



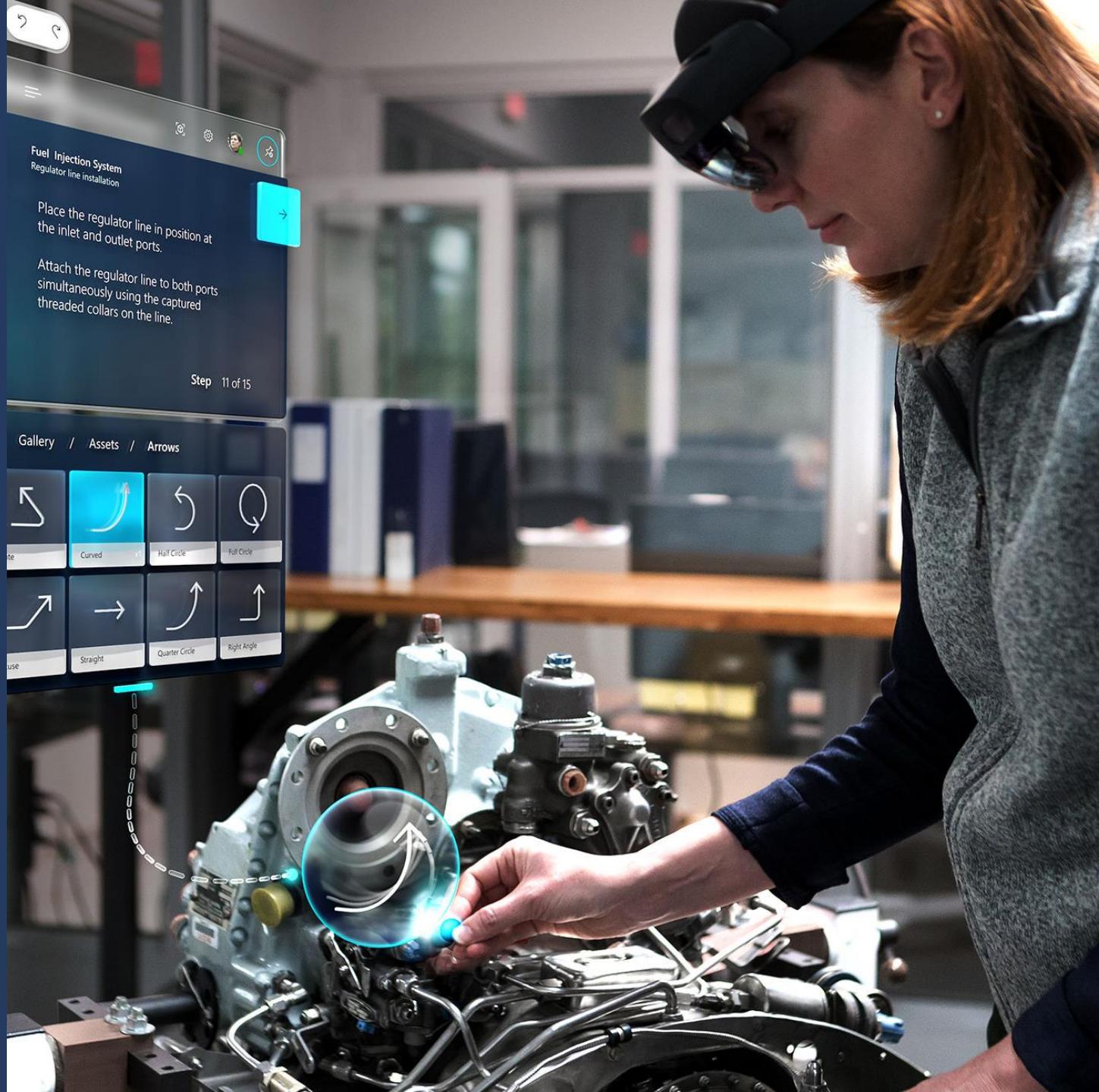
# Azure アプリケーション認証認可 1Day ワークショップ

## 体験型ワークショップ資料 #1

Microsoft  
エンタープライズアーキテクト統括本部  
AppDevアーキテクト技術本部  
クラウド ソリューションアーキテクト

Shiho Asa

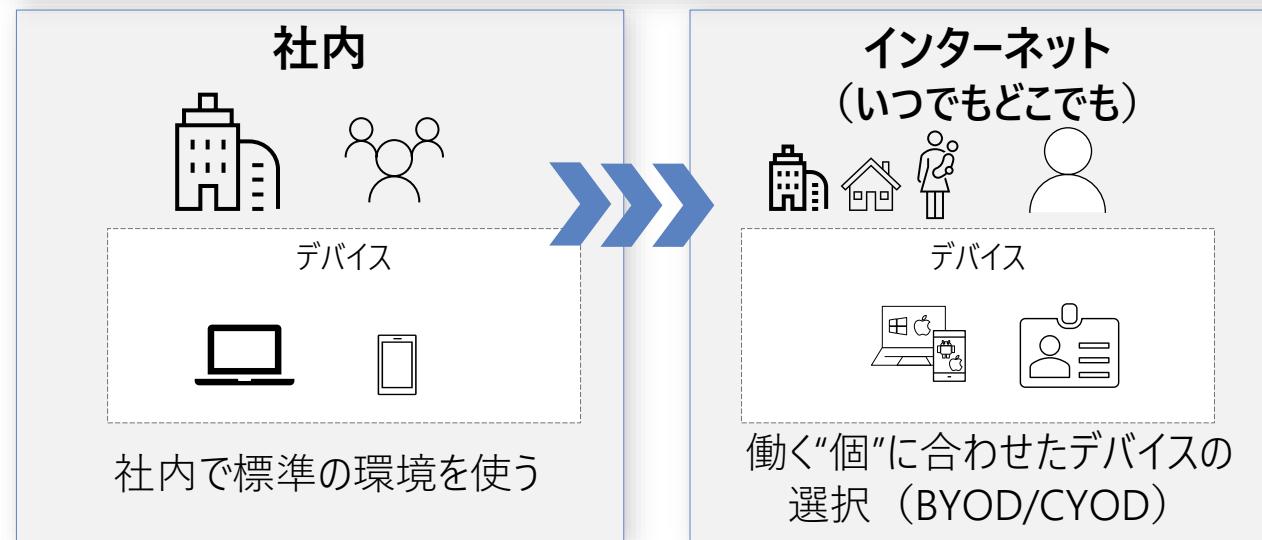
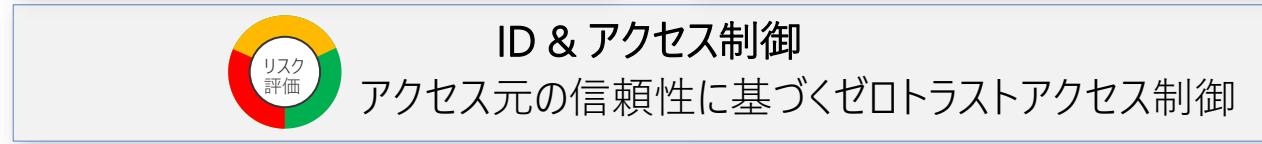
Oct 20<sup>th</sup>, 2022





フロントエンドWebアプリにコーディングレスで  
認証機能を追加しよう！

# ビジネスの変化に合わせたアーキテクチャの変化



ビジネスの拡大スピードに合わせたセキュアで柔軟性に富んだクラウド

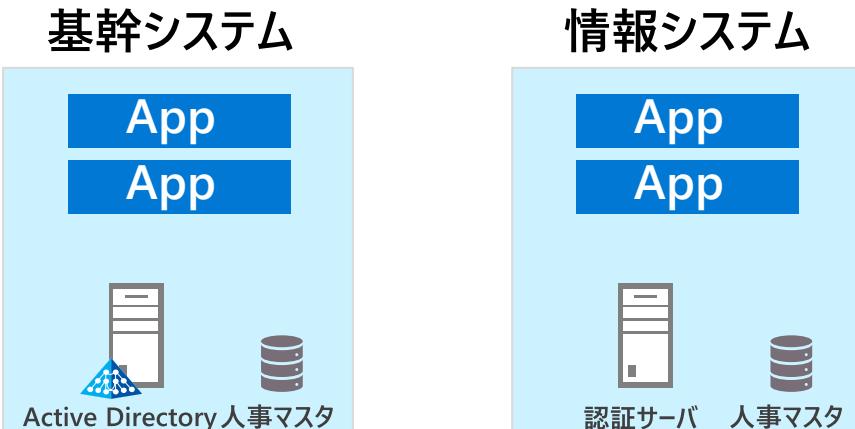
- ・デジタルサイロを無くした一貫したセキュリティ統制の実現
- ・セキュリティレベルに合わせた認証 / 利用実態の把握

働く“個”に対応した働き方の選択

- ・ジョブ(会社/部署/職務)/働き方(ダイバシティ)に合わせたデバイスの選択
- ・リモート業務に合わせたエンドポイントセキュリティ

# 企業システムにおける認証基盤の重要性

今まで

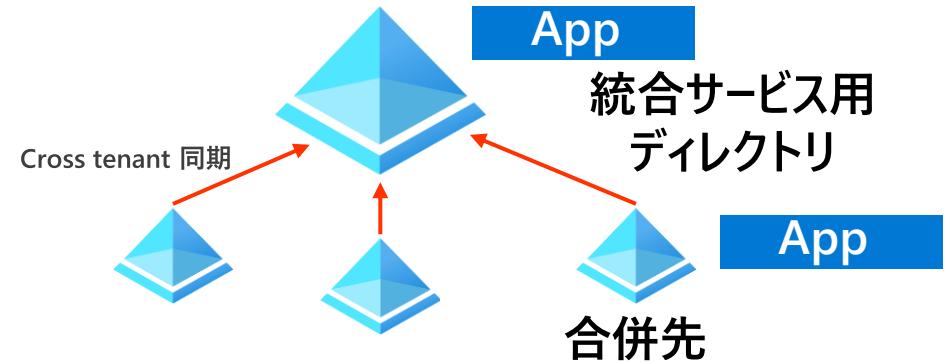


- ・ユーザー統合
- ・システム利用の為の基幹システム改変



結果、長期的なシステム統合プロジェクト  
が発足しビジネス統合の足かせに

これから



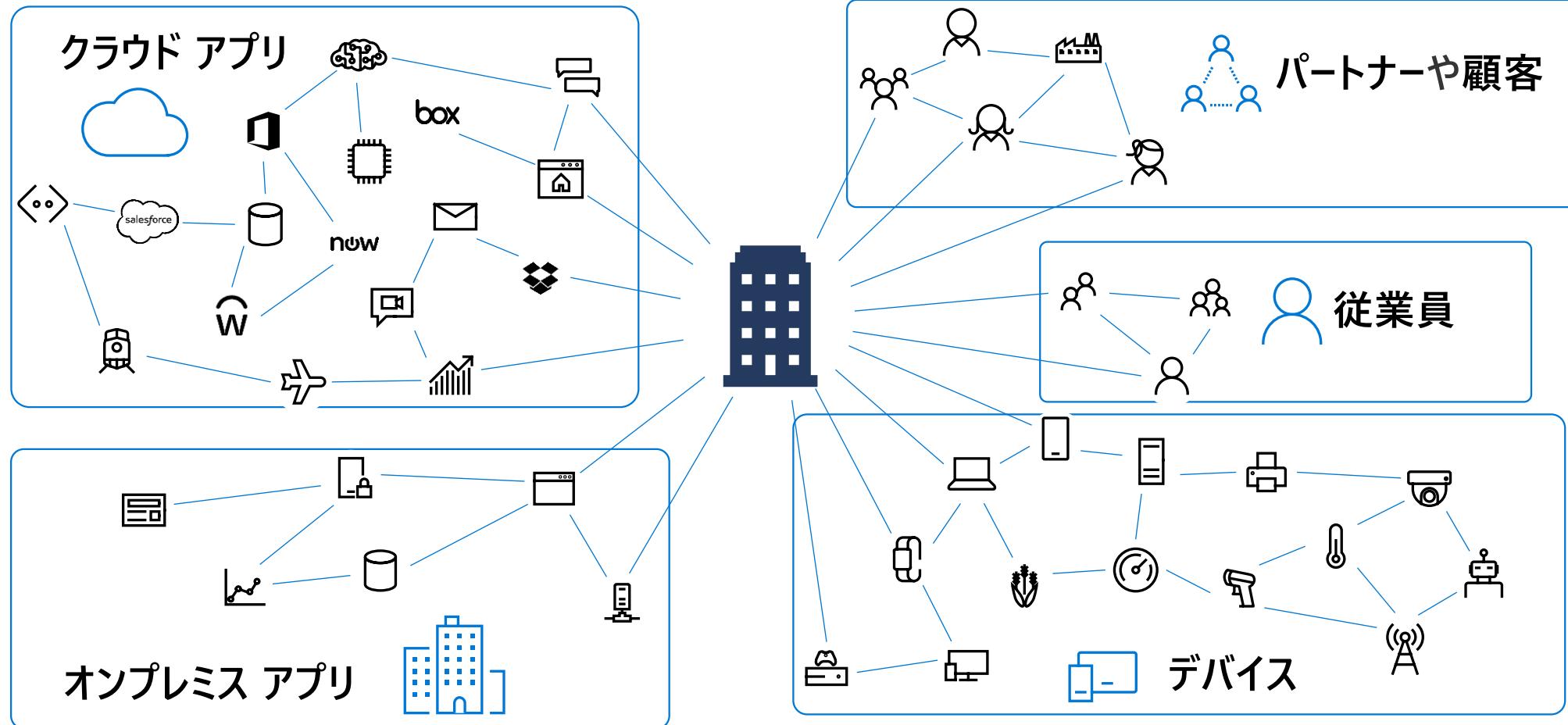
- ・認証元の切り替えや Cross Tenant 同期などを活用し、迅速に統合



世の中の標準(クラウド認証技術)を使う  
事ですぐに統合、ビジネスの開始

# IT 部門のセキュリティ境界と管理対象

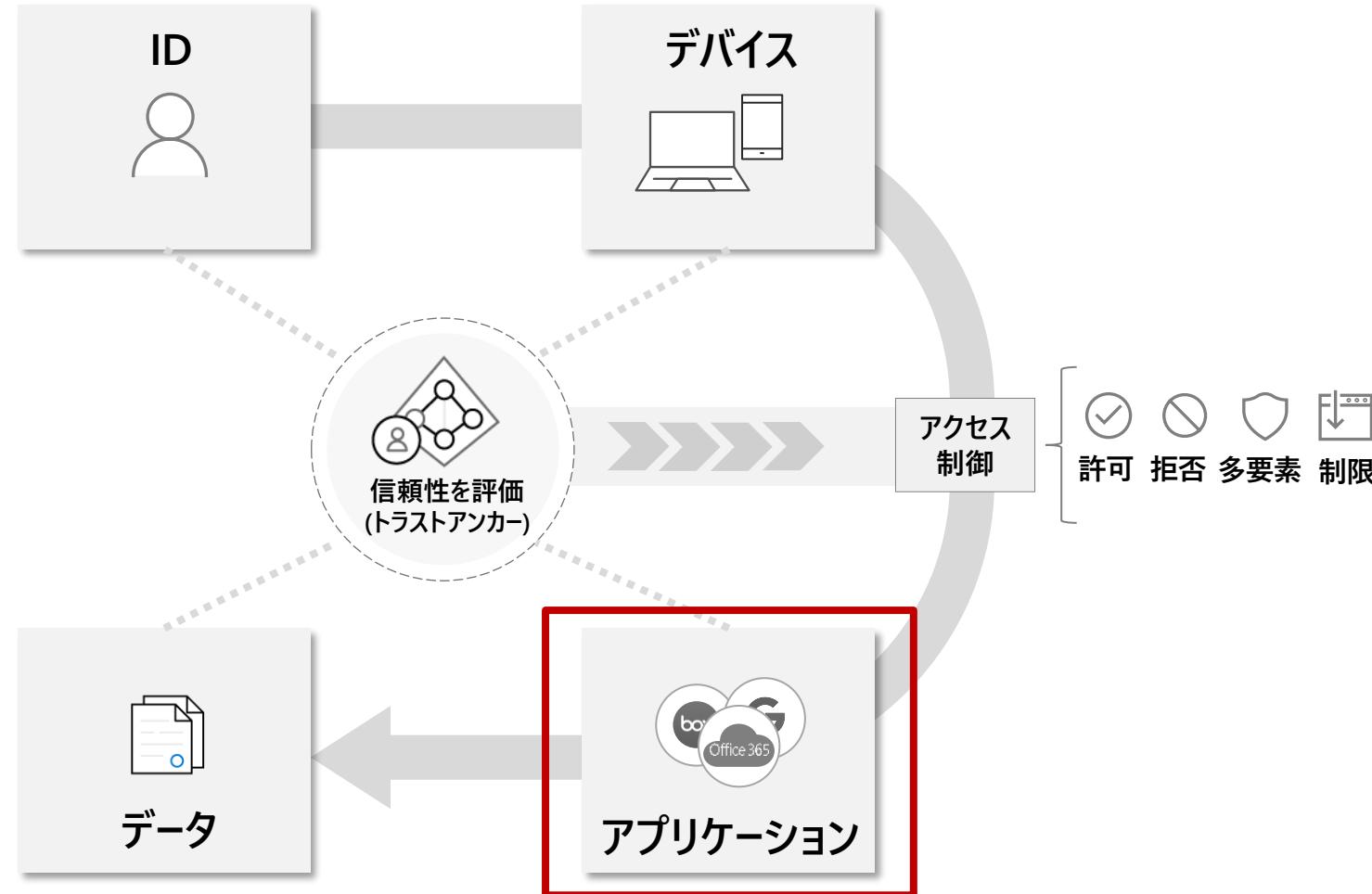
ひと昔前はセキュリティの境界線がしっかり区切りがあったが  
現在は利活用するものや、ユーザーの種類が増え、管理対象の境界線が曖昧になっている



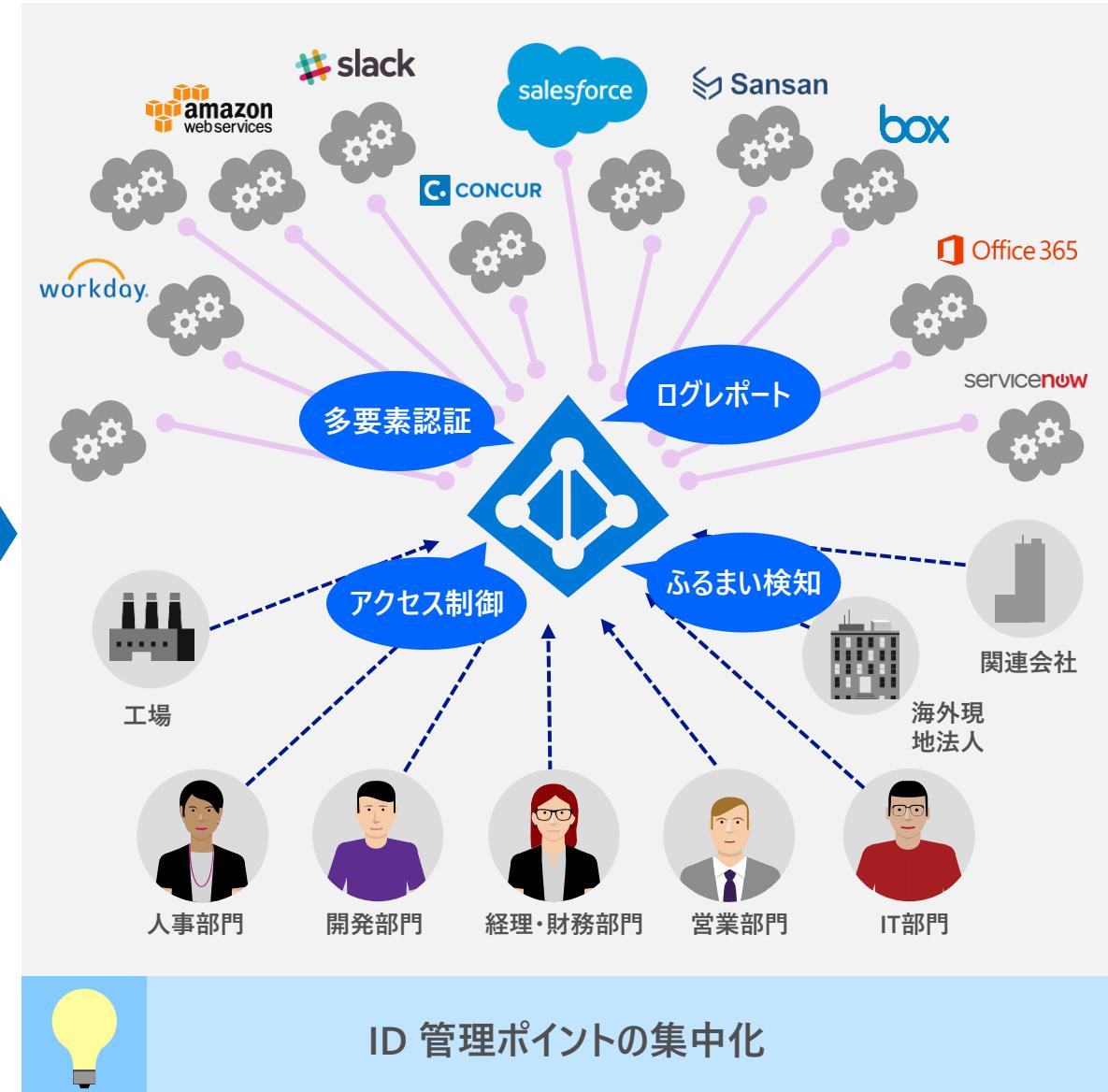
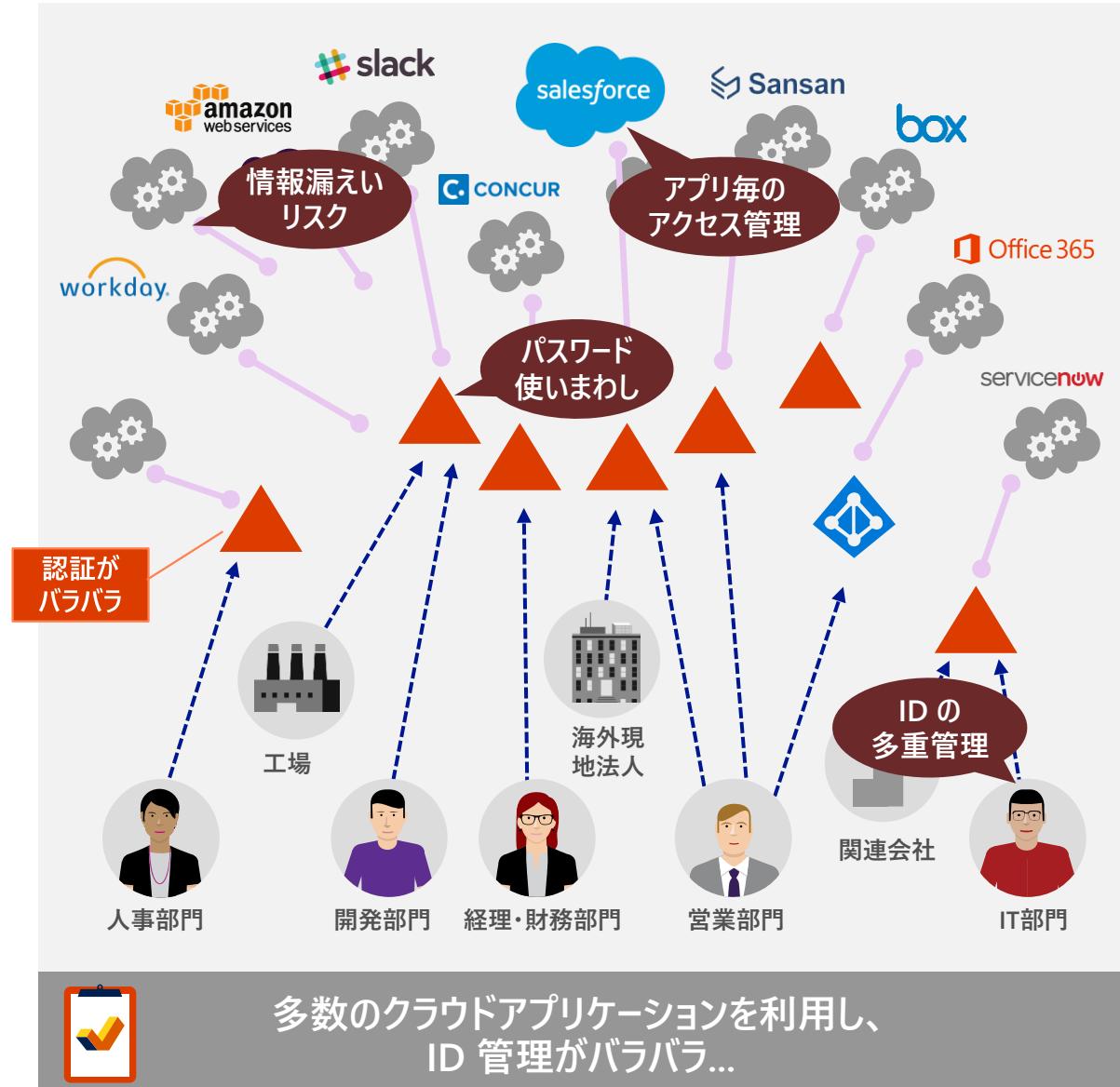
# NIST：ゼロトラスト アーキテクチャの 7 原則

1. すべてのデータソースとサービスをリソースとみなし、識別する
2. 暗黙的な信頼は排除する。
3. 要求元の信頼は、アクセスが許可される前に評価する
4. リソースへのアクセスは、ユーザー ID / デバイスの評価と、リソース / データの重要度によって“**動的ポリシー**”により決定する
5. 組織が所有するシステムを監視して、安全な状態のままであることを確認する
6. ユーザー認証は動的であり、アクセスが許可される前に厳密に実施する
7. 資産の現状を把握しセキュリティの監視を常時のものとする事

参考情報：NIST Zero Trust Architecture : “2. Zero Trust Architecture



# シングルサインオンによる ID 集中管理によるセキュリティ



# Part1: はじめての認証認可 ~カレンダーアプリを作ってみよう~

# 認証と認可

**認証 :**

システムを利用する際の本人確認  
(あなたが本当にあなたであることを証明すること)

**認可 :**

認めて許可すること

# Office 365の例～認証と認可～



Aさん

私は A です。  
自分のメールボックスの予定表を見たいです



あなたが A さんであることを確認しました（認証）

予定表を参照する権限があるため、見ることを許可します  
(認可)

# Azure Active Directory とは

- Azure Active Directoryは、ユーザやアプリケーションの認証と認可をするために使われるサービス
- Microsoft 365（旧称: Office 365）を使っている場合は、Azure Active Directoryを使って、OutlookやPowerPoint情報へのアクセスの認証認可をしている
- Azureの各サービス（例: App Service）のリソース管理（例: 作成、削除）をするときの認証認可としても利用

<https://docs.microsoft.com/azure/active-directory/fundamentals/active-directory-whatis>

# Microsoft Graph API とは

- Microsoft 365の情報（例: Outlook）やAzure ADの情報をAPI経由で取得・更新
- アプリケーションから呼び出すためには
  - アプリケーション開発者もAzure Active Directoryの理解が必要
  - 対象Microsoft 365のグローバル管理者の承認が必要

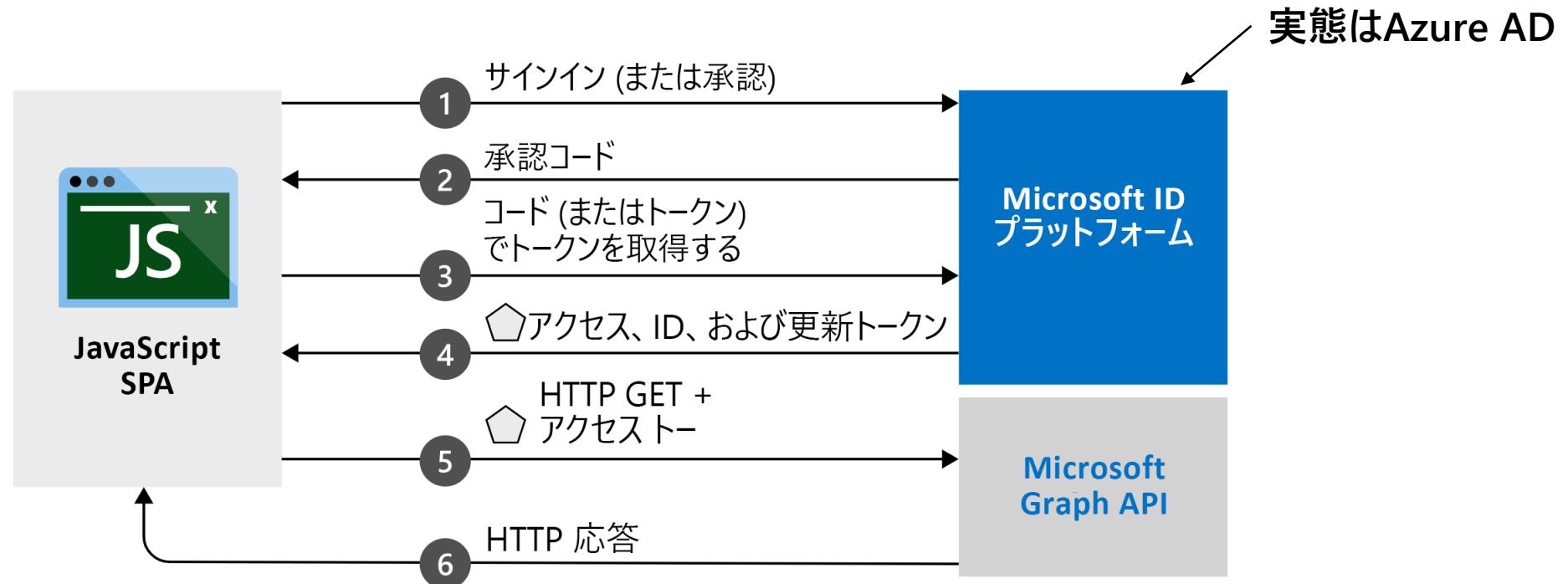


<https://docs.microsoft.com/graph/overview>

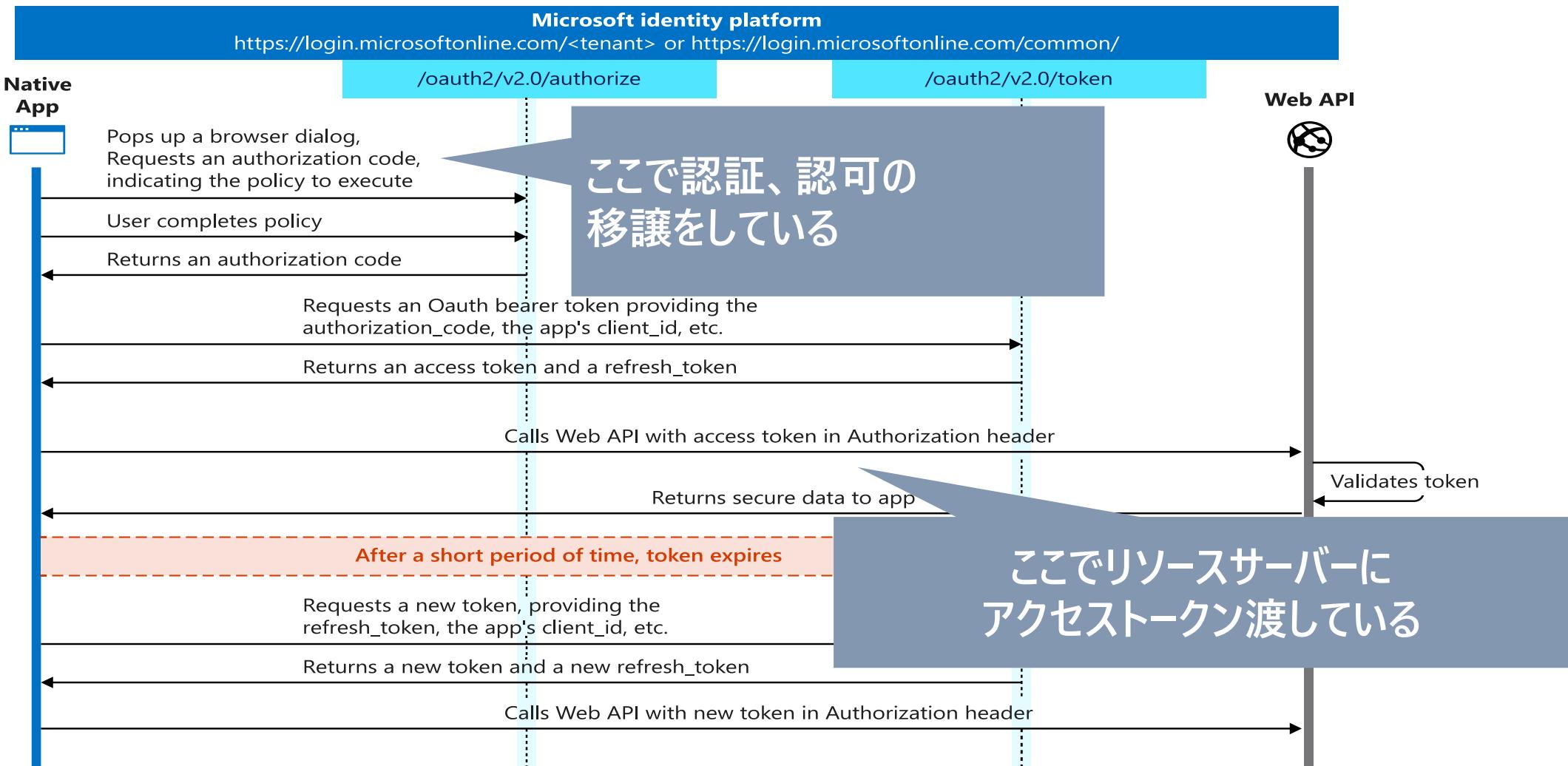
<https://docs.microsoft.com/ja-jp/learn/modules/msgraph-intro-overview/>

# Microsoft Graph API の呼び出し

ユーザが自身の認証認可情報を用いて、Graph APIを呼び出すことだけでなく、  
アプリケーションが自身の認証認可情報を用いて、Graph APIを呼び出すことが可能

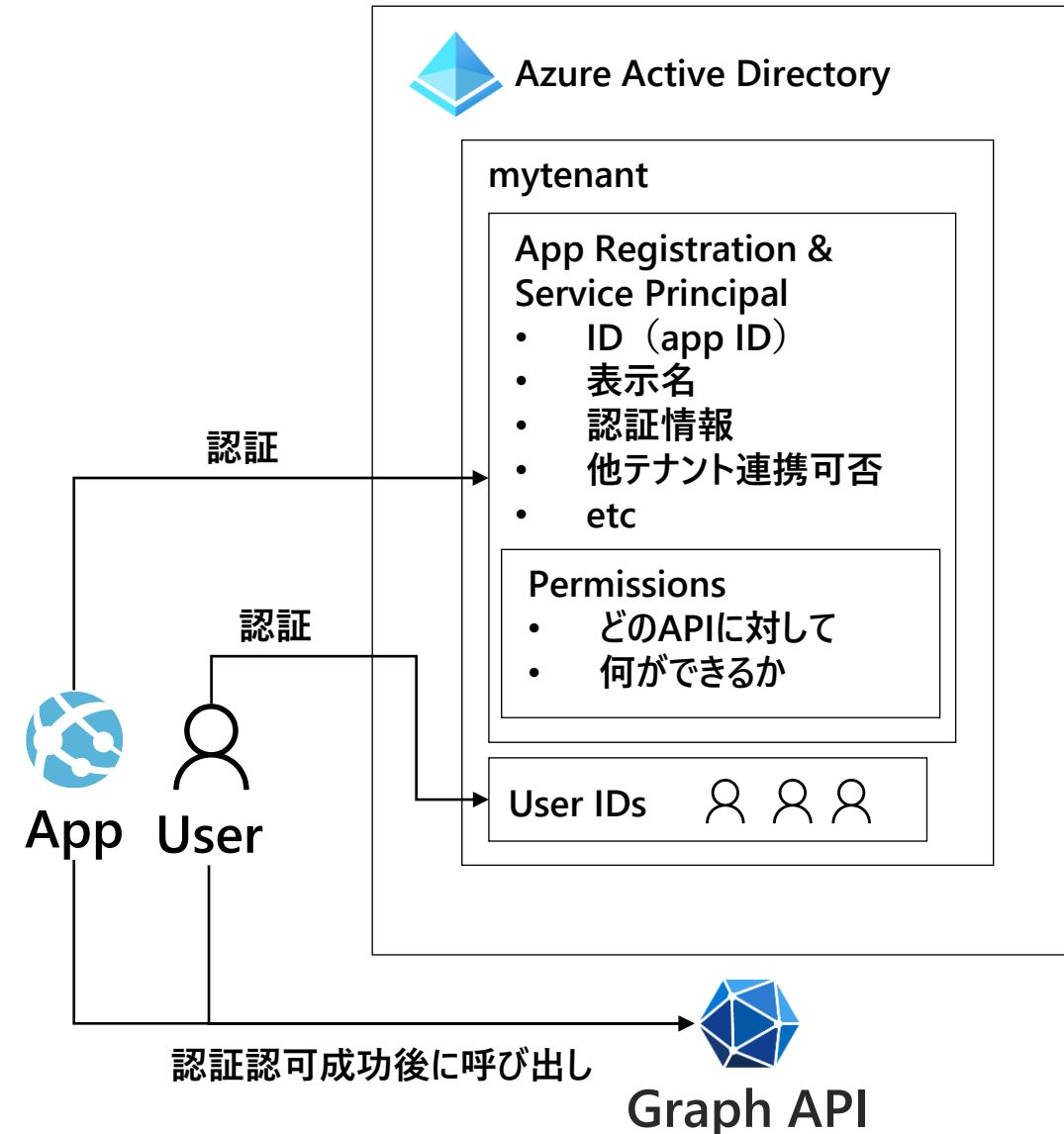


# 参考: OAuth2 のシーケンス



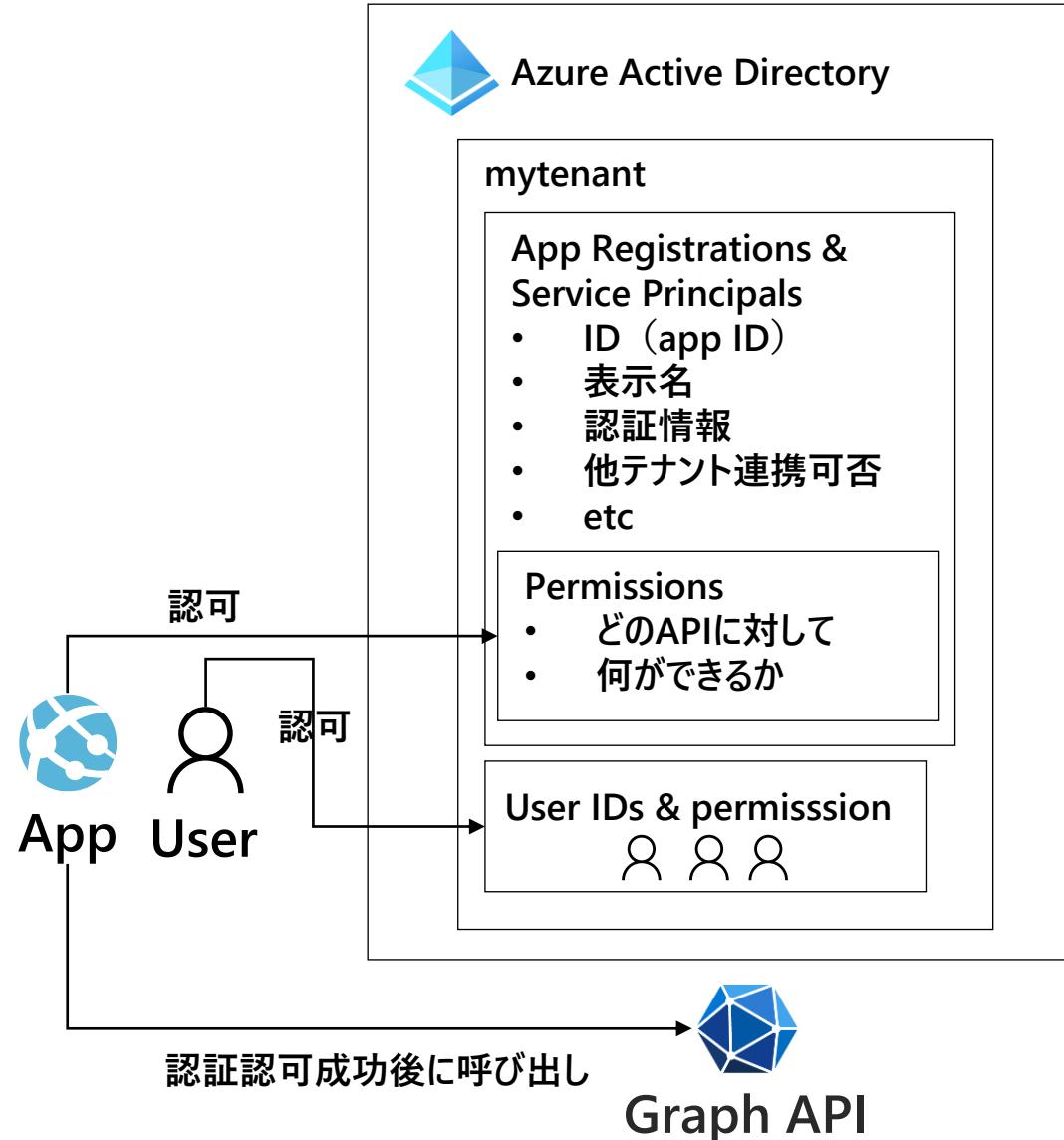
# Graph API に対する認証認可の認証 ~ Authentication ~

- ユーザもアプリケーションもGraph APIに接続するためには、Azure Active Directoryに存在が登録されている必要がある
- アプリケーションもユーザと同じく、登録作業が必要でありそれを、「App Registration / アプリ登録」と呼ぶ
- アプリ登録をすると、アプリケーションのアイデンティティである **Service Principal** が生成される
- Service Principal のシークレット（パスワード）シークレットを用いて、自身をAzure ADに認証



# Graph API に対する認証認可の認可 ~Authorization~

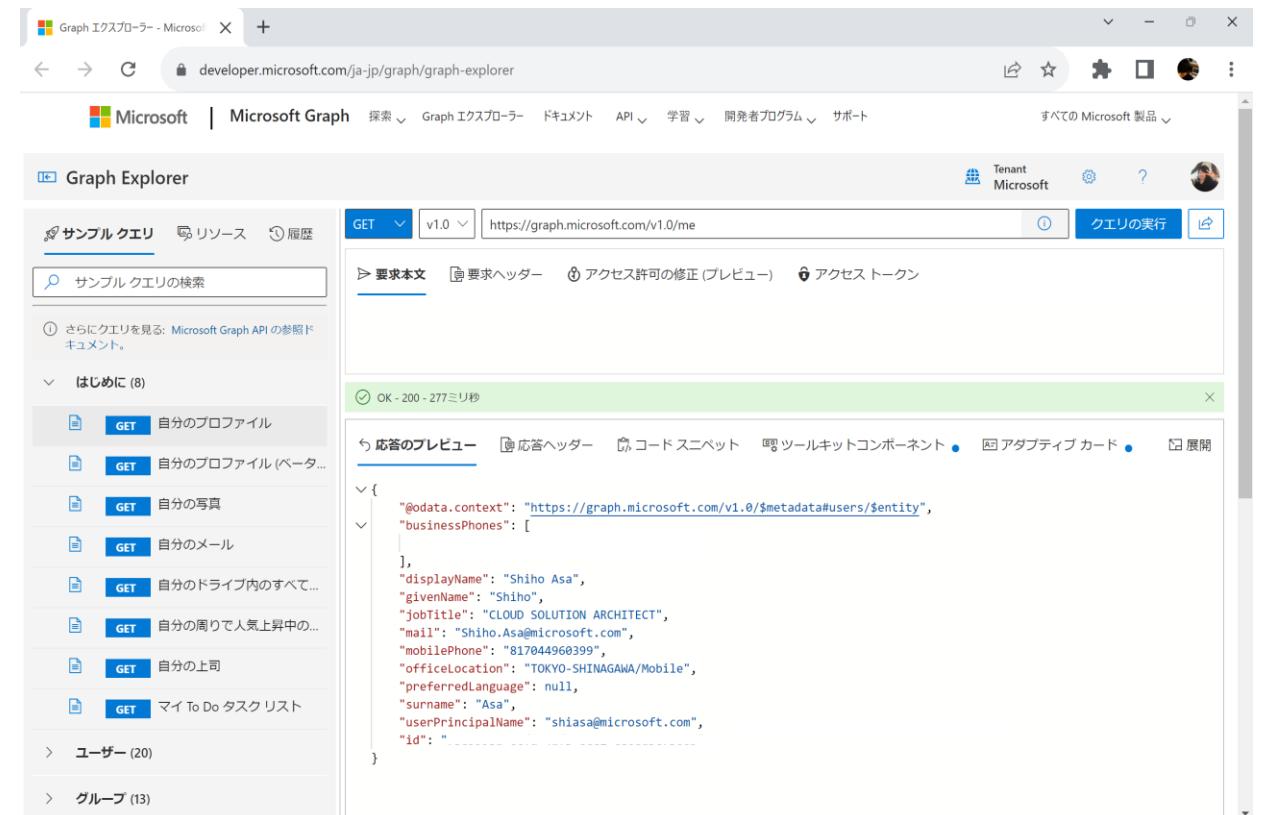
- ユーザもアプリケーションもそれぞれ、データに対する閲覧権限・更新権限が異なる
- 基本的には、既定で自身のデータに対する閲覧権限・更新権限はあるが、自身のデータ以外の場合は、Azure Active Directoryの管理者から権限をもらう必要がある
- 例えば、アプリケーションがユーザのカレンダーに会議を作成する場合は、Calendars.ReadWriteという権限が必要



# 参考: Microsoft Graph エクスプローラ

- Graph APIで取得できる情報を確認できるサイト
- Microsoft では、Graph API のインターフェースおよびパラメータ値などを手軽に確認（検証）できるように Graph Explorer を提供
- 一般的な Graph API 処理についてはアプリケーションを開発することなく、Graph Explorer (Web サイト) で確認することが可能

<https://developer.microsoft.com/ja-jp/graph/graph-explorer>



<https://docs.microsoft.com/ja-jp/learn/modules/msgraph-intro-overview/4-graph-explorer>

# 参考: Microsoft Graph ツールキット

- 任意のJavaScriptフレームワークを使用してアプリを Microsoft 365 に接続するために使用する Web コンポーネント
- React用/SPFx用ツールキットもあり
- ログイン・ユーザプロファイル・カレンダー・タスク・プランナーなどのコンポーネントが用意されている

```
npm install @microsoft/mgt-react
```

A screenshot of a web application interface featuring a person card component. The card displays a profile picture of Megan Bowen, her name, title (Auditor), and department (Finance). Below the card are two buttons: "Send email" and "Start chat". At the bottom, there are tabs for "html", "js", and "css". The "js" tab is active, showing the following code:

```
1 <mgt-person-card person-query="me"></mgt-person-card>
2
```

A screenshot of a web application interface featuring an agenda component. It shows a list of events: "Company Meeting" from 9:30 AM - 12:00 PM and "New Product Regulations Touchpoint" from 11:00 AM - 11:30 AM. Both events are located in "Conf Room Rainier". To the right, there is a thumbnail of a group of people and the text "+5". At the bottom, there are tabs for "html", "js", and "css". The "js" tab is active, showing the following code:

```
1 <mgt-agenda></mgt-agenda>
2
```

A screenshot of a web application interface featuring a people component. It displays a grid of user profiles. At the bottom, there are tabs for "html", "js", and "css". The "js" tab is active, showing the following code:

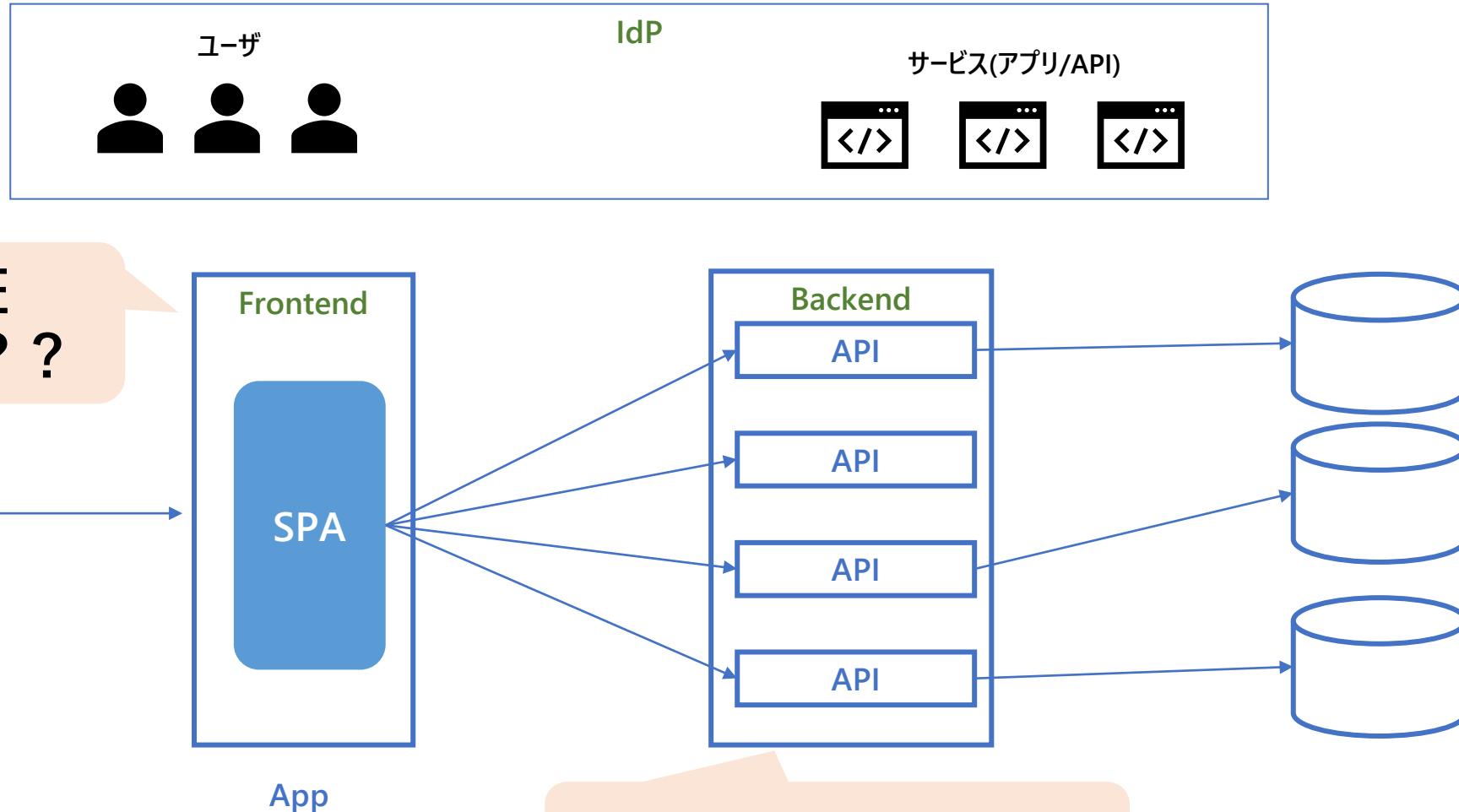
```
1 <mgt-people show-max="5"></mgt-people>
2
```

<https://learn.microsoft.com/ja-jp/graph/toolkit/overview>

<https://learn.microsoft.com/ja-jp/training/modules/msgraph-toolkit-intro/>

## Part 2: AppService を使ったWeb アプリケーションの認証認可

# クラウドネイティブなWebアプリケーションの基本構成



# Web アプリで Azure AD 認証を構成するには

## Web アプリケーションでやること

適切なトークンを持つリクエストに応じたアクセス制御を行う

適切なトークン情報を持たない場合 Azure AD の認可エンドポイントへ誘導する

その際に必要とするアプリケーション情報を指定する

トークンを返却してもらうための URI を指定する

相互に設定情報を事前に交換しておく

## Azure Active Directoryでやること

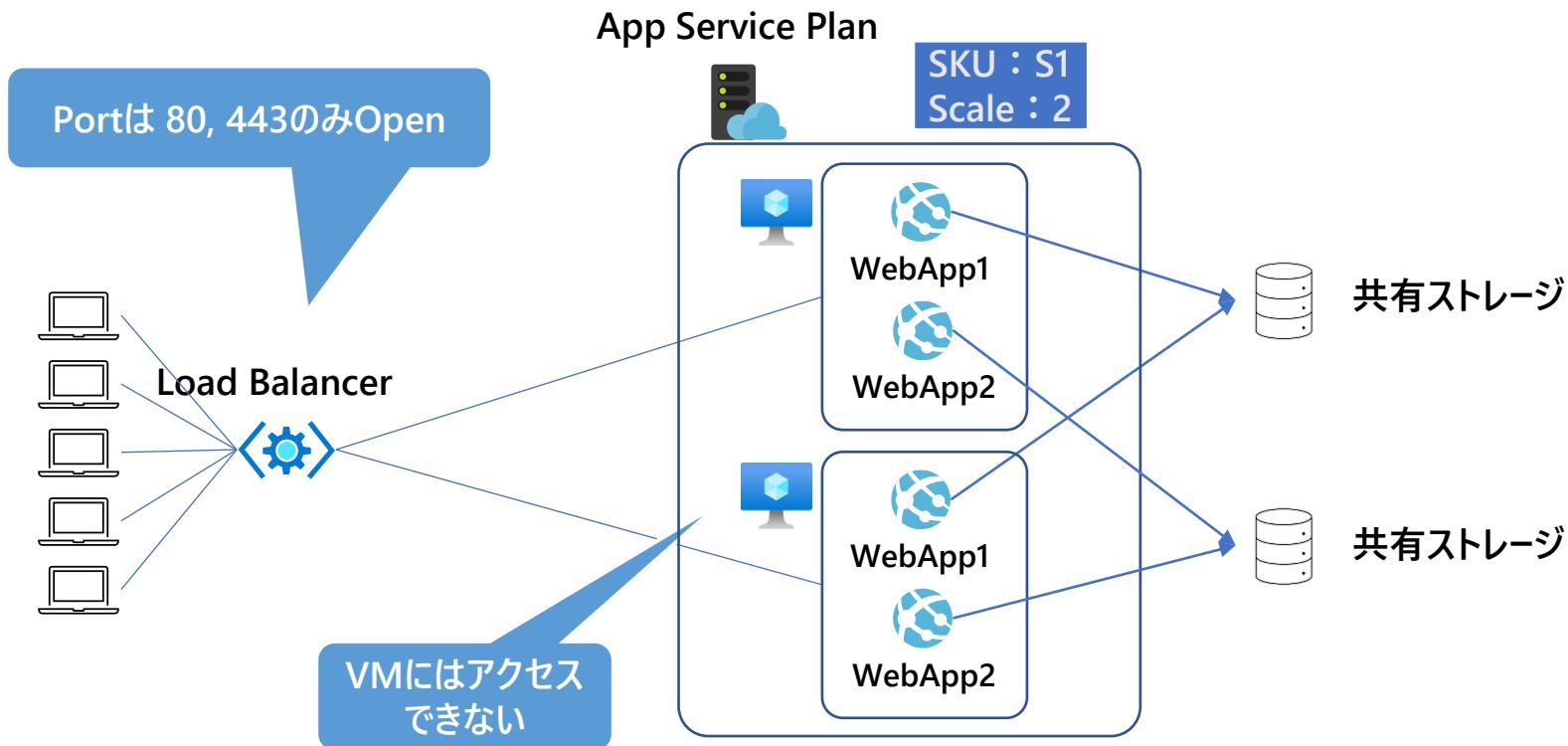
アクセスしているユーザーの正当性を確認する

登録されたアプリに対してユーザーの同意を取得する  
トークンを発行してアプリケーションに返却する

この設定により Microsoft 365 と同等の認証処理が可能なアプリケーションが構築できる

# AppService とは

- ・ 負荷分散機能やスケーリング機能を提供するアプリケーション実行環境のマネージドサービス
- ・ ロードバランサ/WebアプリケーションをホスティングするVM 共有ストレージから構成
- ・ App Service Plan という Azure リソースが VM のサイズや数を定義し、その上にWebアプリケーションをホスティング
- ・ 専用の管理ツール(Kudu)/認証認可/



# AppService とは

- Webアプリケーションの実行必要な非機能要件が組み込み機能として用意されている

デプロイメント

- クイック スタート
- デプロイ スロット
- デプロイ オプション
- デプロイ センター (プレビュー)

App Service プラン

- App Service プラン
- クォータ
- App Service プランの変更

設定

- アプリケーション設定
- コンテナーの設定
- 認証/承認**
- Application Insights
- マネージド サービス ID
- バックアップ
- カスタム ドメイン
- SSL 設定
- ネットワーク
- スケール アップ (App Service...)
- スケール アウト (App Service...)

監視

- アラート (クラシック)
- 診断ログ
- ログ ストリーム
- プロセス エクスプローラー
- ライブ HTTP トラフィック
- 開発ツール
- アプリの複製
- SSH
- 高度なツール
- App Service Editor (プレビュ...)
- パフォーマンス テスト
- リソース エクスプローラー
- 運用環境でのテスト
- 拡張機能

監視

リソース グループ (変更)  
pakuuestaticsabf0  
状態  
Running  
場所  
Southeast Asia  
サブスクリプション (変更)  
MTC Tokyo サブスクリプション  
サブスクリプション ID  
c278d8fb-fe4b-4594-b3d3-207b1fa8e91e  
タグ (変更)  
タグを追加するにはここをクリック

問題の診断と解決  
セルフサービス機能により、Web アプリケーションのクラッシュやパフォーマンス低下などの問題を識別して解決するのに役立ちます。

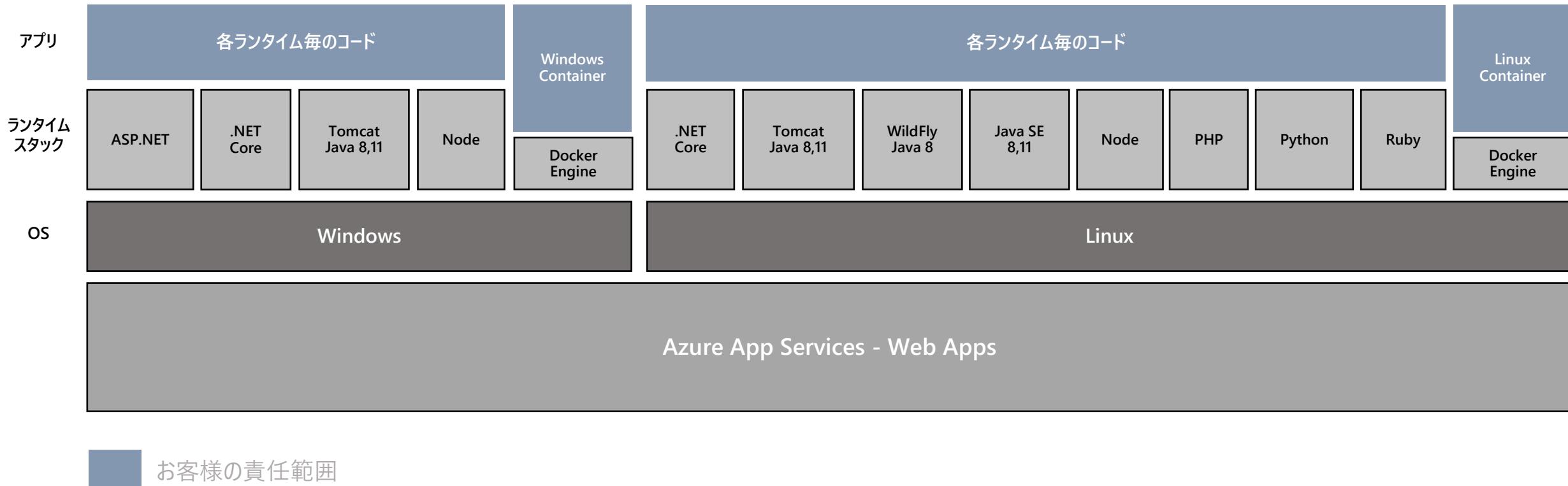
App Service Advisor  
App Service Advisor は、App Service プラットフォームのアラートとエラーハンドリングを向上するための分析情報を提供するものです。推奨事項はその詳細、優先度、アリに対する影響に基づいて整理された状態で提示されます。

HTTP 5xx  
100  
80

受信データ  
100B  
80B

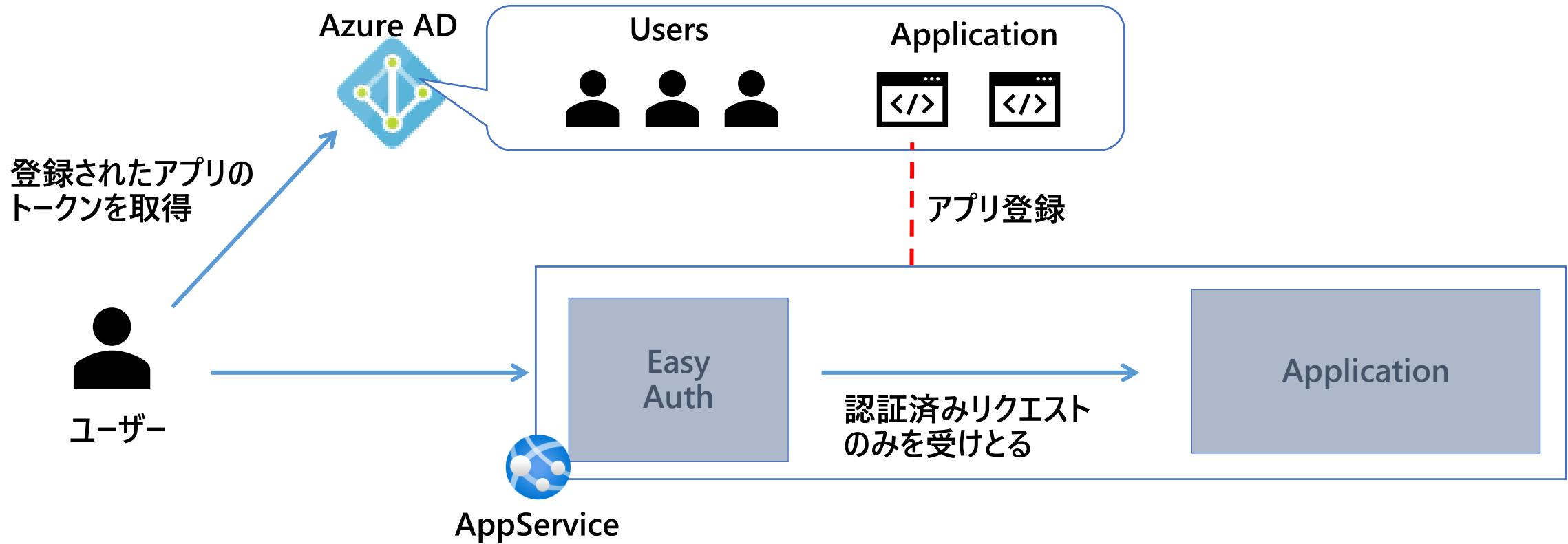
# AppService がサポートする構成

- Microsoft が OS の運用管理やランタイムスタックのセットアップ・運用を行う
- 利用者は「アプリケーションのコード」もしくは「コンテナの中身」にのみ集中することが可能



# AppService 組み込み認証 の動作イメージ

- Web Apps は Azure AD を信頼し、認証を委託する “クライアント” アプリケーション
- ユーザーは AAD で認証を受け、Web App が自身の情報にアクセスすることに同意する
- Web Apps は指定のディレクトリで発行された認可コードを持つリクエストを処理する



# AppService の組み込み認証の構成

- AADへのアプリ登録、設定情報の交換、トークンチェック、トークンがない場合のリダイレクト、ユーザーアプリへのトークンの伝搬などを自動で行う
- Azure ポータルで認証に使用している AADテナントでユーザー認証する場合、かつ、Azureポータルのログインユーザーにアプリ登録権限がある場合に限る

## AppServiceの 認証設定



## Azure AD に登録されたアプリ

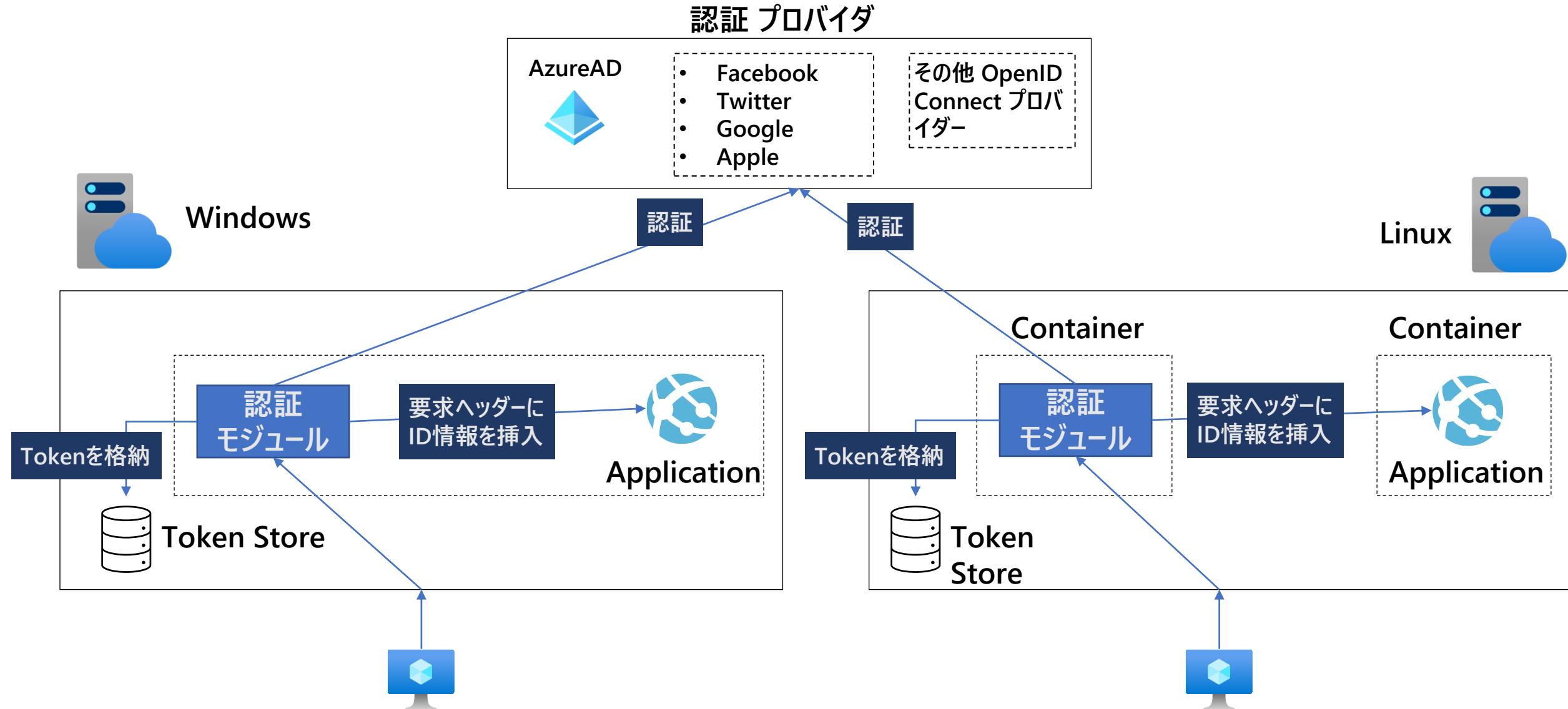
# ユーザ割り当て

アプリケーションへのアクセスをディレクトリ内の特定のユーザーに限定したい場合は「ユーザーの割り当て」を行う  
組み込み認証を有効にしただけでは、当該ディレクトリに登録されたユーザーであると認証されているだけ  
(=組織のユーザーであれば全員使用できる)



表示名	オブジェクトID	ユーザー
Ayumu Inaba	00000000-0000-0000-0000-000000000000	ユーザー
Inaba Ayumu	00000000-0000-0000-0000-000000000000	ユーザー
WV tenant creator	00000000-0000-0000-0000-000000000000	ユーザー
sal admin	00000000-0000-0000-0000-000000000000	ユーザー

# AppService の組み込み認証 の構成



# AppService の組み込み認証

- 組み込みの認証と認可の機能 (Easy Auth と呼ばれることがある) を提供
- Web アプリ / RESTful API / モバイル バックエンド / Azure Functions でユーザーのサインインを追加  
HTTP リクエストヘッダーに ID 情報を挿入する
- 次の認証プロバイダーをサポート

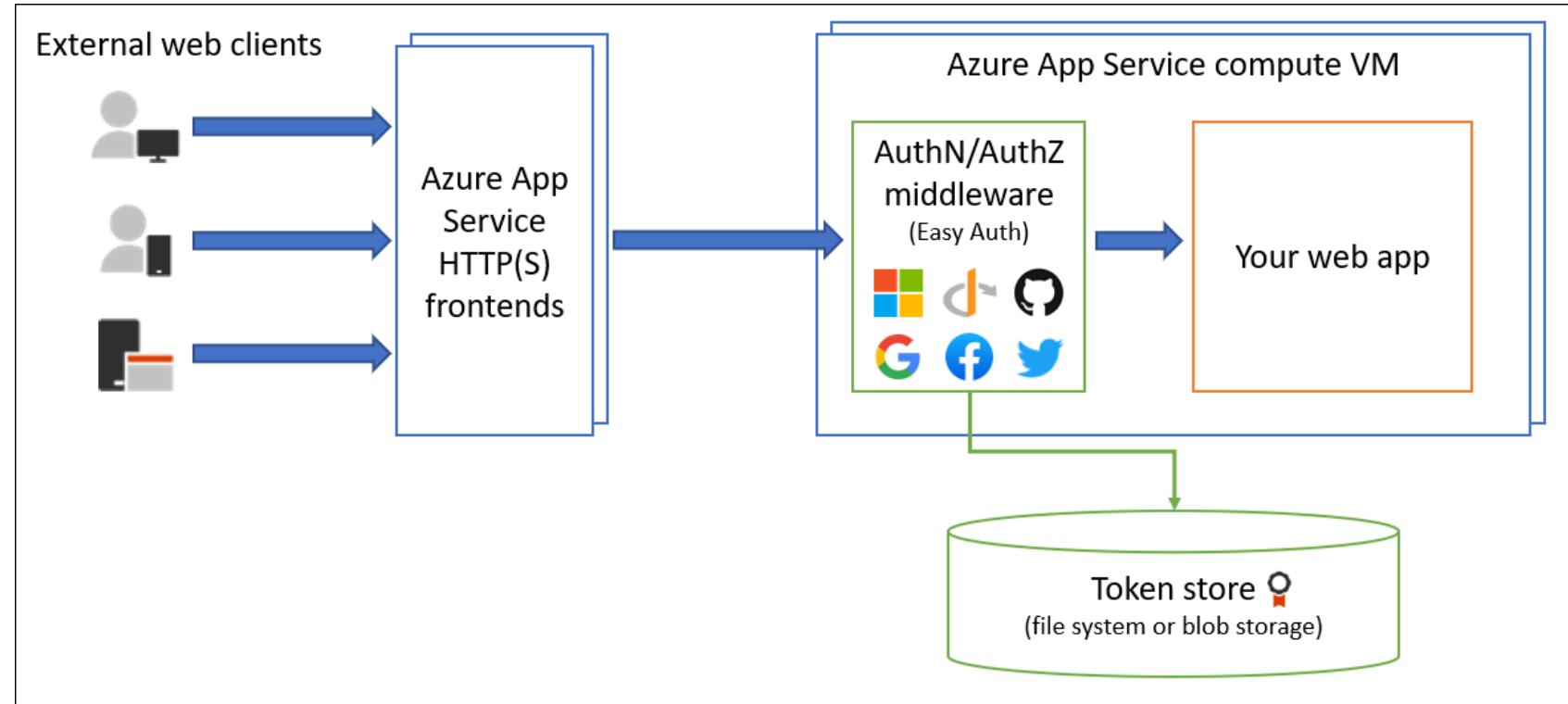
プロバイダー	サインイン エンドポイント	使用方法に関するガイダンス
<a href="#">Microsoft ID プラットフォーム</a>	<code>/.auth/login/aad</code>	<a href="#">App Service Microsoft ID プラットフォーム ログイン</a>
<a href="#">Facebook</a>	<code>/.auth/login/facebook</code>	<a href="#">App Service Facebook ログイン</a>
<a href="#">Google</a>	<code>/.auth/login/google</code>	<a href="#">App Service Google ログイン</a>
<a href="#">Twitter</a>	<code>/.auth/login/twitter</code>	<a href="#">App Service Twitter ログイン</a>
<a href="#">OpenID Connect プロバイダー</a>	<code>/.auth/login/&lt;providerName&gt;</code>	<a href="#">App Service OpenID Connect ログイン</a>

<https://learn.microsoft.com/ja-jp/azure/app-service/tutorial-auth-aad?pivots=platform-linux>

# AppService の組み込み認証

## プラットフォーム ミドルウェアとは

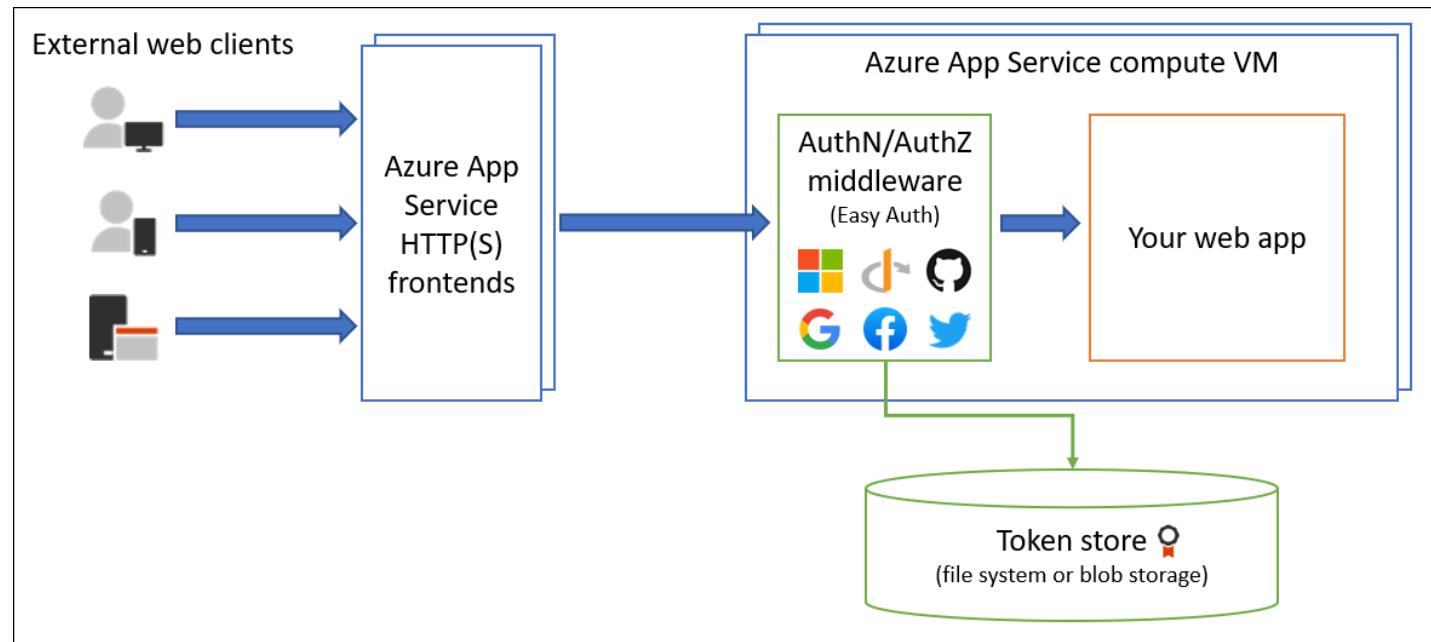
- 指定された ID プロバイダーを使用してユーザーとクライアントを認証する
- 構成済みの ID プロバイダーによって発行された OAuth トークンを検証 / 格納 / 更新する
- 認証されたセッションを管理



# AppService の組み込み認証

## トークンストア とは

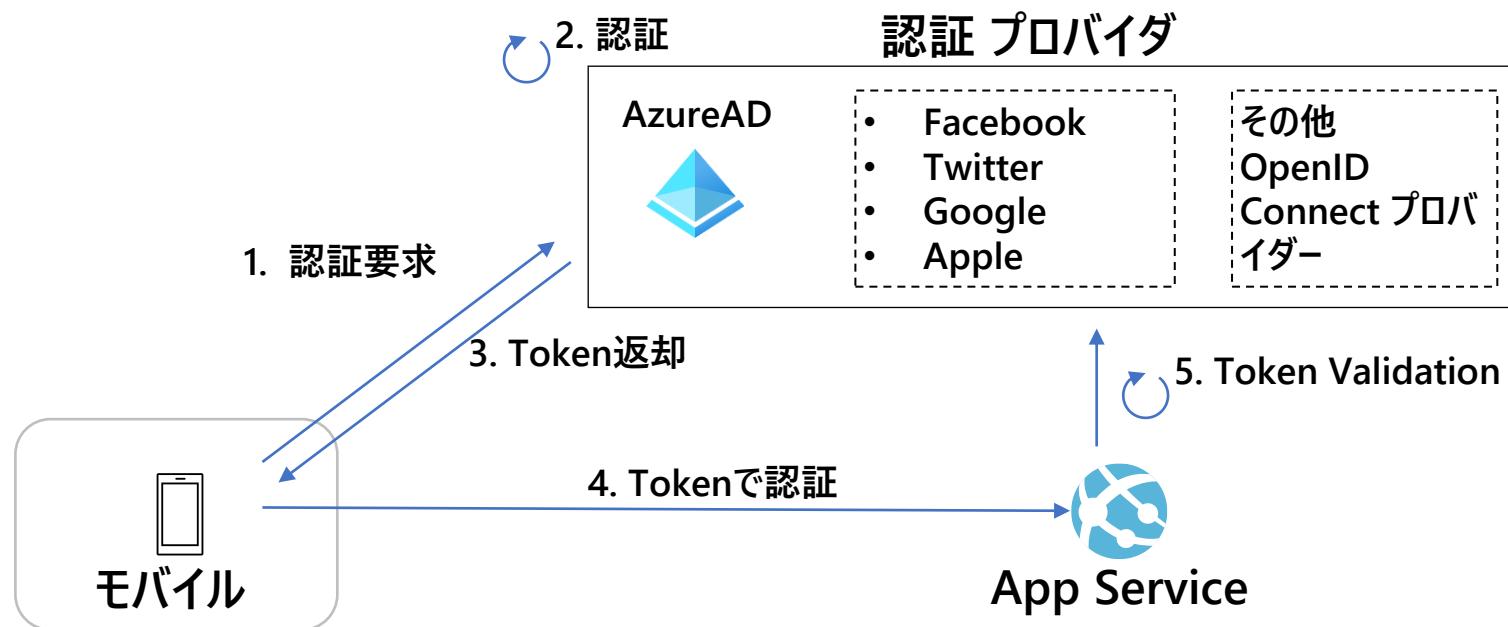
- Web アプリ/REST API / モバイル アプリのユーザーに関連付けられているトークンのリポジトリ
- トークンが必要になったらトークンを取得し、トークンが無効になったらトークンを更新するよう App Service に指示する
- ID トークン/アクセス トークン/リフレッシュトークンは認証されたセッションに対してキャッシュされ、関連付けられているユーザーだけがアクセスできる
- アプリでトークンを使う必要がない場合は、トークンストアを無効にできる



# AppService の認証フロー

## 認証プロバイダーの SDK を使う場合

- REST API、Azure Function、JavaScript ブラウザー クライアント、柔軟なサインイン プロセスを必要とするブラウザー アプリで用いられるフロー
- アプリケーションは、ユーザーを手動でプロバイダーにサインインさせてから、検証のために App Service に認証トークンを送信
- アプリケーションのコードがサインイン プロセスを管理するので、“クライアント主導のフロー” または “クライアント フロー” とも呼ばれる
- 認証プロバイダーの SDK を使ってユーザーをサインインさせるネイティブ モバイル アプリにも適用



# トークンの取得

- AppServiceがプロバイダー固有のトークンをリクエストヘッダーに挿入する
- クライアントコード(モバイルアプリやブラウザー内のJavaScriptなど)の場合は、HTTP GET要求を /.auth/me に送信するとJSON形式で取得できる
- トークンストアを有効にする必要がある

プロバイダー	ヘッダー名
Azure Active Directory	X-MS-TOKEN-AAD-ID-TOKEN X-MS-TOKEN-AAD-ACCESS-TOKEN X-MS-TOKEN-AAD-EXPIRES-ON X-MS-TOKEN-AAD-REFRESH-TOKEN
Facebookトークン	X-MS-TOKEN-FACEBOOK-ACCESS-TOKEN X-MS-TOKEN-FACEBOOK-EXPIRES-ON
Google	X-MS-TOKEN-GOOGLE-ID-TOKEN X-MS-TOKEN-GOOGLE-ACCESS-TOKEN X-MS-TOKEN-GOOGLE-EXPIRES-ON X-MS-TOKEN-GOOGLE-REFRESH-TOKEN
Twitter	X-MS-TOKEN-TWITTER-ACCESS-TOKEN X-MS-TOKEN-TWITTER-ACCESS-TOKEN-SECRET

```
[  
  {  
    "access_token": "xxx",  
    "expires_on": "2022-10-13T09:55:19.3734464Z",  
    "id_token": "xxx",  
    "provider_name": "aad",  
    "user_claims": [  
      {  
        "typ": "aud",  
        "val": "xxx"  
      },
```

<https://learn.microsoft.com/ja-jp/azure/app-service/configure-authentication-oauth-tokens#retrieve-tokens-in-app-code>

<https://learn.microsoft.com/ja-jp/azure/app-service/tutorial-auth-aad?pivots=platform-linux>

# トークンの更新

- プロバイダーのアクセストークンが期限切れになった場合は、アプリケーションの `/.auth/refresh` エンドポイントに GET 呼び出しを行って、トークンの期限切れを回避することができる
- App Service は認証されたユーザーのトークンストア内のアクセストークンを自動的に更新する
- トークンの更新が動作するためには、トークンストアにプロバイダーの更新トークンが含まれている必要がある

```
function refreshTokens() {  
  let refreshUrl = "/.auth/refresh";  
  $.ajax(refreshUrl) .done(function() {  
    console.log("Token refresh completed successfully.");  
  }) .fail(function() {  
    console.log("Token refresh failed. See application logs for details.");  
  });  
}
```

<https://learn.microsoft.com/ja-jp/azure/app-service/configure-authentication-oauth-tokens#retrieve-tokens-in-app-code>

# 参考: AppService 活用時の注意点 ~共同責任モデル~

## ■ OS レイヤまでは Microsoft により提供される

- 物理的な故障・セキュリティに対する対応は Microsoft により自動的になされる
- OS のセキュリティについても Microsoft により管理
- 物理的なネットワークについては Microsoft 管理だが、  
アプリケーションに対するネットワークアクセスは一部ユーザ管理となっている
- OS と表現する中には Web Apps と協調して動作する補助アプリケーション等も含む

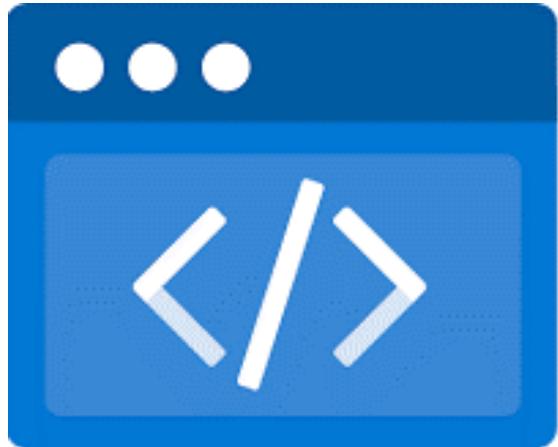
## ■ ランタイム・ミドルウェアについては共同責任モデル

- ミドルウェア・ランタイムは Microsoft 提供ではあるが、  
ログ等については Microsoft サポートで取得できない
- ログを確認したうえで問題があると疑われるようであれば Microsoft 側に問い合わせ
- ファイルサーバ領域内へのアクセス権や、VM へのログイン権限はない



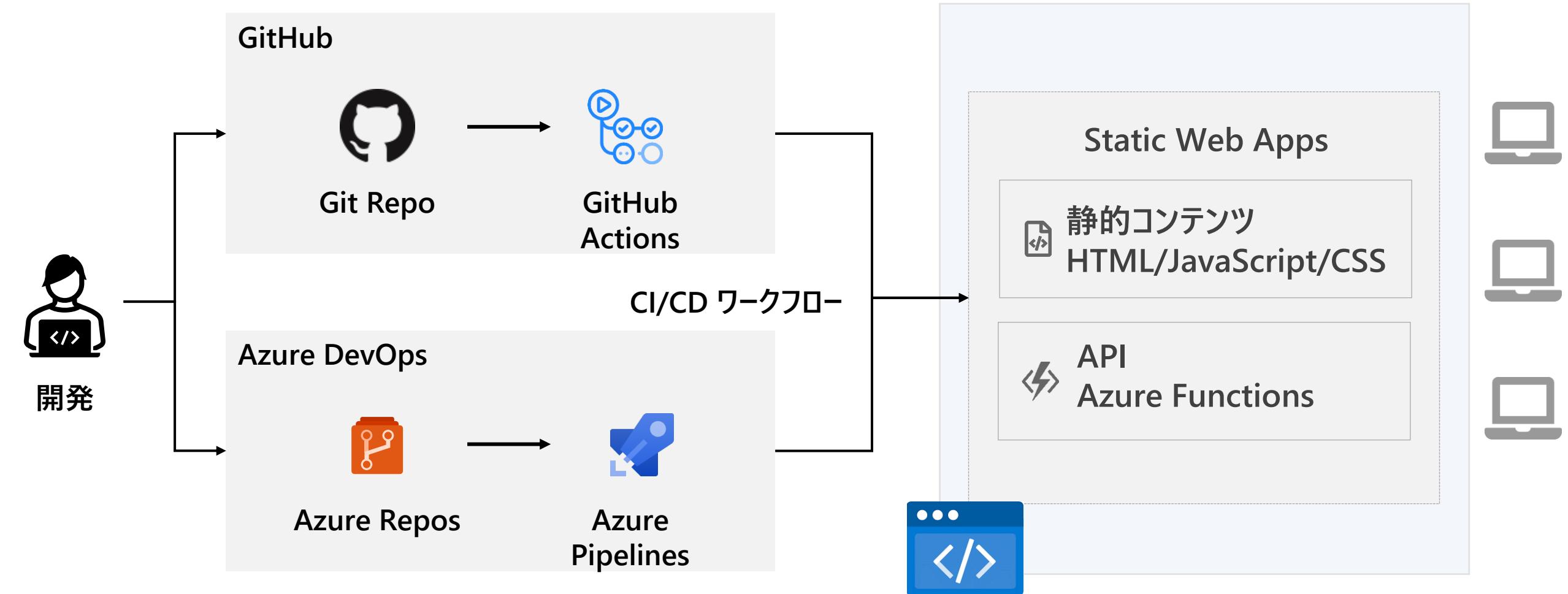
## Part 3: シングルページアプリケーションとAPIの認証認可

# Static Web Apps とは



- ・ HTML、CSS、JavaScript、画像などの**静的コンテンツの Web ホスティング**
- ・ Azure Functions によって提供される 統合 API サポート
- ・ 指定したリポジトリの変更によってビルドとデプロイがトリガーされる ファーストクラスの GitHub 統合および Azure DevOps 統合
- ・ グローバルに分散された静的コンテンツ
- ・ カスタムドメイン、無料の SSL 証明書
- ・ Azure Active Directory、GitHub、および Twitter などの**認証プロバイダーの統合**
- ・ カスタマイズ可能な認可ロールの定義と割り当て
- ・ 発行前にサイトのプレビューのために pull request を検知して Staging 環境を自動作成

# Static Web Apps の開発フロー



# Static Web Apps のプランと価格

	Free プラン	Standard プラン
価格	無料	¥1,008 / アプリ / 月
Web ホスティング	✓	✓
GitHub 統合	✓	✓
Azure DevOps 統合	✓	✓
グローバル分散された静的コンテンツ	✓	✓
SSL 証明書の無料自動更新	✓	✓
ステージング環境	アプリあたり3	アプリあたり10
アプリの最大サイズ	アプリあたり250MB	アプリあたり500MB
カスタムドメイン	アプリあたり2	アプリあたり5
API のための Functions	マネージド	マネージドまたは持ち込み
認証プロバイダー統合	事前構成済み	カスタム登録
<a href="#">SLA</a>	無し	✓

# Static Web Apps の組み込みの認証機能

- デフォルトで次の認証プロバイダーが有効 (事前の設定不要)

プロバイダー	ログイン ルート
Azure Active Directory	/.auth/login/aad
GitHub	/.auth/login/github
Twitter	/.auth/login/twitter

- ユーザーはログイン後、既定では anonymous ロールと authenticated ロールに属する
- `staticwebapp.config.json` ファイルに定義されたルールで、アクセス制限
- 組込みの招待システムを使用して、ユーザにカスタム ロール割り当て
- API 関数を使用し、ログイン時にプログラムでユーザーにカスタム ロールを割り当てることができます。
- 認証プロバイダーの制限や無効化も可能

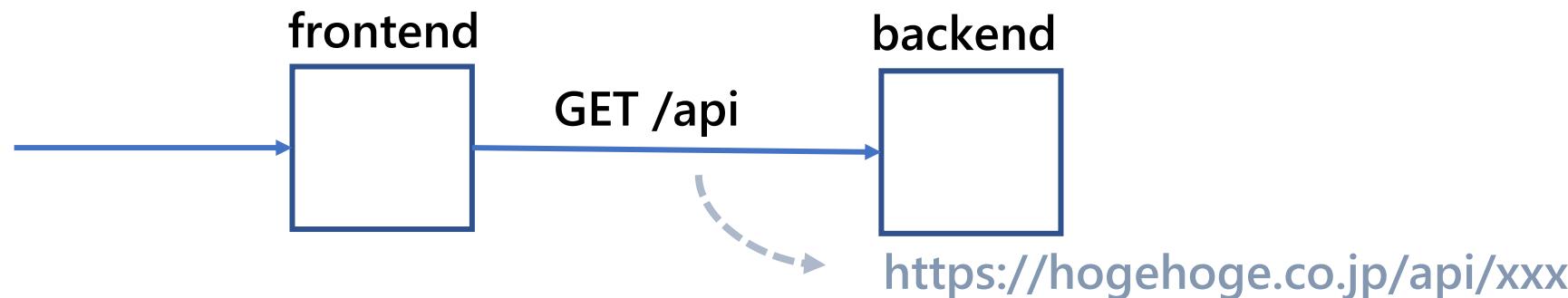
<https://learn.microsoft.com/ja-jp/azure/static-web-apps/authentication-authorization>

# Static Web Apps のAPI 連携機能

- Static Web Apps とバックエンドAPI を連携する機能

サービス	Static Web Apps マネージドAPI	ユーザが作成した API
<a href="#">Azure Functions</a>	✓	✓
<a href="#">Azure API Management</a>		✓
<a href="#">Azure App Service</a>		✓
<a href="#">Azure Container Apps</a>		✓

- カスタム CORS ルール不要でフロントエンド Web アプリから /api ルートを使用したバックエンドへのルーティング



<https://learn.microsoft.com/ja-jp/azure/static-web-apps/apis-overview>

# Static Web Apps のAPI 連携機能

- Azure Functions バックエンドを介して Static Web Apps で使用できる API 関数は クライアント アプリケーションと同じユーザー情報にアクセス可能
- API は、ユーザーを特定できる情報を受け取り、ユーザーが認証されているか、必要なロールと一致する場合は、独自の チェックを実行しない
- アクセス制御規則は、`staticwebapp.config.json` ファイルで定義

```
module.exports = async function (context, req) {
    const header = req.headers['x-ms-client-principal'];
    const encoded = Buffer.from(header, 'base64');
    const decoded = encoded.toString('ascii');

    context.res = {
        body: {
            clientPrincipal: JSON.parse(decoded),
        },
    };
};
```

<https://learn.microsoft.com/ja-jp/azure/static-web-apps/user-information?tabs=javascript>

# Static Web Apps のAPI 連携機能

- ユーザ認証とロールベースの承認 データに直接アクセスできる統合セキュリティ
- `/.auth/me` でクライアントプリンシパルデータにアクセス可能

```
async function getUserInfo() {  
  const response = await fetch('/.auth/me');  
  const payload = await response.json();  
  const { clientPrincipal } = payload;  
  return clientPrincipal;  
}  
  
console.log(await getUserInfo());
```

プロパティ	説明
<code>identityProvider</code>	<u>ID プロバイダー</u> の名前
<code>userId</code>	Azure Static Web Apps 固有のユーザーの一意識別子。値は、アプリごとに一意 同じユーザーでも、異なる Static Web Apps リソースでは異なる <code>userId</code> 値が返却される
<code>userDetails</code>	ユーザーのユーザー名またはメール アドレス ( <u>ユーザーのメール アドレス</u> を返すプロバイダーもあれば、 <u>ユーザー ハンドル</u> を送信するプロバイダーもある)
<code>userRoles</code>	<u>ユーザーに割り当てられたロール</u> の配列
<code>claims</code>	<u>カスタム認証プロバイダー</u> によって返されるクレームの配列

# 参考: 組み込み認証がある PaaS サービス

以下の3サービスに組み込み認証機能があるが、対応プロバイダーや機能・実装が違うので注意が必要

サービス	AppService (WebApps/Functions)	Container Apps	Static Web Apps
対応しているプロバイダー	<ul style="list-style-type: none"><li>• Azure AD</li><li>• Microsoft アカウント</li><li>• Facebook</li><li>• Google</li><li>• Twitter</li></ul>	<ul style="list-style-type: none"><li>• Azure Active Directory</li><li>• Facebook</li><li>• GitHub</li><li>• Google</li><li>• Twitter</li><li>• カスタム OpenID Connect</li></ul>	<ul style="list-style-type: none"><li>• Azure Active Directory</li><li>• GitHub</li><li>• Twitter</li></ul>
デフォルトで有効化されているか	×	×	○
トークンストアの有無	○	×	×
Persistent Cookie	×	×	○
/.auth/me エンドポイント	○	×	○
Client-directed sign-in	○	○	×

[Azure に実装されている 3 つの Easy Auth \(Web Apps / Static Web Apps / Container Apps\) の実装を再確認した](#)

# 参考: Microsoft Authentication Library(MSAL) とは

- ・ 主要なプラットフォーム向けの開発ライブラリを提供している
- ・ これを使うと 開発者は “いろいろ” 楽になる



JavaScript



.NET



Android



iOS



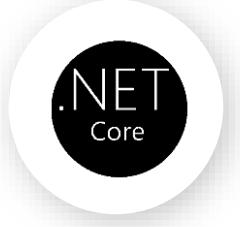
Python



Java



Angular



Microsoft  
Identity Web

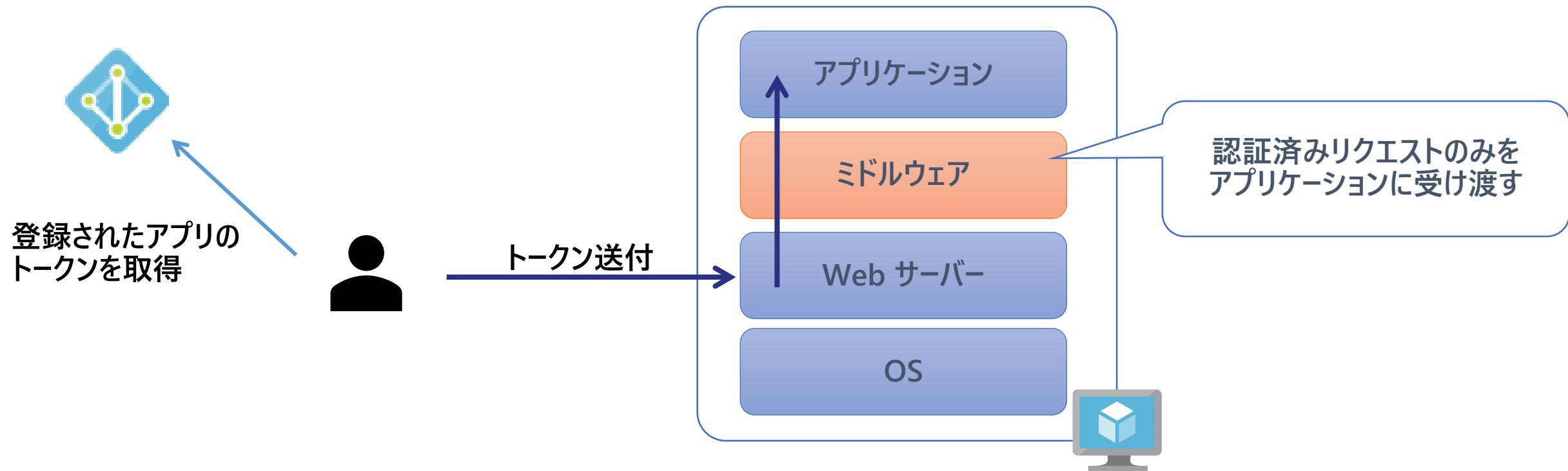
## 参考: アプリ開発者向けのドキュメント

<http://aka.ms/AADDev>

- 開発者が知りたい情報を集約したサイト
- 利用するプラットフォーム向けの MSAL ライブラリやコードサンプルがある
  - [MSAL の概要 - Microsoft identity platform | Microsoft Docs](#)
  - [Microsoft ID プラットフォームのコード サンプル | Microsoft Docs](#)
  - [Azure AD に登録できる「アプリ」と「リソース」、「API 権限」を理解する | Japan Azure Identity Support Blog \(jpazureid.github.io\)](#)

# 参考: AKS 上のアプリケーションの認証

- EasyAuthに相当する処理をお客様側でアプリケーションとして実装する必要がある
- ミドルウェアや SDKが Azure AD認証に対応している場合には、それを使用すると良い
- Spring Security など

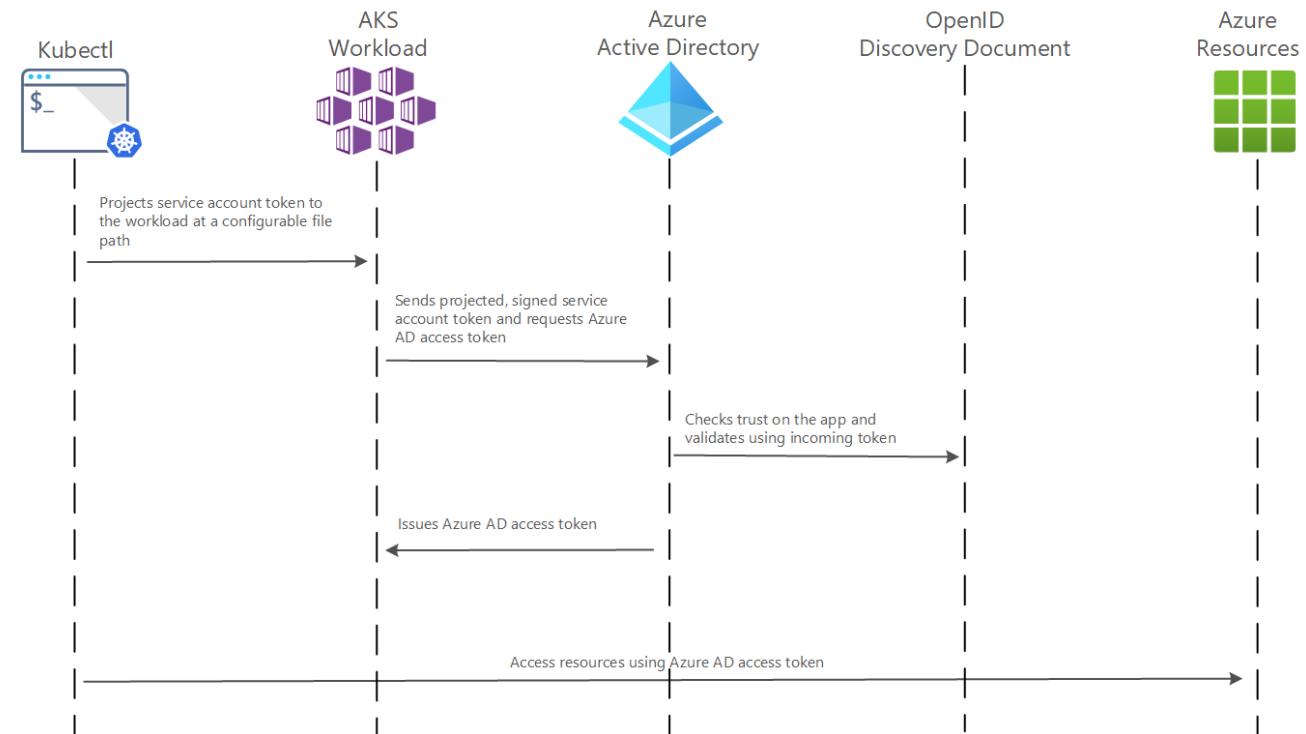


# 参考: AKS Workload Identity (プレビュー)

アプリケーション(Pod) レベルでマネージド ID を割り当てる機能

- Pod マネージド ID を使用すると、Azure ADを介して、ホストされているワークロードまたはアプリケーションからリソースにアクセスできる
- AKS クラスターがトークン発行者として機能し、Azure ADは OpenID Connect を使用して公開署名キーを検出し、サービス アカウント トークンの信頼性を確認してから、Azure AD トークンと交換
- Azure Identity SDK または Microsoft Authentication Library (MSAL)を使用して、サービス アカウント トークンを Azure AD トークンと交換

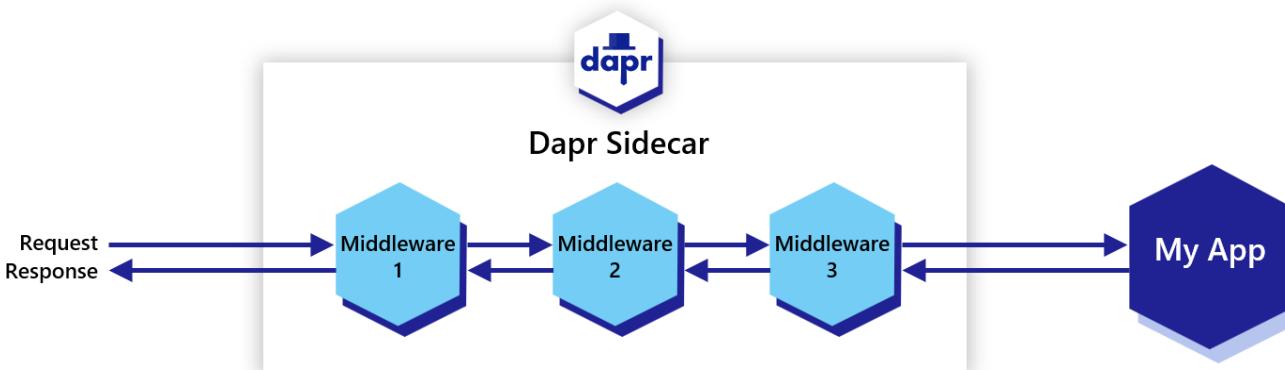
Pod Identity(プレビュー)ではなく、こちらを推奨



<https://azure.github.io/azure-workload-identity/docs/>

# 参考: Dapr Sidecar による実装

- Dapr (Distributed Application Runtime)はオープンソースの分散アプリケーションランタイムでKubernetes で動作可能
- サービス間メソッド呼び出し・状態管理・シークレットストア・パブ/サブメッセージングの機能を提供
- DaprはDapr OAuth 2.0ミドルウェアを使用すると、Authorization Code Grant フローを使用して、Dapr エンドポイントでOAuth承認ができる
- エンドポイント API に認証トークンを挿入することもできる  
(クライアント資格情報付与フロー)



```
apiVersion: dapr.io/v1alpha1
kind: Component
metadata:
  name: oauth2
  namespace: default
spec:
  type: middleware.http.oauth2
  version: v1
  metadata:
    - name: clientId
      value: "<your client ID>"
    - name: clientSecret
      value: "<your client secret>"
    - name: scopes
      value: "<comma-separated scope names>"
    - name: authURL
      value: "<authorization URL>"
    - name: tokenURL
      value: "<token exchange URL>"
    - name: redirectURL
      value: "<redirect URL>"
    - name: authHeaderName
      value: "<header name>"
```

<https://docs.dapr.io/operations/security/oauth/>

<https://docs.dapr.io/developing-applications/integrations/azure/authenticating-azure/>



# REST API に認証機能を追加しよう！

# Microsoft Cognitive Services

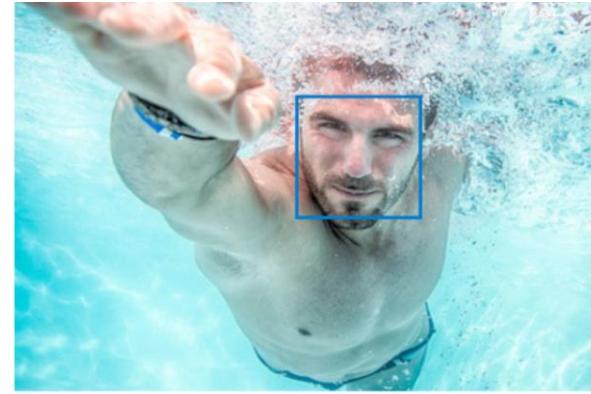
視覚

Face

Emotion

Computer Vision

Video



音声

Speech

Speaker Recognition

Custom Recognition

言語

Spell Check

Linguistic Analysis

Text Analytics

Language Understanding

知識

Academic Knowledge

Knowledge Exploration

Entity Linking

Recommendations

検索

Web Search

Image Search

News Search

Video Search

Computer Vision API

Speaker Recognition API レビュー

## Microsoft Cognitive Services

APIs Documentation > API Reference

**POST** Analyze Image

**POST** Describe Image

**POST** Get Thumbnail

**GET** List Domain Specific Models

**POST** OCR

**POST** Recognize Domain Specific Content

**POST** Tag Image

### Computer Vision API - v1.0

The Computer Vision API provides state-of-the-art algorithms to process images and return information. For example, it can be used to determine if an image contains mature content, or it can be used to find all the faces in an image. It also has other features like estimating dominant and accent colors, categorizing the content of images, and describing an image with complete English sentences. Additionally, it can also intelligently generate image thumbnails for displaying large images effectively.

#### Analyze Image

This operation extracts a rich set of visual features based on the image content.

Two input methods are supported -- (1) Uploading an image or (2) specifying an image URL. Within your request, there is an optional parameter to allow you to choose which features to return. By default, image categories are returned in the response.

A successful response will be returned in JSON. If the request failed, the response will contain an error code and a message to help understand what went wrong.

Http Method

POST

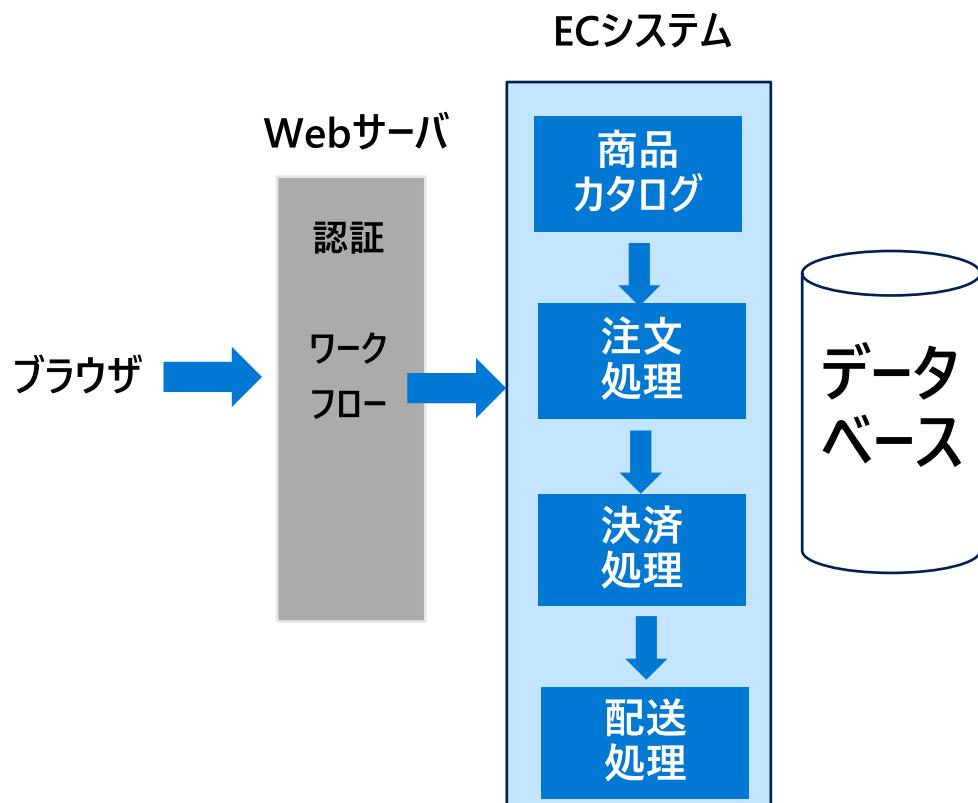
[Open API Testing Console](#)

Request URL

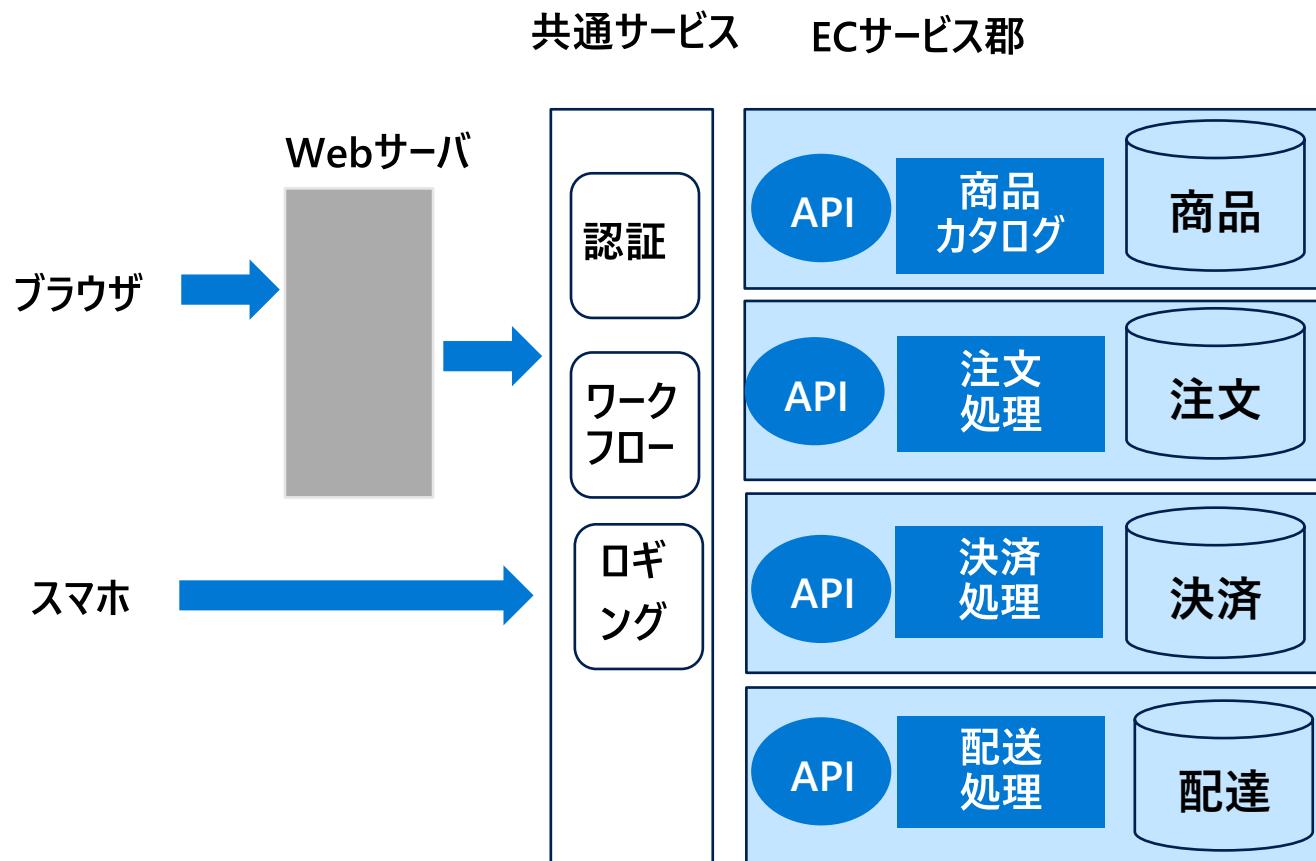
[https://api.projectoxford.ai/vision/v1.0/analyze\[?visualFeatures\]\[&details\]\[&language\]](https://api.projectoxford.ai/vision/v1.0/analyze[?visualFeatures][&details][&language])

# モノリシックからマイクロサービス (APIベース)

## モノリシック



## マイクロサービス



開発者管理

ブランディングされた  
開発者ポータル

IP フィルタリング

アクセス制限

レート制限

XML から JSON

URL のマスク

## API 提供は簡単。それだけで大丈夫ですか？

SOAP から REST

パフォーマンス

パートナーによる  
アクセス

問題追跡

利用分析

ステータス コード

キャッシュ

クロス ドメイン  
呼び出し

# Azure API Management の機能概要

## API の保護と最適化

- ポリシーでアクセス制限、データ変換、...

## 開発者エクスペリエンス

- ポータル利用で容易に開発

## 統一された管理/分析

- リアルタイムの使用量、応答性能、正常性分析

## あらゆる配置に対応

- インターネットも VPN / ExpressRoute も

## 柔軟なスケーリング

- スケールアップ / スケールアウト

# API のライフサイクル

API 構築 → API 発行 → API 利用 → API 運用



API 開発者



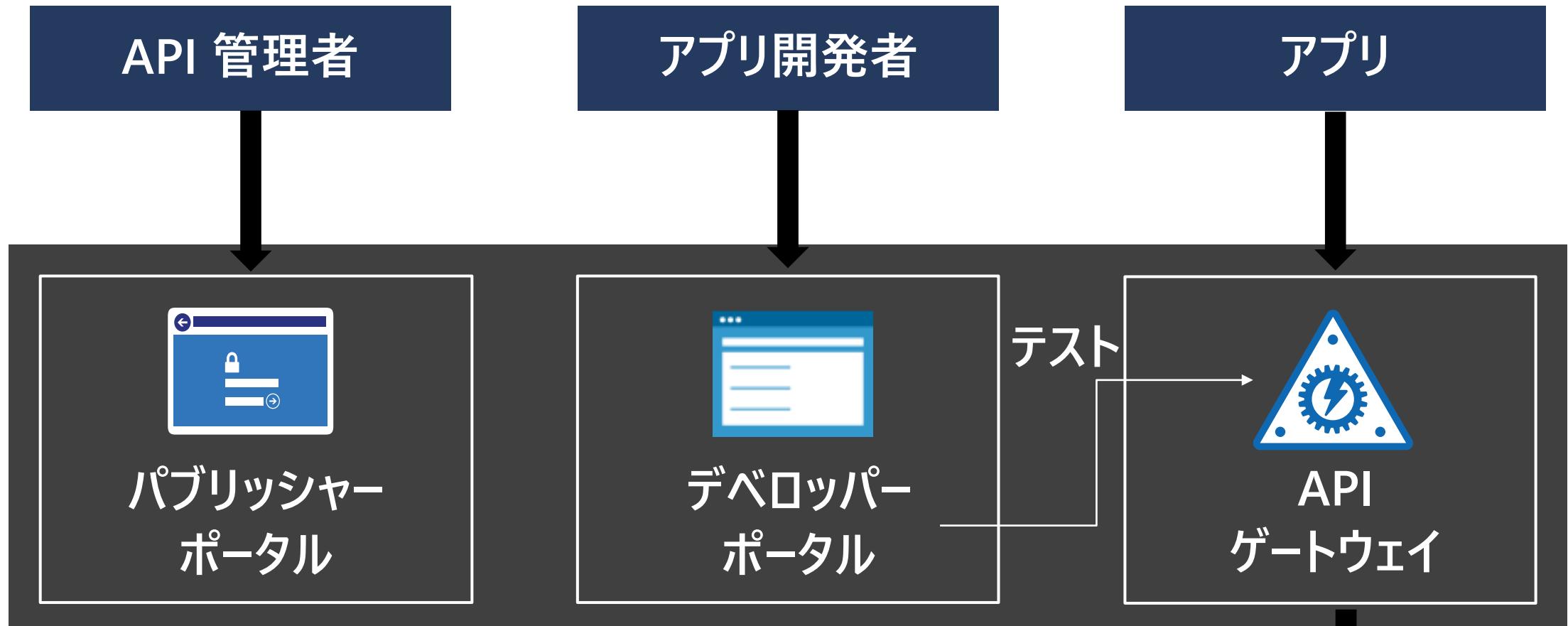
API 管理者



アプリ開発者



API 管理者



Azure API Management

API

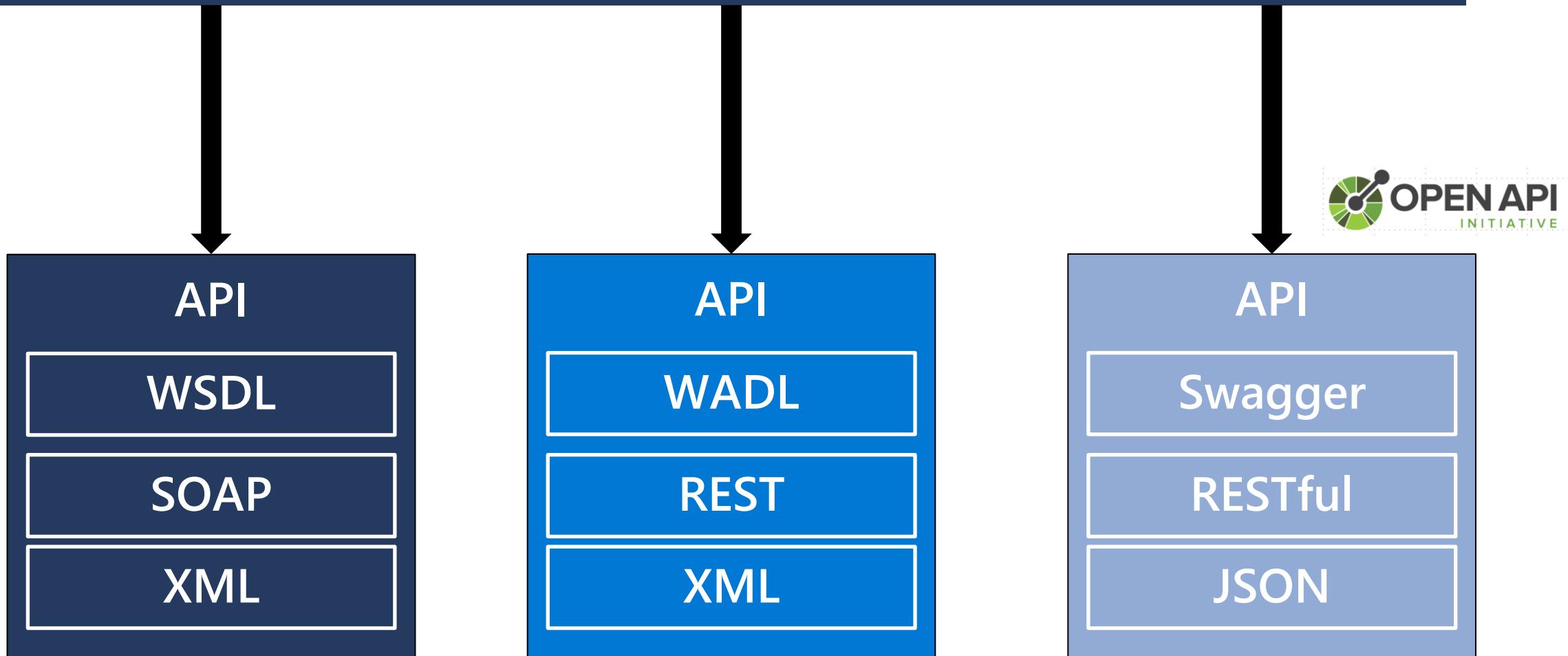
## API 構築

## API 発行

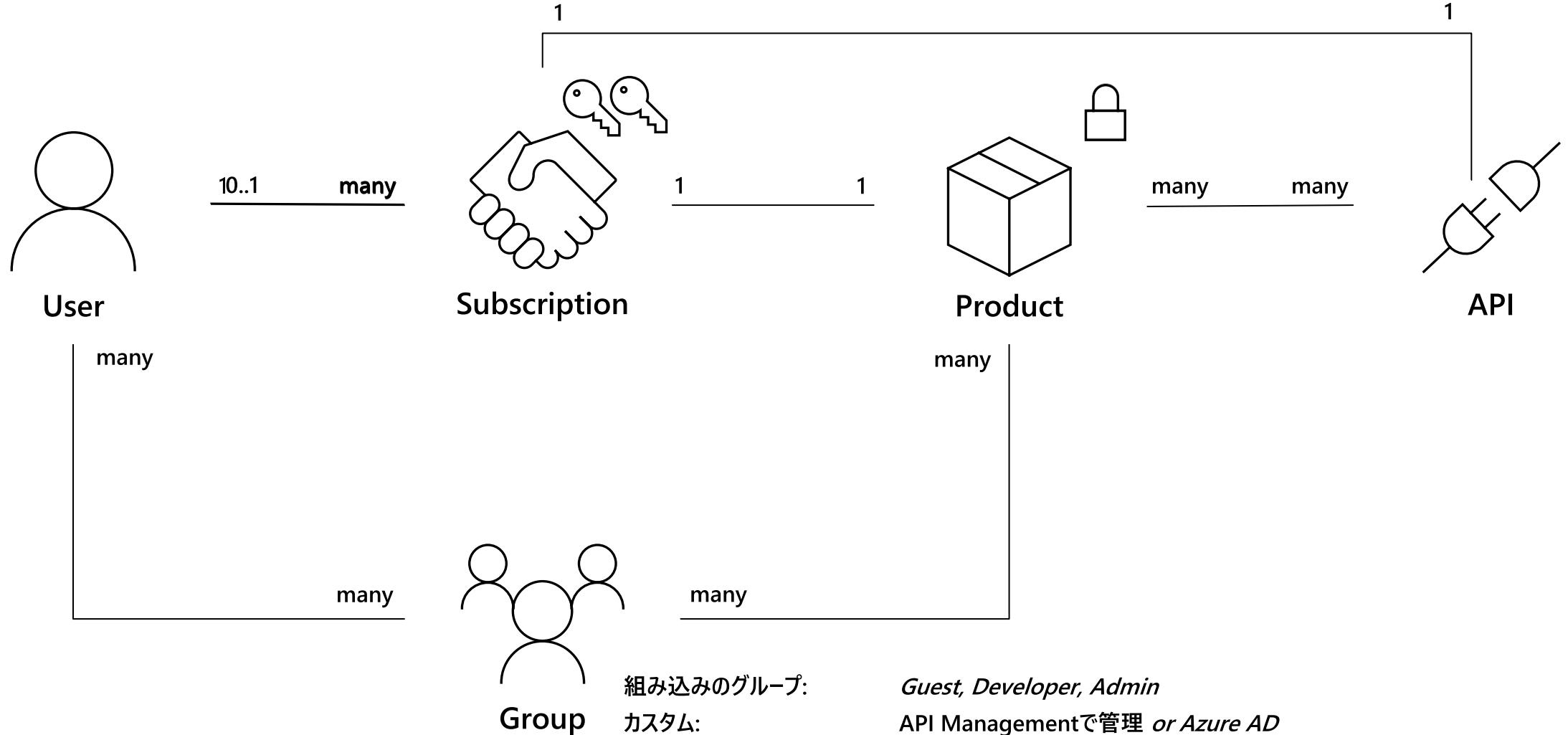
## API 利用

## API 運用

アプリ（API コンシューマ）



# APIのユーザ/グループ/製品/API/サブスクリプション の関係



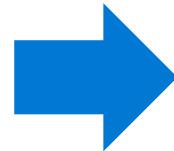
データ変換

スロットリング / quota / IP  
フィルタ...

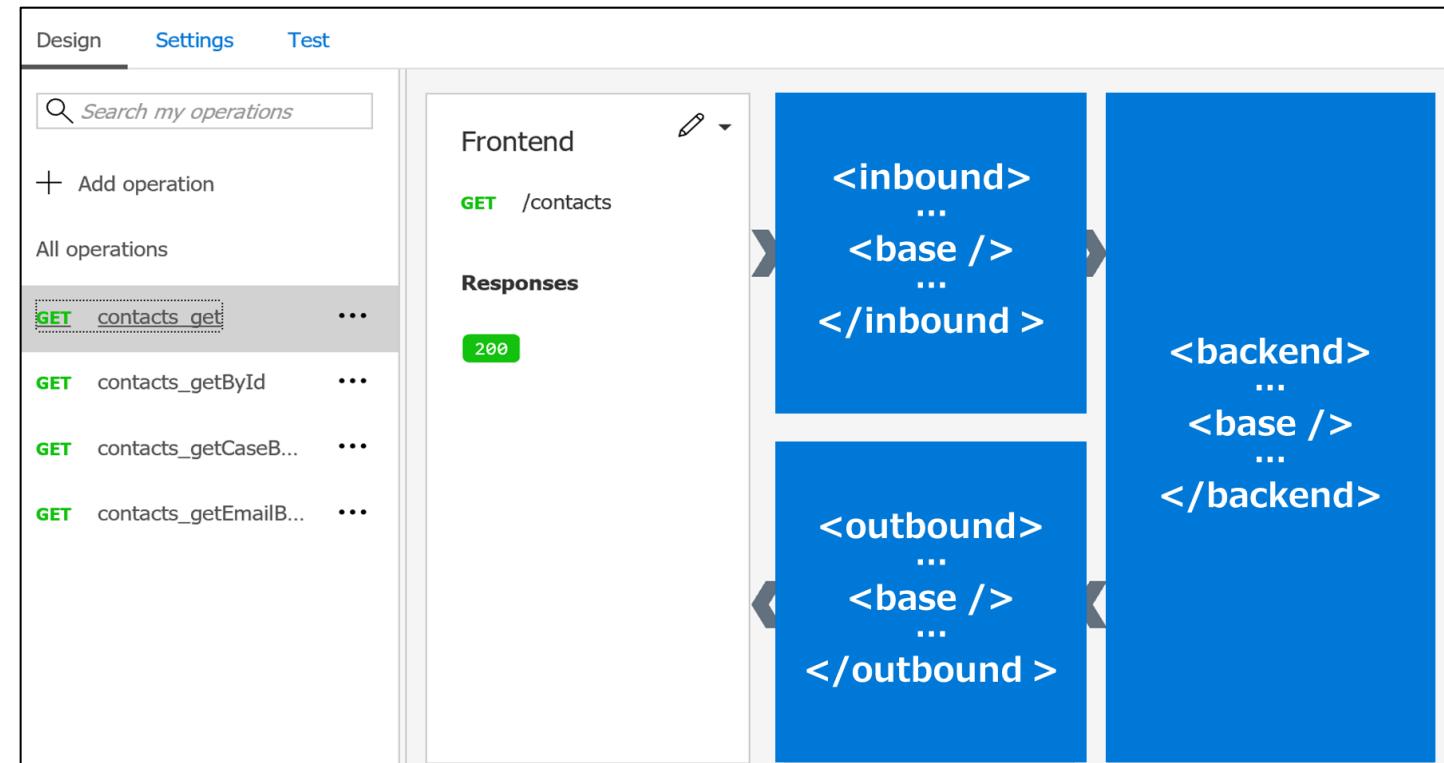
キャッシュ

外部サービス

その他いろいろ



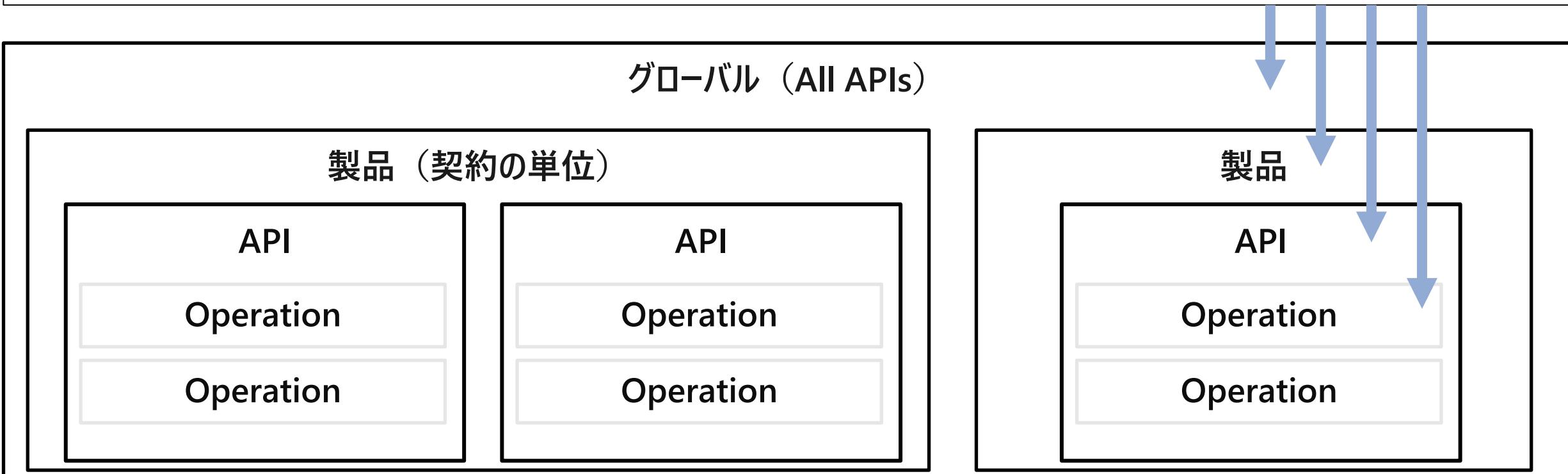
ポリシーによる API の保護と最適化



# ポリシーとスコープ

```
<rate-limit-by-key calls="10" renewal-period="60"  
counter-key="@(context.RequestIpAddress)"/>
```

```
<quota-by-key calls="10000" bandwidth="40000" renewal-period="3600"  
counter-key="@context.Subscription.Key" />
```



# ポリシーとスコープ

GET /foo/bar HTTP/1.1  
Host: api.constoso.com  
Key: 0123456789

*from caller*



inbound

0123456789

CORS

LOG

global

RATE

QUOTA

product

/foo

JWT

api

/bar

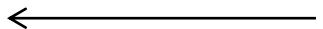
operation

CACHE

URL

BODY

*to caller*



outbound

*to backend*



# API Management のポリシー

## アクセス制限ポリシー

- HTTP ヘッダーを確認する
- 呼び出しレートをサブスクリプション別に制限する
- 呼び出しレートをキー別に制限する
- 呼び出し元 IP を制限する
- 使用量のクオータをサブスクリプション別に設定する
- 使用量のクオータをキー別に設定する
- JWT を検証する

## 高度なポリシー

- 制御フロー
- 要求を転送する
- Event Hub にログを記録する
- 再試行
- 応答を返す
- 1 方向の要求を送信する
- 要求を送信する
- 要求メソッドを設定する
- 状態コードを設定する
- 変数の設定
- トレース
- 待機

# API Management のポリシー

## 認証ポリシー

- 基本認証
- クライアント証明書による認証

## キャッシュ ポリシー

- キャッシュから取得
- キャッシュに格納
- キャッシュから値を取得
- 値をキャッシュに格納
- キャッシュから値を削除

## クロス ドメイン ポリシー

- クロスドメイン呼び出しを許可
- CORS (クロス オリジン リソース共有)
- JSONP

## 変換ポリシー

- JSON から XML への変換
- XML から JSON への変換
- 本文内の文字列の検索および置換
- コンテンツ内の URL のマスク
- バックエンド サービスの設定
- 本文の設定
- HTTP ヘッダーの設定
- クエリ文字列パラメーターの設定
- URL の書き換え

## API 構築

## API 発行

## API 利用

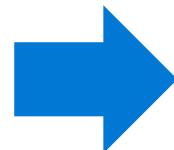
## API 運用

API の説明

テスト アプリ

サンプル コード

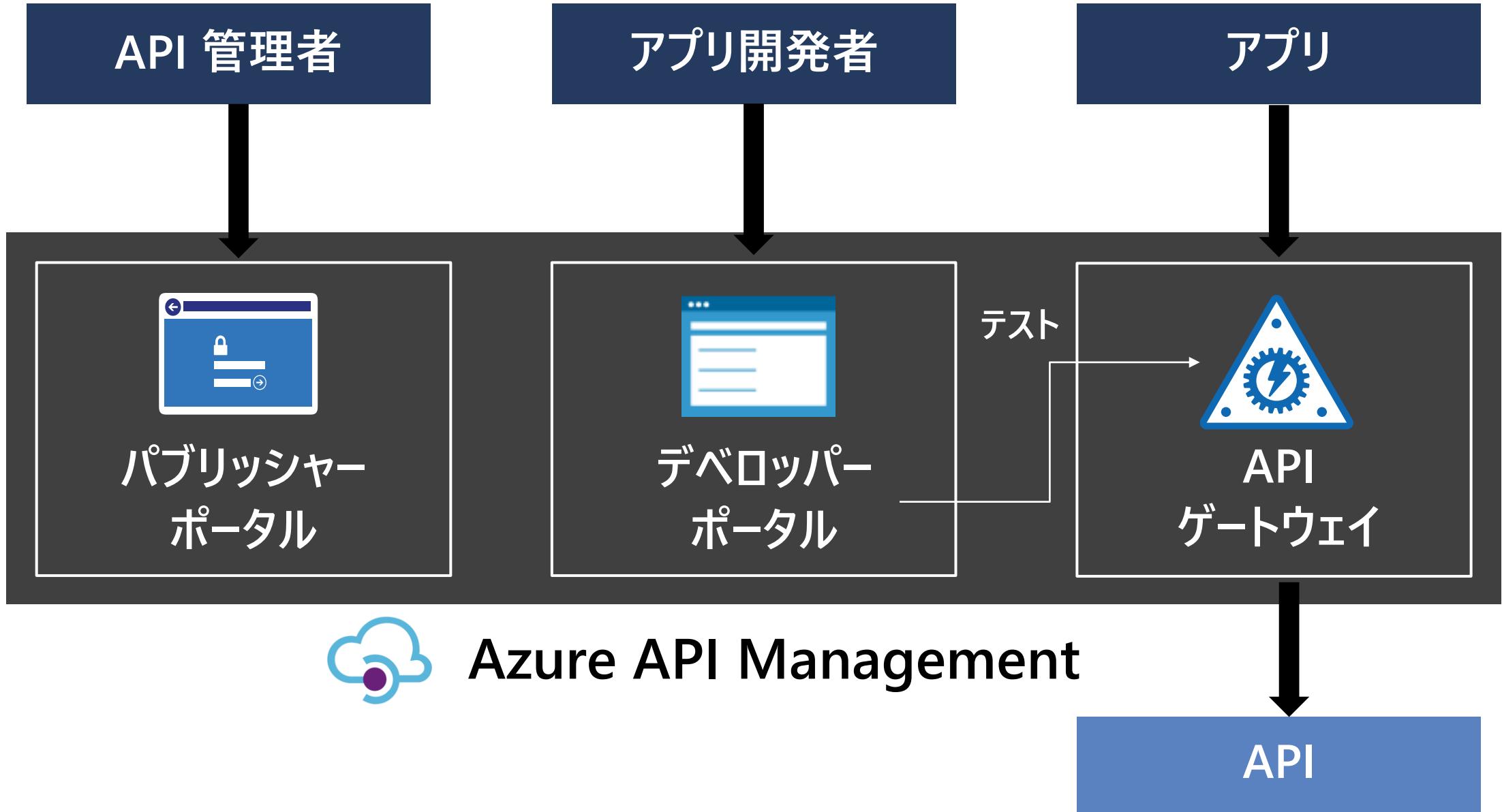
キー管理



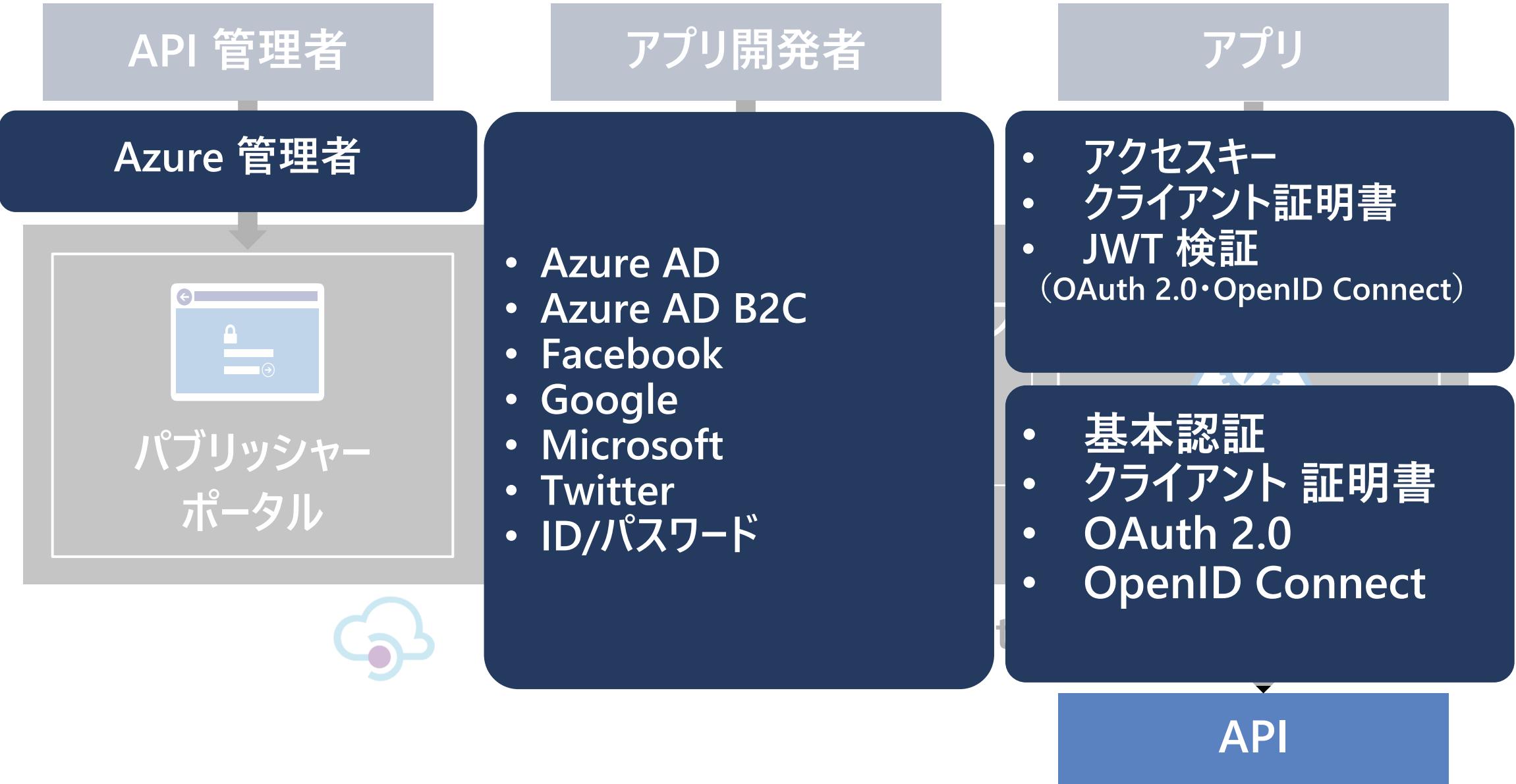
## 開発者ポータルの提供

Name	In	Required	Type	Example

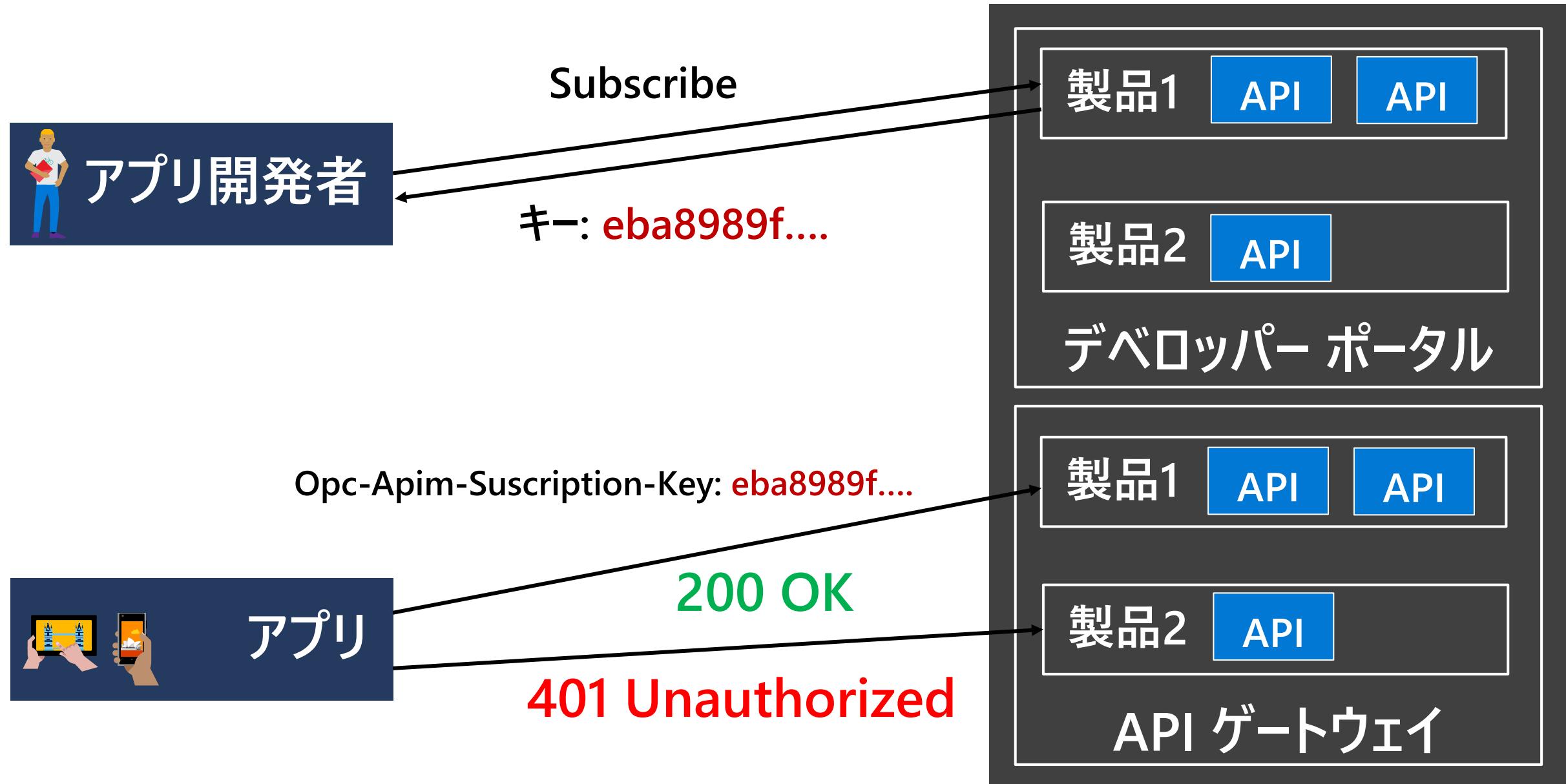
# API Management のアクセス制御



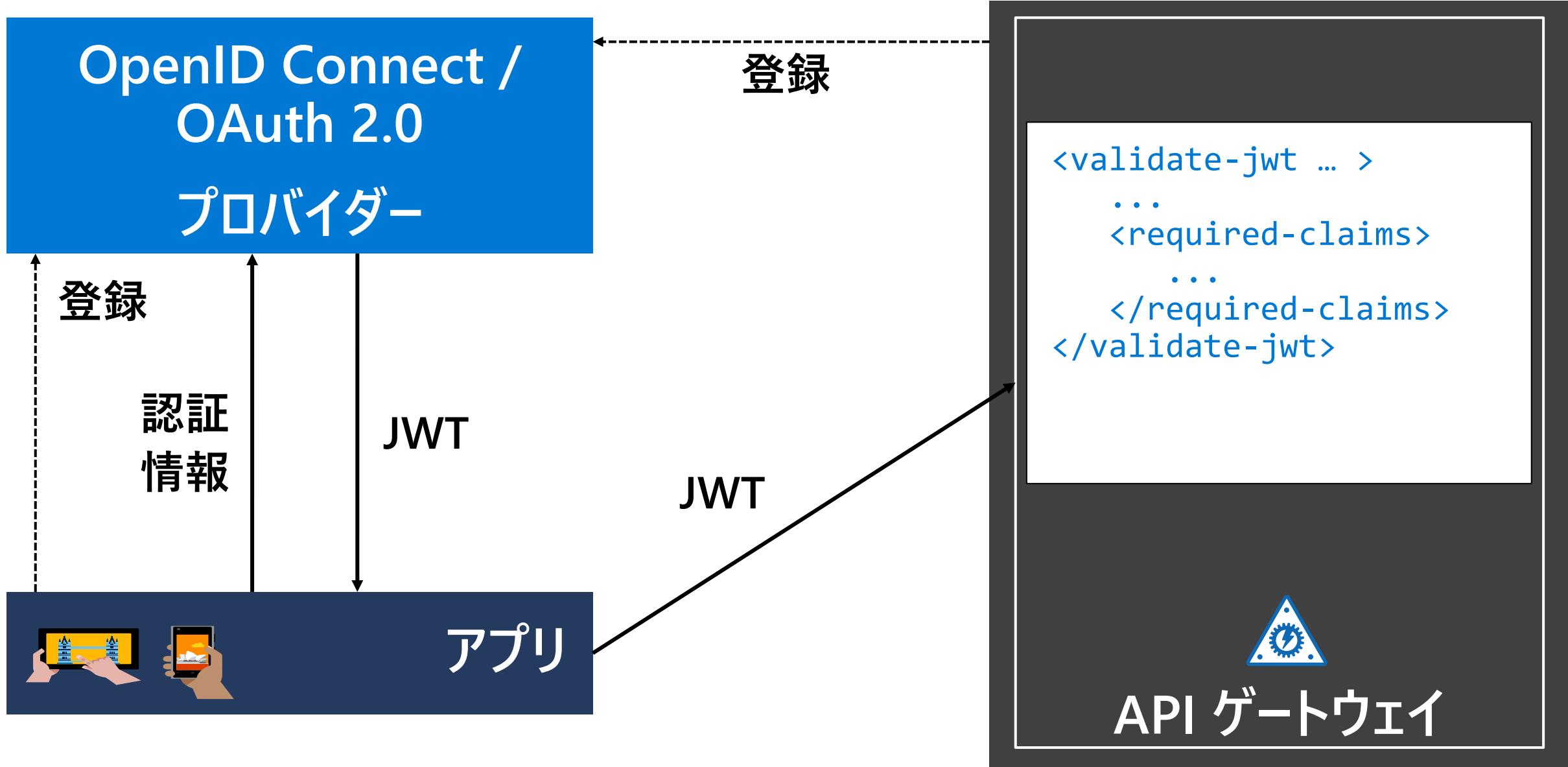
# API Management のアクセス制御



# サブスクリプションキーによる API 認証



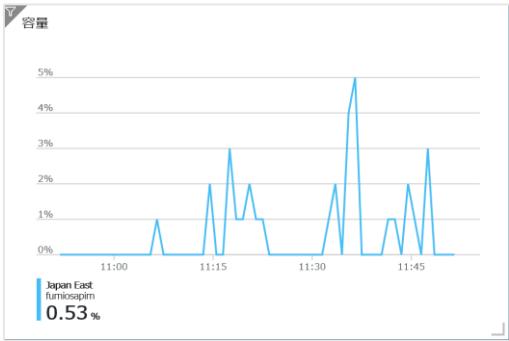
# JWT (JSON Web Token) 検証



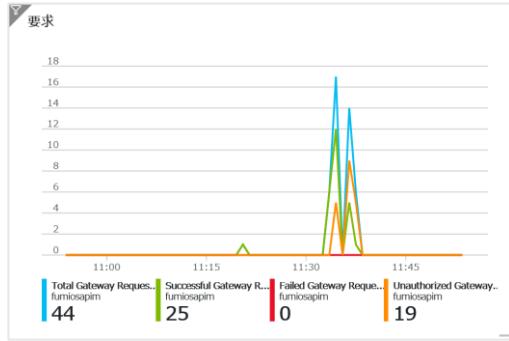
# API の監視

## メトリック

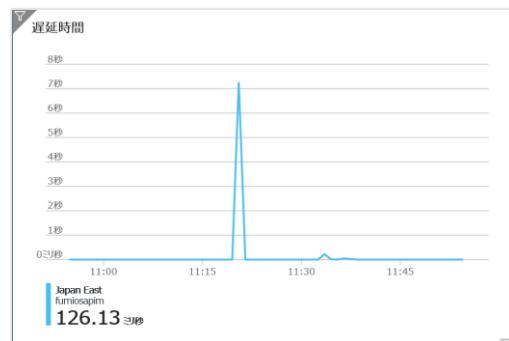
### ・容量



### ・リクエスト数



### ・遅延時間



## Application Insights

- 利用統計情報
- パフォーマンスの調査
- エラーの調査

The screenshot shows the Application Insights interface for the 'fumiosapim Japan East' application. It includes:

- A top navigation bar with '過去 24 時間' (Last 24 hours), 'フィードバック' (Feedback), '詳細情報' (Detailed Information), and a refresh button.
- A 'マップ コンポーネントの更新' (Map Component Update) section showing a network diagram with three components: 'fumiosapim Japan East' (1 instance, 63%, 40ms, 80 calls), 'fumioscalcapci BACKEND' (3 instances, 145ms, 24 calls), and 'echoapi BACKEND' (2 instances, 288ms, 2 calls).
- A 'Analytics 内のビュー' (View in Analytics) section showing '上位の失敗した要求 (名前別)' (Top Failed Requests by Name):

名前	カウント
GET /Calc/api/add	36
GET /echo/resource	8
PUT /echo/resource	5

A 'エラーの調査' (Error Investigation) button is also present.
- A '最も遅い要求 (名前別)' (Slowest Requests by Name) section showing '期間 (平均)' (Average Duration):

名前	期間 (平均)
GET /Calc/api/div	149.3 ミリ秒
GET /echo/resource-cached	115.8 ミリ秒
GET /Calc/api/mul	23.7 ミリ秒

A 'パフォーマンスの調査' (Performance Investigation) button is also present.

# API の統計情報

## 管理ポータル

Developers						
Name	Successful calls	Blocked calls	Failed calls	Other calls	Total calls	Bandwidth
Administrator	43	0	0	3	46	46.0 KB
Fumio Decode	33	1	3	0	37	39.3 KB
Anonymous	0	0	0	0	0	0.0

Products						
Name	Successful calls	Blocked calls	Failed calls	Other calls	Total calls	Response time, Avg
demo	76	1	3	3	83	874 ms
Calculator , Employee Certification,...						
Starter	0	0	0	0	0	0 ms
Echo API						
Unlimited	0	0	0	0	0	0 ms
Echo API						

Subscriptions						
Name	Successful calls	Blocked calls	Failed calls	Other calls	Total calls	Response time, Avg
Administrator / demo	43	0	0	3	46	1,407 ms
596da3d0ec72a90d3cd67...						
Fumio Decode / demo	33	1	3	0	37	180 ms
demo						

## REST API

GET

```
https://management.azure.com/subscriptions/subid/resourceGroups/rg1/providers/Microsoft.ApiManagement/service/apimService1/reports/bySubscription?$filter=timestamp ge datetime'2017-06-01T00:00:00' and timestamp le datetime'2017-06-04T00:00:00'&api-version=2018-06-01-preview
```

## Power BI

