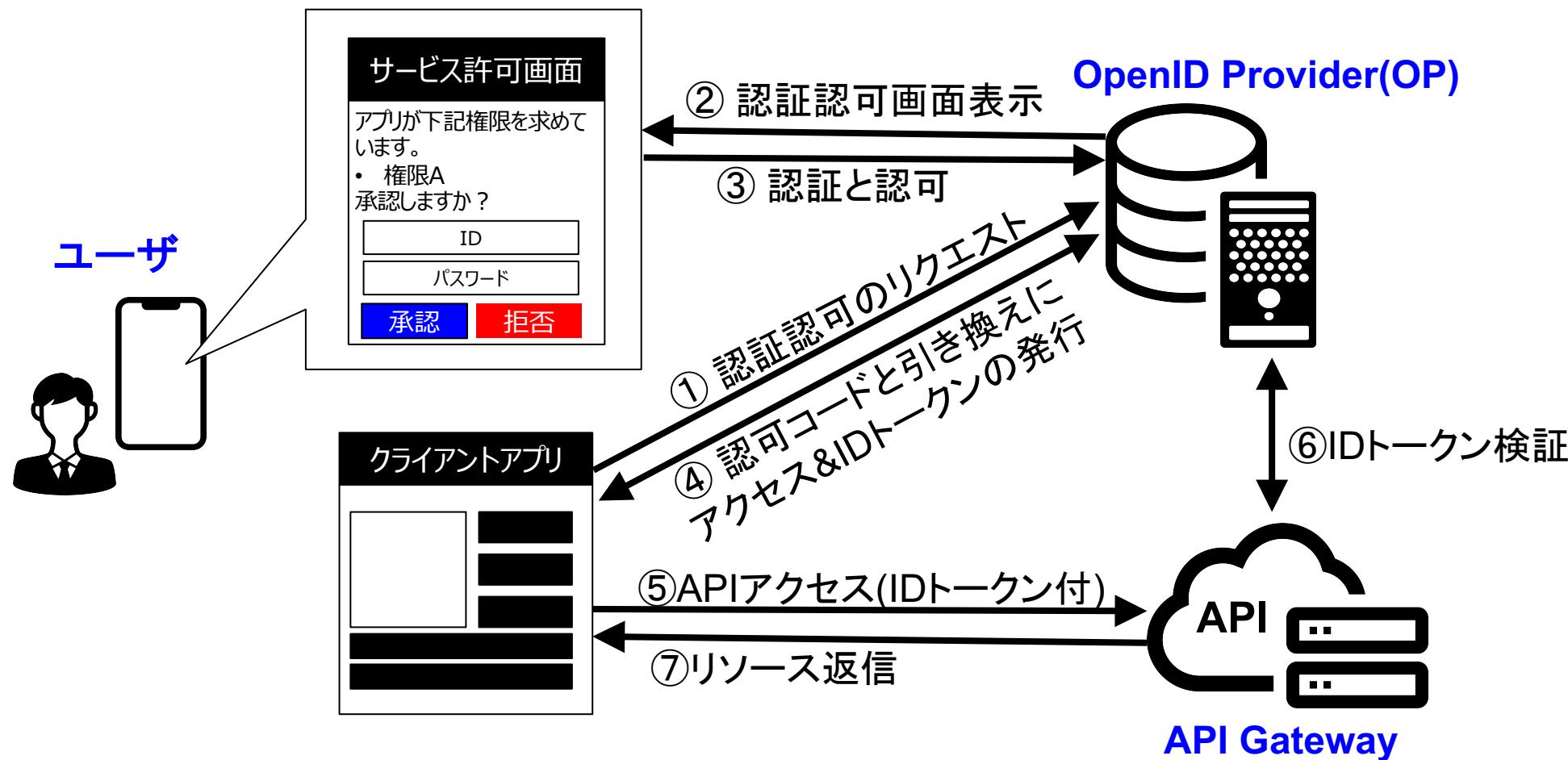
**OAuth2.0**

**OpenID2.0**

**OpenID Connect**

# OpenID Connectのしくみ

OpenID Connectは、様々なタイプの認証(Basic認証やFIDO認証など)と連携し、第三者が提供するクライアントアプリ（スマホアプリやWebアプリ）にユーザー情報を提供することなく、安全に認証認可のしくみを提供します。下図は認可コードフローというOpenID Connectが対応するフローのうちの一例です。



# OIDCのセキュリティ攻撃への対応

トークンを利用してセッションレスなWebAPIと相性のよいシングルサインオンを実現できますが、ステートレスで独立独歩なトークンならではの様々なセキュリティ攻撃の対象となるため、OIDCの正しい利用が求められます。

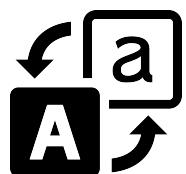
## 中間者攻撃



HTTPもしくはHTTPSの誤った利用をつくことで  
**中間者がトークンを盗難する攻撃**。

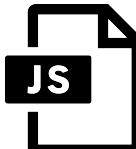
例えば、ネットワークトラフィックを監視しているプロキシ上ではSSLが解かれ認証トークン情報をみれたりするためそこから漏洩したりさまざまな方法が考えられる。これを防止するには、アプリケーション全体で**HTTPSとセキュアクッキー**を使用することが求められる。

## トークン置き換え攻撃



クライアントアプリ開発者が  
悪意を持って、**バックエンドサーバに送るアクセストークンを別のものに差し替えること**で別ユーザーでログインさせるといった攻撃。  
対策としては、バックエンドサーバにアクセストークンを送るインプリシットフローをやめて認可コードフローを使う。もしくはアクセストークンを使わず、IDトークンを使うことで対応する。

## クロスサイトスクリプティング



攻撃者が攻撃先のブラウザで実行されているアプリに**悪意を持ってJavaScriptコードを挿入する可能性**があります。挿入されたコードは、**認証トークンを読み取って攻撃者に送信します**。これは、**HTTP Only**または**セキュアクッキー**を使用して認証トークンを保存することで簡単に防ぐことができます。また、JavaScriptでアクセスできる場所であるローカルストレージを使用して認証トークンを保存しないようにします。

## リクエスト強要(CSRF)



不特定多数を対象にログインしたユーザーが自分の意志とは関係なく攻撃者が意図した行為をさせる攻撃。例えば、**物品を購入に対する支払いスクリプトを他人に実行させ代わりに決済させるような攻撃**です。対策としては認可リクエストにstateパラメータのようなクライアントのセッションに紐づく値をセットすることで**単にURIリンクをクリックしただけでアクセスができない**ようにします。

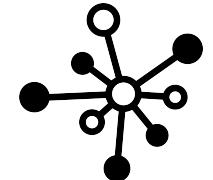
## 認可コード横取り攻撃



## 認可コード横取り攻撃

認可コードフローをサポートするスマホアプリを模したアプリを攻撃者が開発してターゲットユーザーの端末にインストールさせて、**認証コードからアクセストークンを取得する攻撃**。  
対策としては、**認可リクエストを作成したクライアントアプリのみが、認可レスポンスに含まれる認可コードからアクセストークンを取得できる**ようにする。

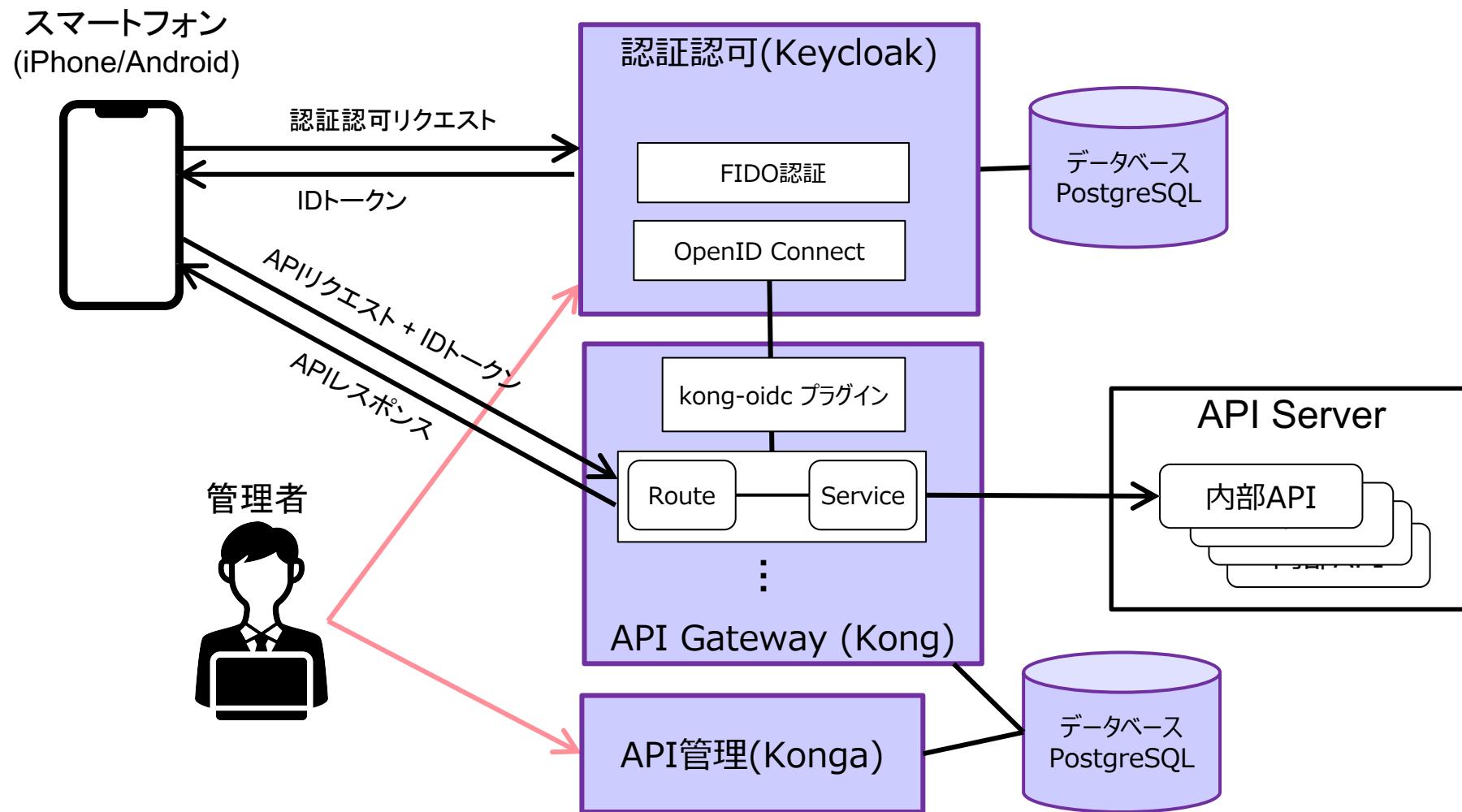
## ミックスアップ攻撃



クライアントアプリが複数の認可サーバに接続している条件で、その中の**悪意のある認可サーバが他の認可サーバの発行したアクセストークンなどを奪おうとする攻撃**。  
対策としては、認可サーバごとにリダイレクトエンドポイントを分けることや認可レスポンスに認可サーバの情報を含め、クライアントアプリはその情報を用いて処理の分岐を行うことで対応する。

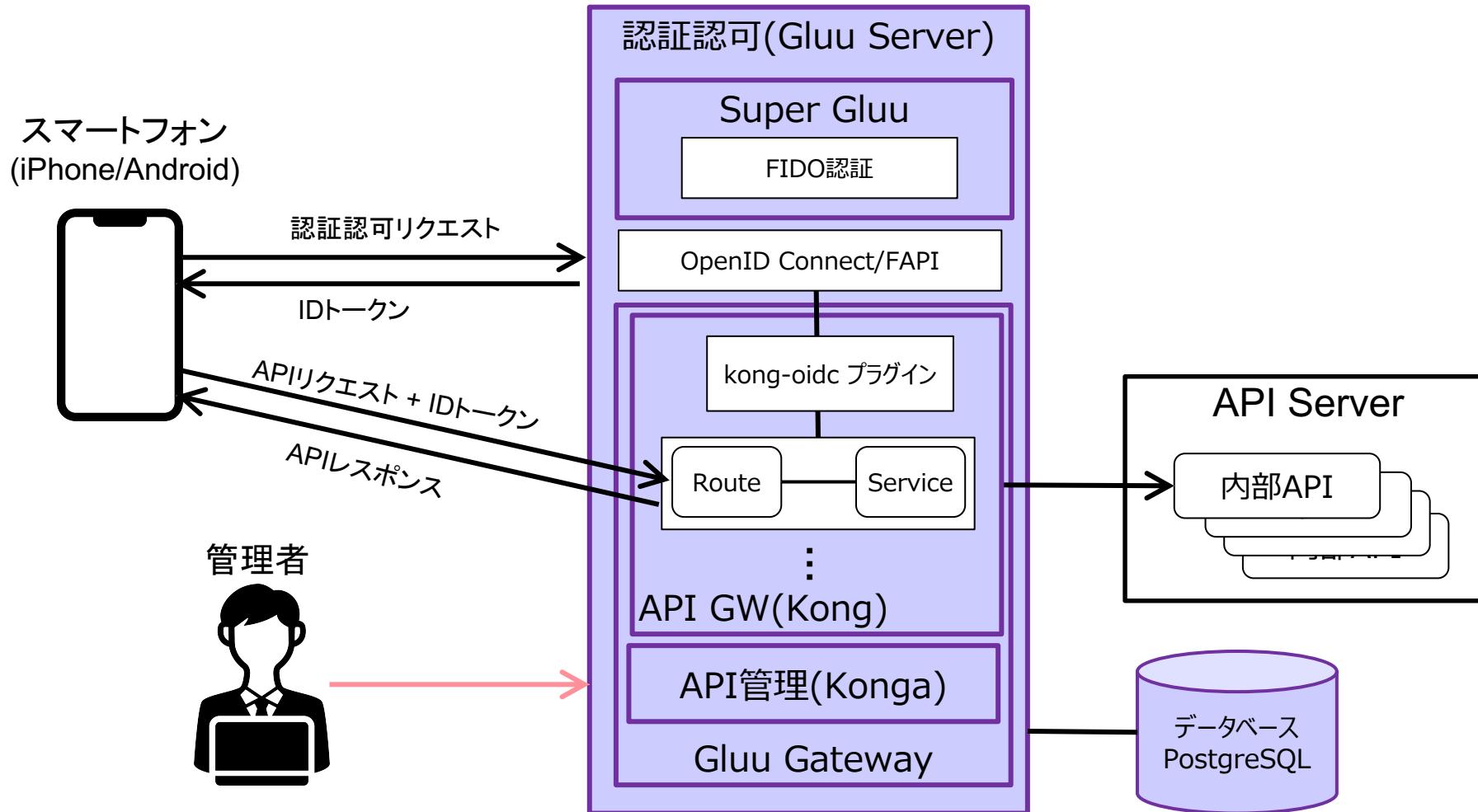
# 認証認可サーバのシステム構成 ~Keycloak~

Keycloakを認証認可サーバとして利用し、Kongと連携してOpenID ConnectとFIDO認証が利用できる。Keycloakは、RedHat JBossプロジェクトが開発を進めており、全機能をソースコードも含め、Apache2.0ライセンスで提供されており、国内外で実績も十分あるが、OIDCの一部とFAPIには対応していない。



# 認証認可サーバのシステム構成 ~Gluu Server~

Gluuを認証認可サーバとして利用すると、OpenID ConnectとFAPI、FIDO認証が利用できる。Gluu Serverは、Gluu社によって開発されており、全機能をOSSライセンスで提供されている。また、Gluu社によって有償サービスが提供されている。



# OpenID ConnectとFAPIの対応状況

OpenID FoundationによるOpenID ProviderおよびFAPI認証の対応状況を下表に示す。

認証されたプロファイル	Keycloak 2.3.0	Gluu Server 4.2	Authlete 2.1
<b>OpenID Provider</b>	一部対応	対応	対応
Basic OP	対応	対応	対応
Implicit OP	対応	対応	対応
Hybrid OP	対応	対応	対応
Config OP	対応	対応	対応
Dynamic OP	対応	対応	対応
Form Post OP	非対応	対応	対応
3 <sup>rd</sup> Party-Init OP	非対応	対応	非対応
<b>FAPI</b>	非対応	対応	対応
R/W OP w/ MTLS	非対応	対応	対応
R/W OP w/ Private Key	非対応	対応	対応
FAPI-CIBA OP poll	非対応	対応	対応
FAPI-CIBA OP ping	非対応	対応	対応

# 参考資料①

## オープンAPI関連

- 総務省金融分野におけるAPI公開 : <https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h30/html/nd133140.html>
- 金融分野におけるオープンAPIに関する取り組み :  
[http://www.kantei.go.jp/jp/singi/it2/senmon\\_bunka/data\\_ryutsuseibi/detakatsuyo\\_wg\\_dai8/siryou2.pdf](http://www.kantei.go.jp/jp/singi/it2/senmon_bunka/data_ryutsuseibi/detakatsuyo_wg_dai8/siryou2.pdf)
- 全銀協公開資料 : <https://www.zenginkyo.or.jp/news/2018/n10918/>

## FIDO関連

- FIDO基礎: <https://fidoalliance.org/fido%e3%81%ae%e7%89%b9%e9%95%b7/?lang=ja>
- FIDOのハンズオン: <https://david3080.github.io/fido2/>
- iPhone14でのFIDO+TouchID対応: <https://developer.apple.com/videos/play/wwdc2020/10670>
- iOSのバージョンの普及率 : <http://smatabinfo.jp/os/ios/index.html>

## OIDC関連①

- OpenID Japanのスペック資料の日本語訳: <https://www.openid.or.jp/document/>
- OAuth2.0日本語訳: <http://openid-foundation-japan.github.io/rfc6749.ja.html>
- OIDC1.0日本語訳: [http://openid-foundation-japan.github.io/openid-connect-core-1\\_0.ja.html](http://openid-foundation-japan.github.io/openid-connect-core-1_0.ja.html)
- NRIのOIDC概要: <https://www.mhlw.go.jp/content/10800000/000537444.pdf>
- AuthleteのOIDC説明動画(OIDC入門編#1): [https://www.youtube.com/watch?v=PKPj\\_MmLq5E](https://www.youtube.com/watch?v=PKPj_MmLq5E)
- AuthleteのOIDC説明動画(OIDC入門編#2): <https://www.youtube.com/watch?v=wAdVvJX9-Vw>
- AuthleteのOpenID Connect 全フロー解説: <https://qiita.com/TakahikoKawasaki/items/4ee9b55db9f7ef352b47>
- OIDC認証: <https://openid.net/certification/>
- オージス総研によるOIDC認証資料: [https://www.ogis-ri.co.jp/news/themistuct/docs/20161216-OpenID\\_TechNight\\_vol14\\_OGIS\\_2.pdf](https://www.ogis-ri.co.jp/news/themistuct/docs/20161216-OpenID_TechNight_vol14_OGIS_2.pdf)

# 参考資料②

## OIDC関連②

- OIDCざっくり説明: <https://josys-navi.hiblead.co.jp/josys-bk openid-connect>
- Yahoo! ID連携ドキュメント: <https://developer.yahoo.co.jp/yconnect/v2/>
- OAuth2.0との違い: <https://www.buildinsider.net/enterprise/openid/connect>
- IdentityServerをベースとしたOAuth2.0とOIDCの違い説明:  
<https://gist.github.com/jawadatgithub/638c11f08ecc0d76b05c>

## セキュリティ脆弱性関連

- OAuth 2.0に潜む「5つの脆弱性」: <https://www.atmarkit.co.jp/ait/articles/1710/24/news011.html>
- OAuth 2.0 Threat Model and Security Considerations(日本語訳):  
<http://openid-foundation-japan.github.io/rfc6819.ja.html>
- リクエスト強要(CSRF): <https://yamory.io/blog/about-csrf/>
- トーカン置き換え攻撃: [https://www.slideshare.net/kura\\_lab/devsumi201413c5](https://www.slideshare.net/kura_lab/devsumi201413c5)

## Keycloak関連

- Keycloak日本語ドキュメント: [https://keycloak-documentation.openstandia.jp/master/ja\\_JP/](https://keycloak-documentation.openstandia.jp/master/ja_JP/)
- Keycloakサーバ管理日本語ドキュメント:  
[https://keycloak-documentation.openstandia.jp/master/ja\\_JP/server\\_admin/index.html](https://keycloak-documentation.openstandia.jp/master/ja_JP/server_admin/index.html)

## Gluu関連

- Gluu: <https://www.gluu.org/>
- Gluuドキュメント: <https://gluu.org/docs/>