



ThemiStruct
テミストラクト

共通ID基盤、API連携認証システムを 取り巻く技術動向

株式会社オージス総研
事業開発本部 テミストラクトソリューション部

八幡 孝

自己紹介



八幡 孝

株式会社オージス総研

統合認証ソリューション担当

OpenAMコンソーシアム

副会長

OpenIDファウンデーション・ジャパン
Enterprise Identity WG

リーダー

統合認証ソリューションを15年以上やってきました



ThemiStruct-WAM

シングルサインオン
認証基盤ソリューション

ThemiStruct-IDM

ID管理ソリューション

ThemiStruct-CM

電子証明書発行・管理
ソリューション

ワンタイムパスワードソリューション

ThemiStruct-OTP

システム監視ソリューション

ThemiStruct-MONITOR

 デモストラクト ThemiStruct
Identity Platform

APIエコノミー時代の
統合認証パッケージ

認証基盤のユースケースが拡大

ユースケース	狙い・特長
社内システム利用のガバナンス強化	認証処理の一元化、人事システム等と連動したタイムリーなIDメンテナンス。
取引先へのシステム提供	取引先ユーザーの確実な認証。IPアドレスや電子証明書の併用。
クラウドサービス利用時、スマホ・タブレット利用時の認証強化	社外からの利用の制限。社外での利用時の追加の認証の実施。社用端末の識別。 クラウドサービスのIDメンテナンス。
顧客（一般消費者）向けの情報提供、サービス提供	SSOによる顧客への利便性の提供。複数アプリへの展開。収集した属性の活用。他社サービスとの連携。
モバイルアプリ化、オープンAPI活用によるアプリ機能、サービスの高度化	・ アプリ内にパスワード保存不要な安全な認証方式、SSOに対応できる認証方式への対応。 ・ 利用者同意に基づく必要最少権限でのデータ連携を実現する認証・認可方式への対応。

現在の認証基盤の主要なユースケース

企業/企業グループ内の業務向けの「**社内統合認証基盤**」を構築する

顧客向けサービスサイトの「**共通ID基盤**」を構築する

オープンAPIを提供するための「**API連携認証システム**」を構築する

統合認証基盤の重要性が高まる

□ サービス時間帯の拡大

- 平日業務時間内の提供から24時間365日の提供へ
- メンテナンスに利用できる時間帯が大幅に縮小

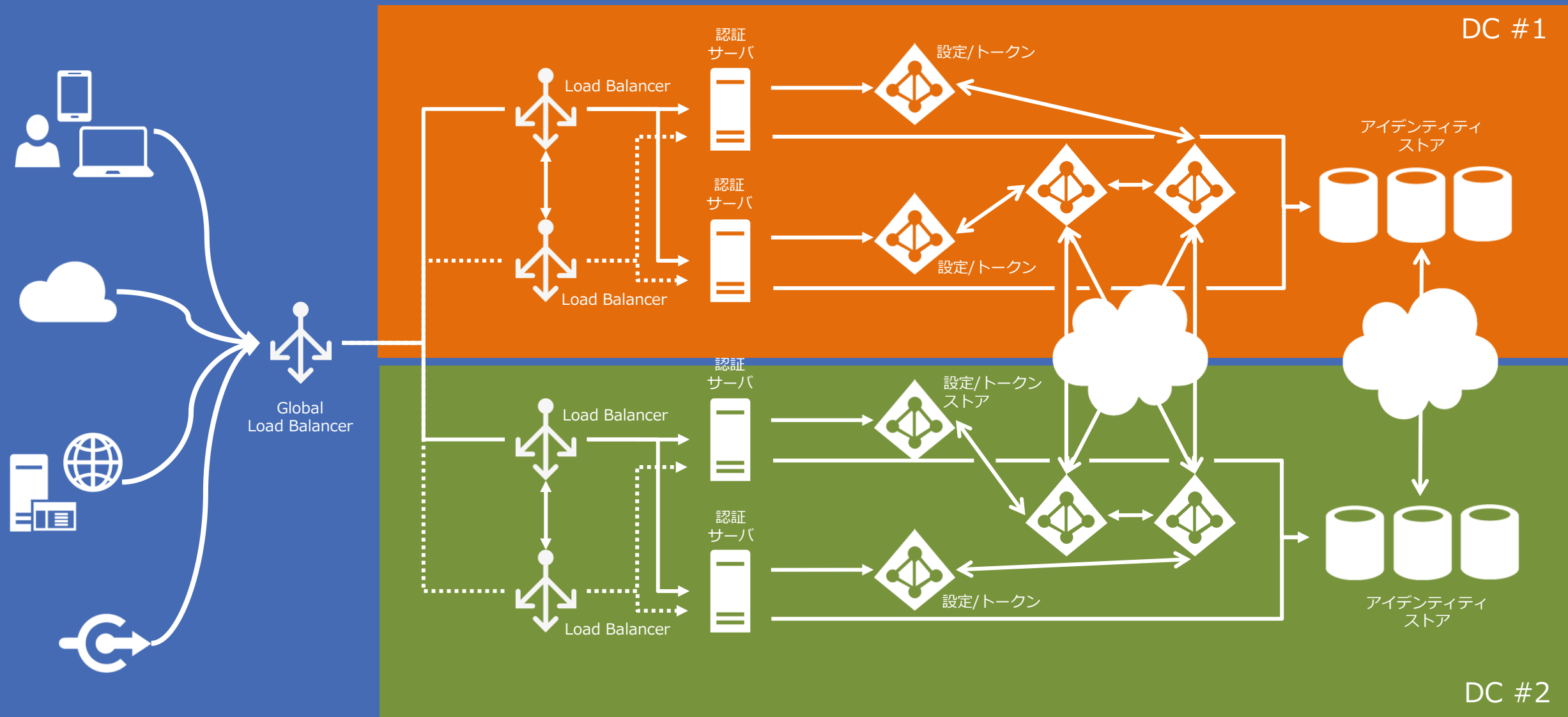
□ 許容停止時間の短縮

- 業務影響を抑えるため障害状態から短時間での回復が必要

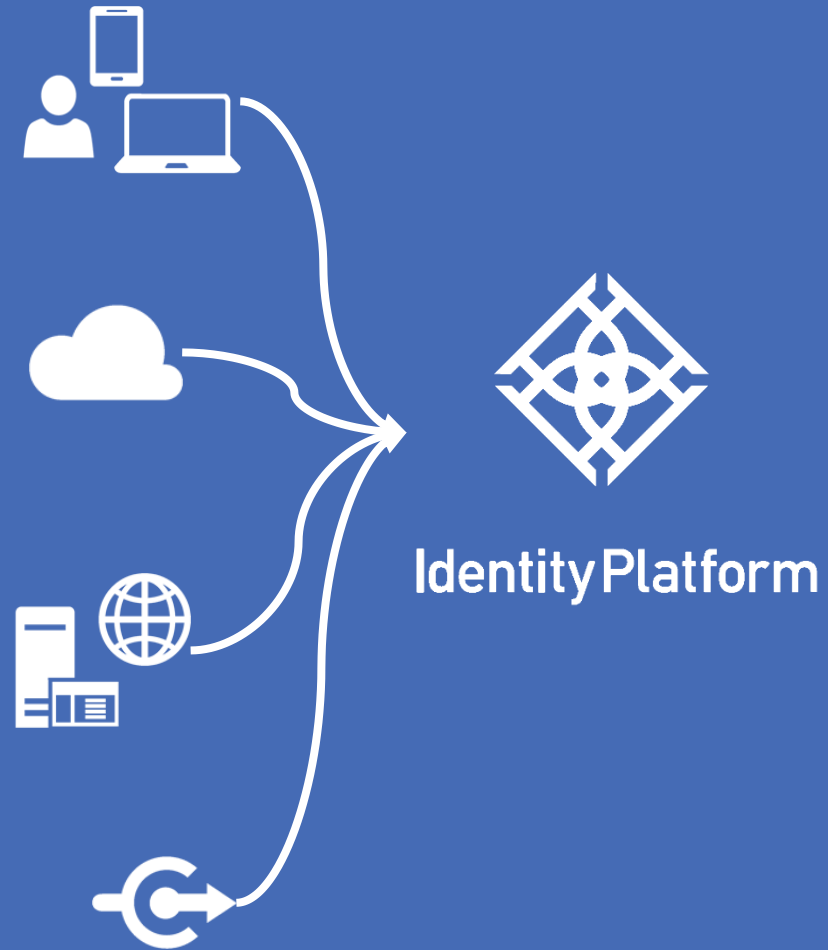
□ アクセス集中時の性能確保

- 出勤時、特定業務の締切日・締切時間、など
- 特定イベント（キャンペーンなど）による一斉アクセス、など

高度な基盤設計が必要に ➡ 入念な可用性、性能のテスト。プロジェクトの巨大化。



これから： 基盤の設計、テストは不要。プロジェクトの迅速なスタートアップ。



ThemiStruct Identity Platform の特長

- Amazon API Gateway, AWS Lambda など、AWSのマネージドサービスのコンポーネントを活用。
- AWSマネージドサービスで提供される可用性、スケーラビリティのメリットを享受。
- ユーザー情報などのデータは、自社VPC内で保持。
- AWSの各サービスコンポーネントの設定、コードの配置を自動化する専用のインストーラーを提供。

現在の認証基盤の主要なユースケース

企業/企業グループ内の業務向けの「**社内統合認証基盤**」を構築する

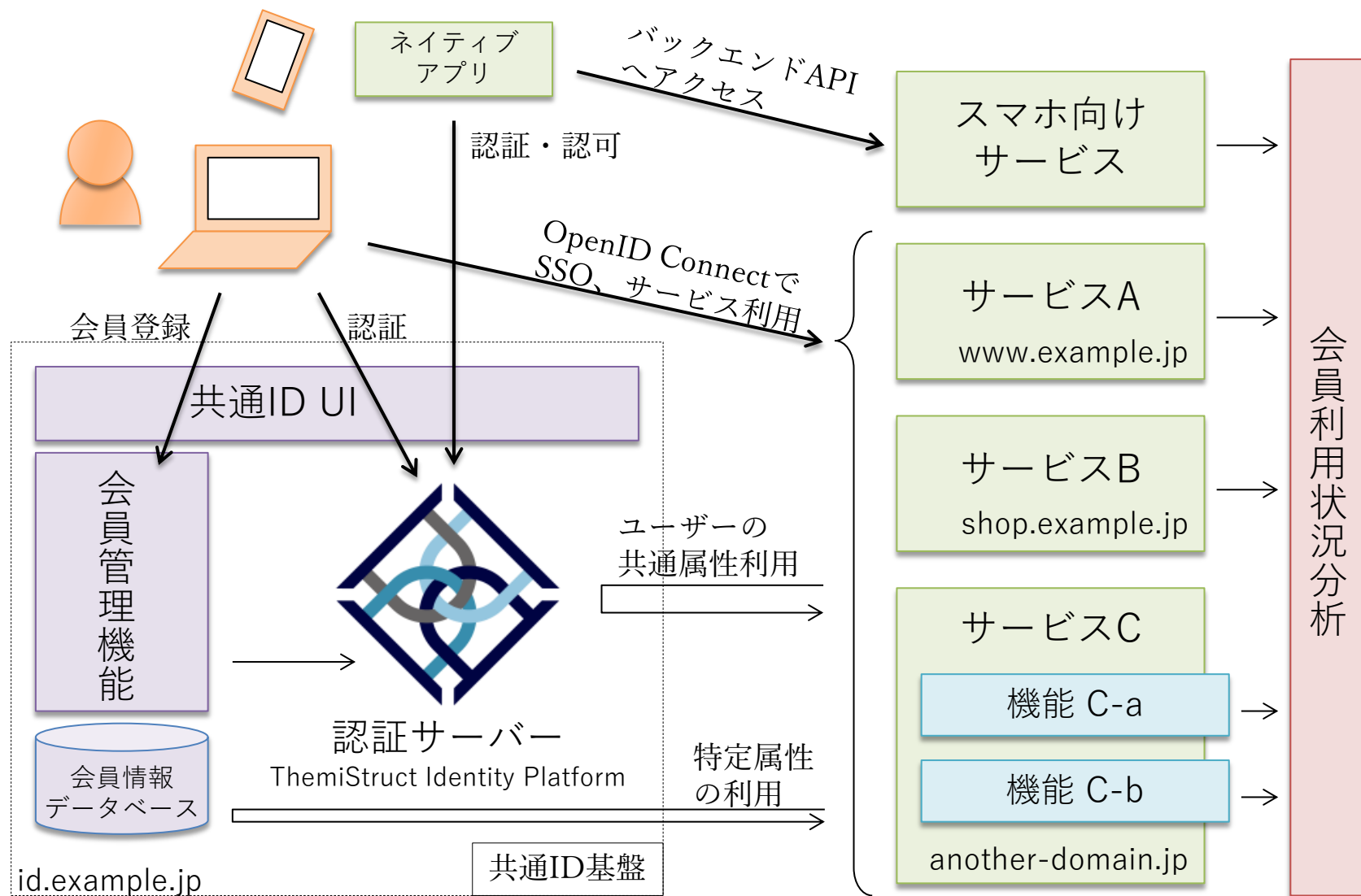
顧客向けサービスサイトの「**共通ID基盤**」を構築する

オープンAPIを提供するための「**API連携認証システム**」を構築する

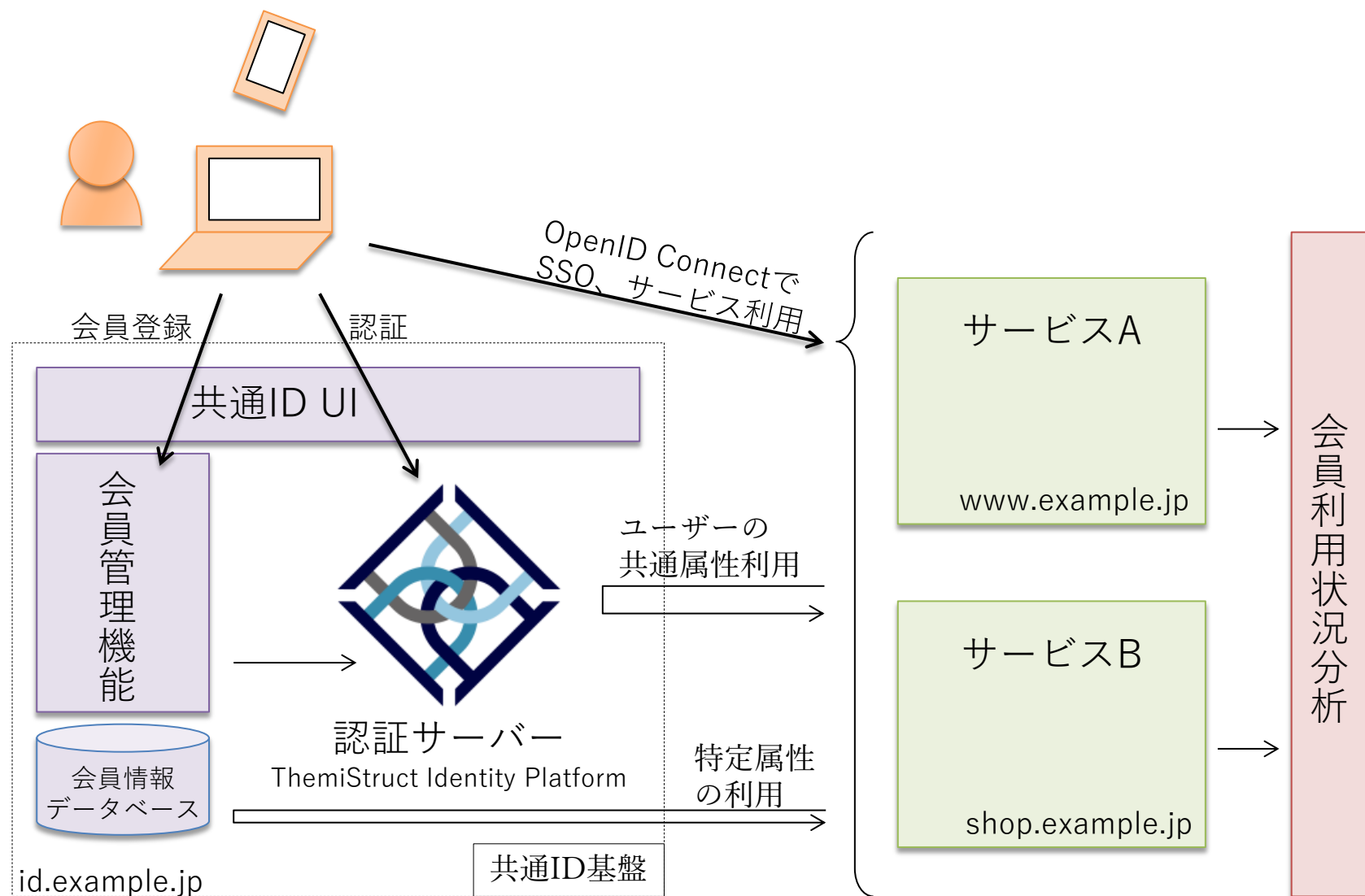
ThemiStruct Identity Platform で
顧客向けサービスサイトの

「共通ID基盤」を構築する

「共通ID基盤」の構築イメージ

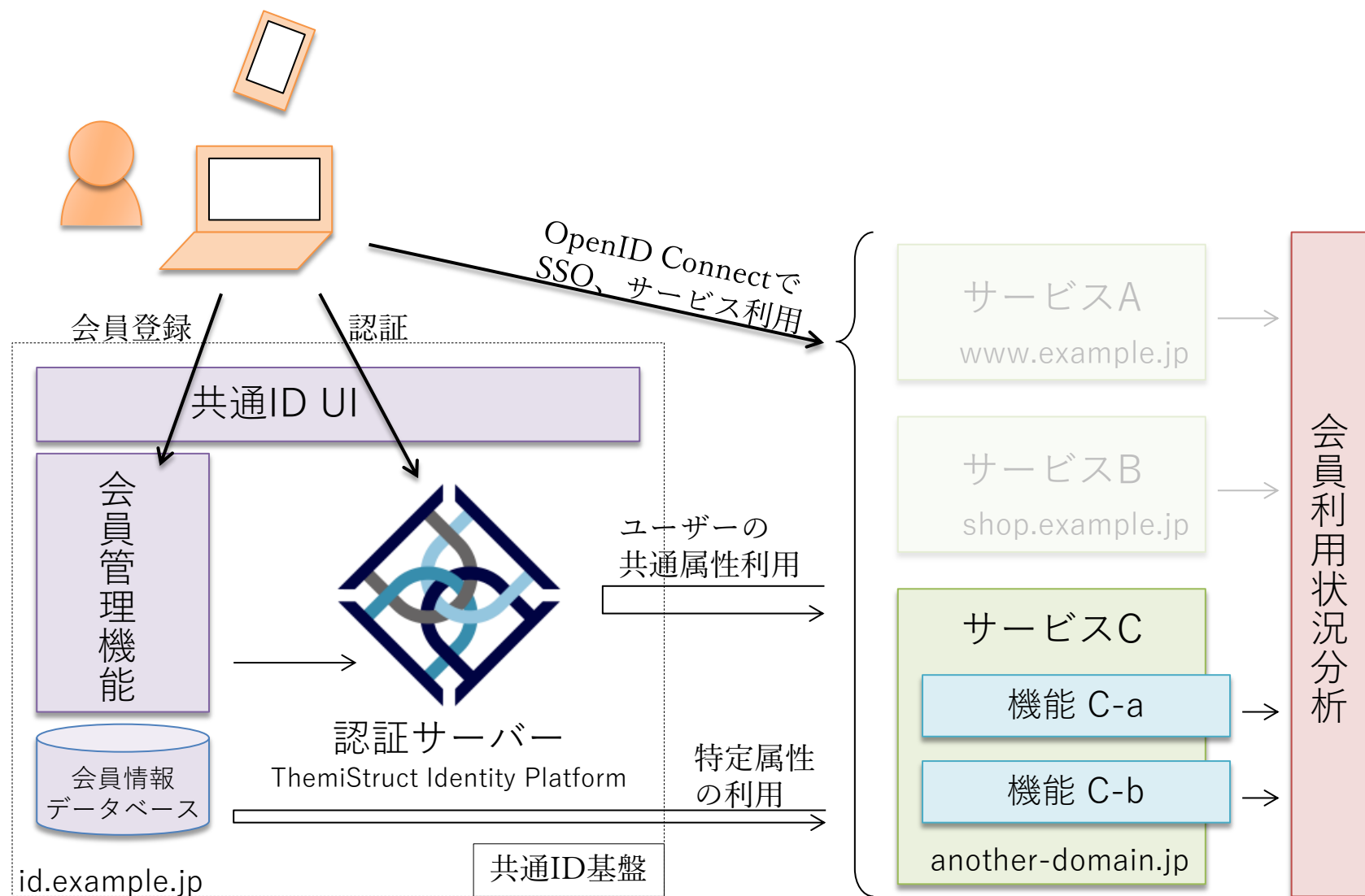


顧客向けサービスのIDを共通化する



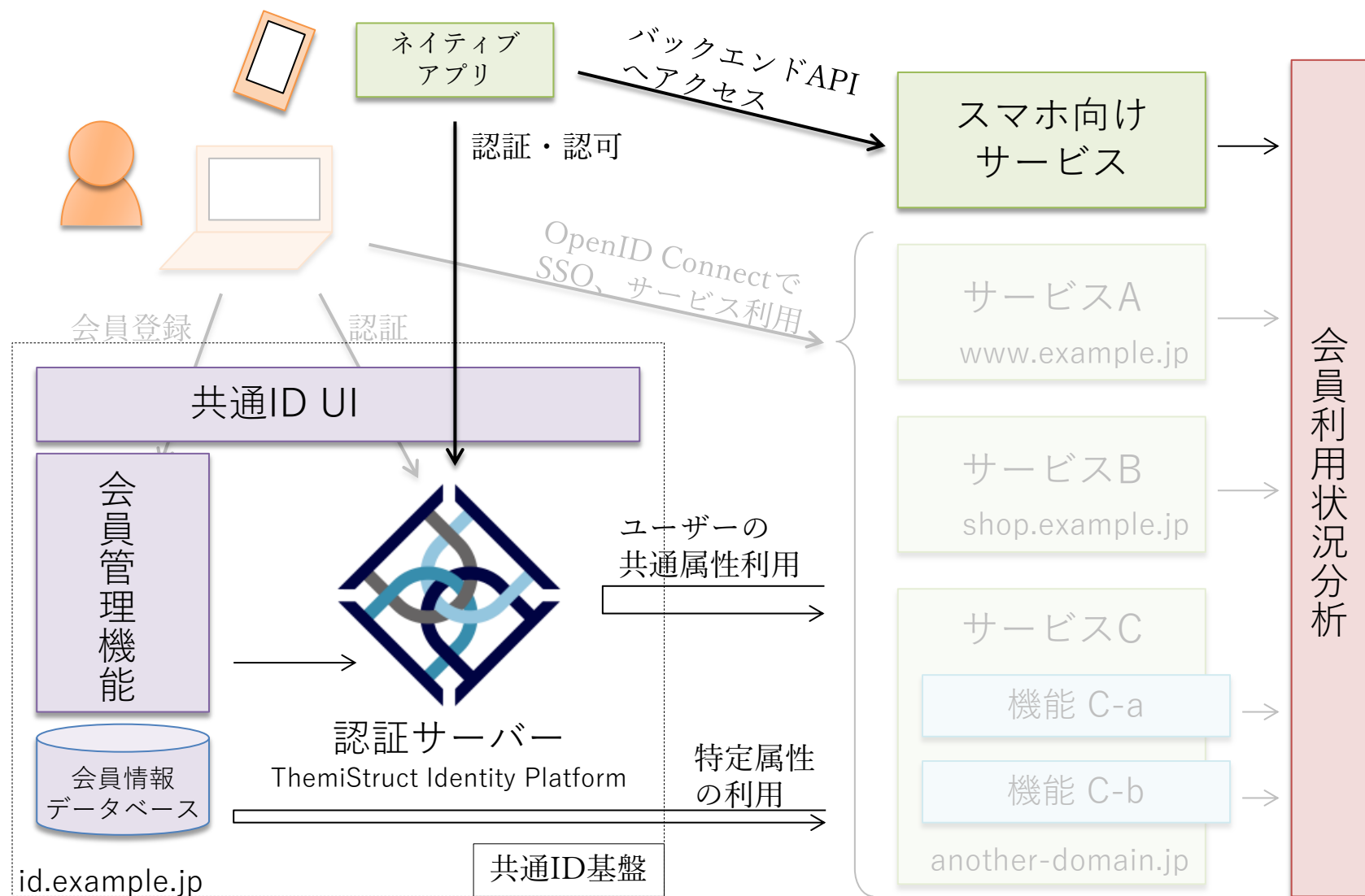
- ✓ 既存サイトに、Themistruct Identity Platform の認証機能を組み込む。
- ✓ 会員管理機能で登録・更新される情報を Themistruct Identity Platform に連携。
- ✓ ユーザーは一回の会員登録、サインインで、全サービスを横断的に利用可能に。
- ✓ 各サービスが共通的に利用する属性は、認証サーバーからサービス利用時に連携することで、会員管理機能への負担を軽減。
- ✓ 全サービスを横断して会員の利用分析が可能に。

提供サービスを追加する、更新する



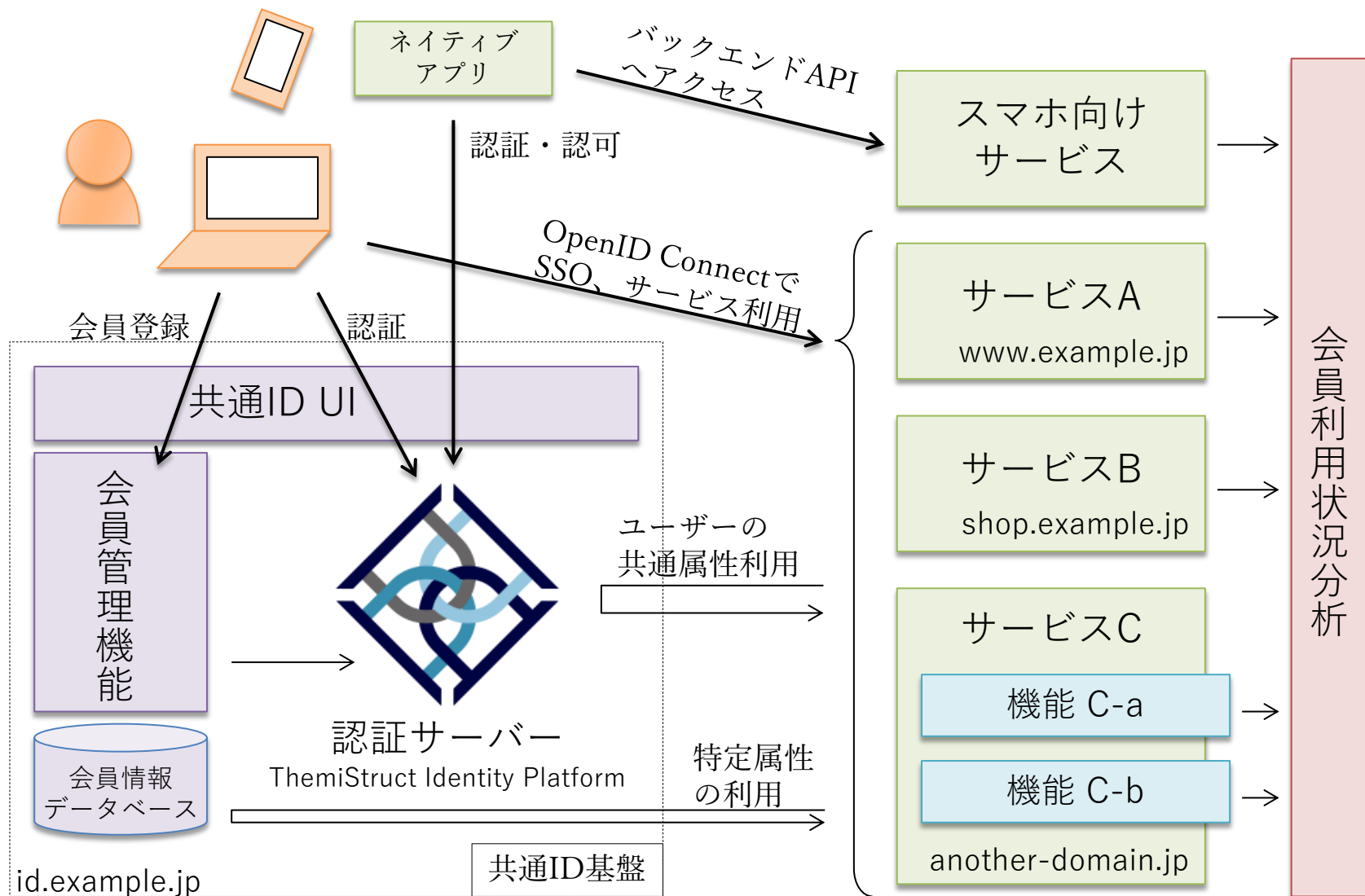
- ✓ ブランド毎に専用ドメインを使い分けるクロスドメイン環境にも対応が可能。
- ✓ 提供するサービスを追加する場合は、共通IDを利用するアプリとして開発するだけでSSO利用が可能に。
- ✓ サービス内の機能を増やすときも、サブシステム毎に開発してSSO接続することで、ユーザーは一つのサービスとして利用可能。
- ✓ サービス、機能の入れ替えが行ないやすい構造に。

スマホ向けサービスを提供する



- ✓ スマホ向けネイティブアプリの提供では、バックエンドAPIアクセスのための認証・認可の処理が必要に。
- ✓ ThemiStruct Identity PlatformのOAuth認可サーバーの機能でネイティブアプリの利用にも対応。

スケーラブル、ハイアベイラブルな共通ID基盤に



- ✓ AWSマネージドサービスにより、アクセスの繁閑にも柔軟に対応が可能。
- ✓ 無停止アップデート機能により、利用者にサービス提供を続けながらThemiStruct Identity Platformのソフトウェアを更新可能。

ThemiStruct Identity Platform で
オープンAPIを提供するための
「API連携認証システム」を構築する

Web API を公開する際の課題

□ クローズドAPI

- 組織内での Web API の利用
- 組織の境界内からのみアクセス可能



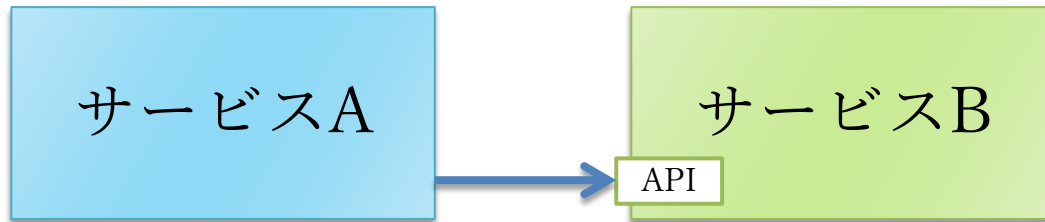
□ オープンAPI

- 組織の境界の外側の第三者への Web API の公開
- Web API 利用者の認証、機能やデータへのアクセス権管理が不可欠

Web API 利用の形態

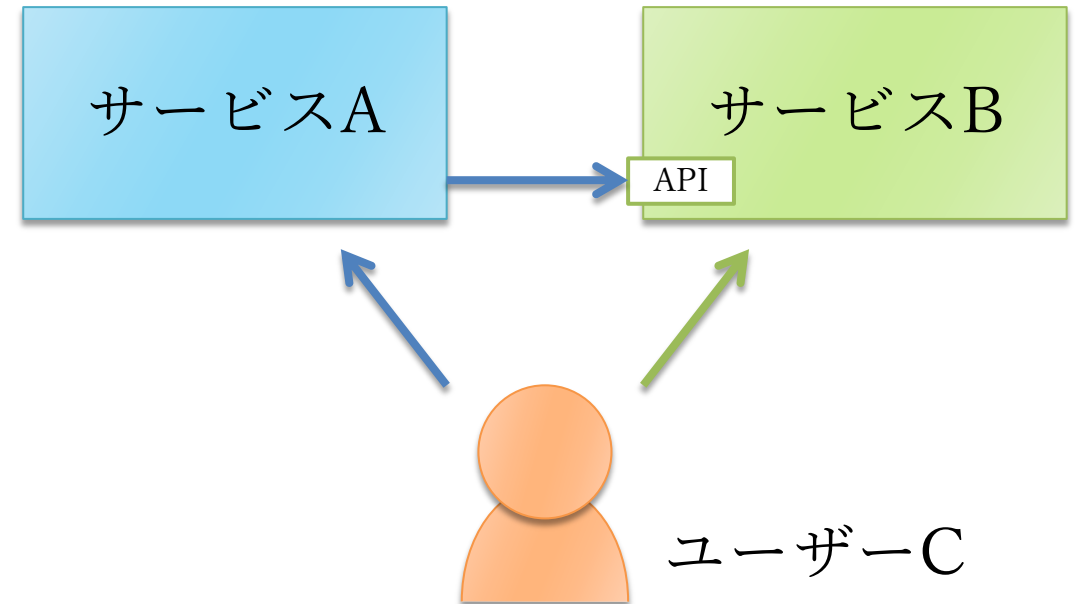
2者間でのAPI利用

- サービスA が サービスB の機能を利用するために API にアクセスする。
- サービスB は サービスA 向けの機能・データを提供する。



3者間でのAPI利用

- ユーザーC は サービスA、サービスB の利用者である。
- サービスA は、ユーザーCの意図により、ユーザーC に代わり サービスB の API にアクセスする。
- サービスB は ユーザーC 向けの機能・データを提供する。



2者間でのAPI利用時の認証

□ API のアクセス元を認証する。

- APIキー
- BASIC認証 (IDとパスワード)
- クライアント証明書
- OAuth 2.0 Client Credentials Grant
- ...

□ アクセス元が認証できれば、API はアクセス権を判断できる。

3者間でのAPI利用時の認証

□ ユーザーC は サービスA に、アクセスできるデーターや操作を限定して、サービスB の API へアクセスさせたい。

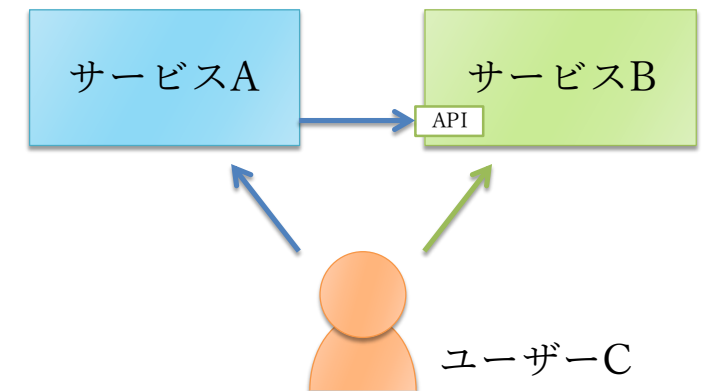
➤ ユーザーIDとパスワードなど、クレデンシャル情報を渡す方式では、全権限をサービスAに与えてしまう。

□ 現時点では OAuth 2.0 の利用が唯一の選択肢

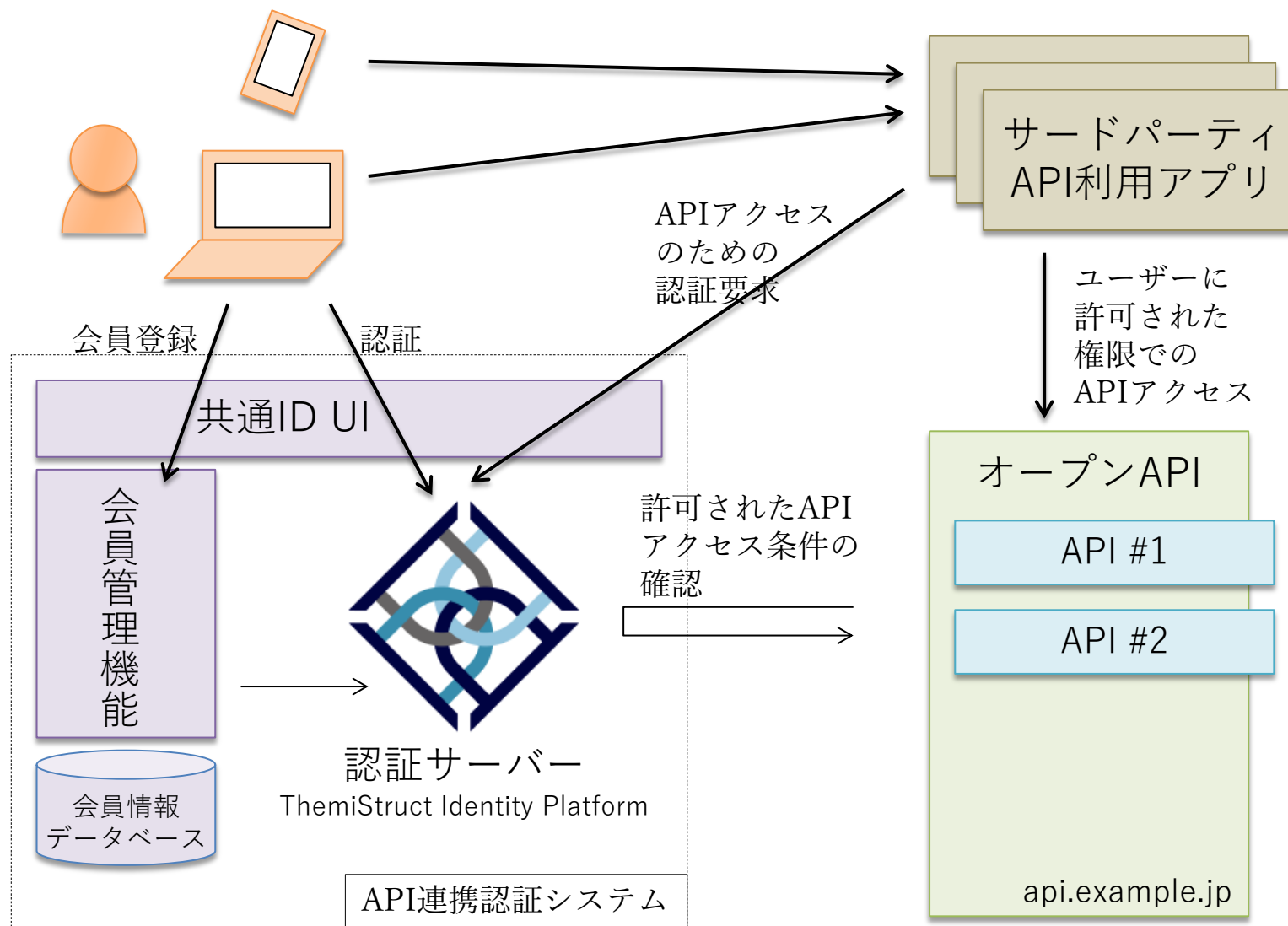
➤ ユーザーC の同意に基づき、ユーザーC が意図した範囲の限定された権限で API へのアクセスを許可する方式。

➤ OAuth 2.0 Authorization Code Grant

➤ OAuth 2.0 Implicit Grant



「API連携認証システム」の構築イメージ



対応すべき OAuth 2.0 仕様の概略 (最小限)

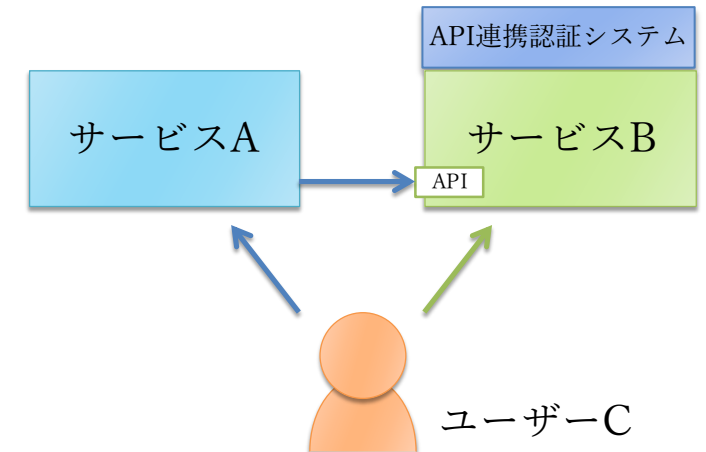
サービス開発者、API開発者も対応が必要

□ オープンAPIの提供者（下図サービスB）

- API連携認証システム（Authorization Server）実装者
- API開発者

□ オープンAPIの利用者（下図サービスA）

- アプリケーション開発者
 - フロントエンド開発者
 - バックエンド開発者



対応が必要となる OAuth の仕様類

#	仕様	AS 実装者	API 開発者	API利用者 アプリ開発者
①	The OAuth 2.0 Authorization Framework [RFC 6749]	○	—	○
.1	Authorization Code Grant	○	—	○
.2	Implicit Grant	○	—	○
.3	Client Credentials Grant	○	—	○
②	The OAuth 2.0 Authorization Framework: Bearer Token Usage [RFC 6750]	—	○	○
③	OAuth 2.0 Token Revocation [RFC 7009]	○	○	○
④	Proof Key for Code Exchange by OAuth Public Clients [RFC 7636]	○	—	○
⑤	OAuth 2.0 Token Introspection [RFC 7662]	○	○	—
⑥	OAuth 2.0 for Native Apps [RFC 8252]	—	—	○

API利用時の構成パターン

□ バックエンドあり (Confidential Client)



□ バックエンドなし (Public Client)



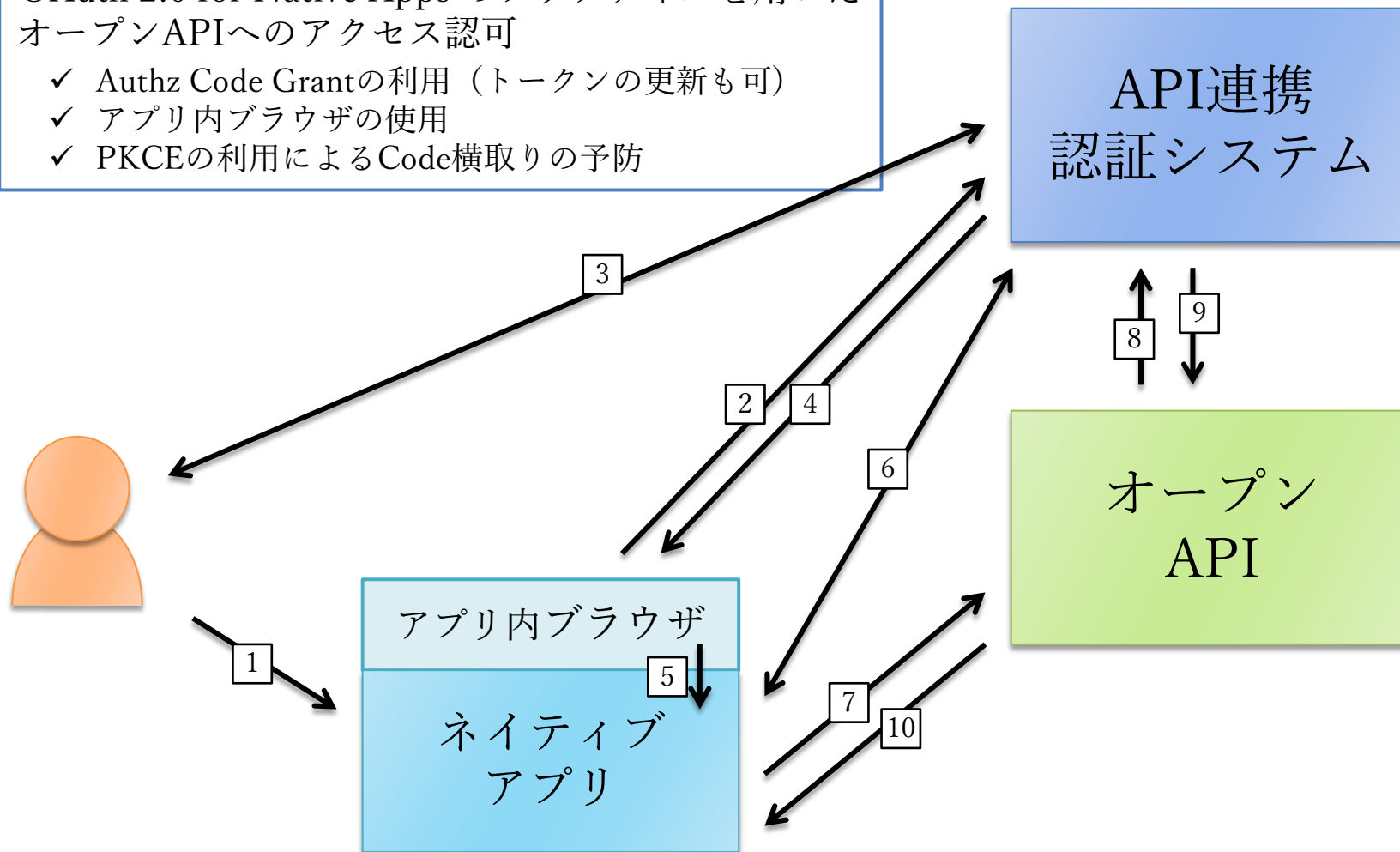
アプリ形態別、対応すべき OAuth の方式

#	ユーザーへの提供形態	バック エンド	OAuthの方式	ポイント
①	ブラウザで動作する JavaScriptアプリ (モバイルウェブ、SPA、など)	あり	Authz Code Grant	バックエンドがオープンAPIへアクセス。 バックグラウンドでのAPIアクセスのため、 トークン自動更新が必要な場合も。
②		なし	Implicit Grant	JSアプリがオープンAPIへアクセス。 トークン更新が必要な場合はブラウザを 介して行なえる。
③	ネイティブアプリ	あり	Authz Code Grant	(① と同じ)
④		なし	Authz Code Grant w/ PKCE	ネイティブアプリがオープンAPIへアクセス。 バックグラウンドでのAPIアクセスの ため、トークン自動更新が必要な場合も。
⑤	Webアプリ	あり (Webアプリ サーバー)	Authz Code Grant	Webアプリケーションサーバーがオープ ンAPIへアクセス。バックグラウンドでの APIアクセスのため、トークン自動更新が 必要な場合も。

ネイティブアプリでのAPIアクセス認可

OAuth 2.0 for Native Apps のプラクティスを用いたオープンAPIへのアクセス認可

- ✓ Authz Code Grantの利用（トークンの更新も可）
- ✓ アプリ内ブラウザの使用
- ✓ PKCEの利用によるCode横取りの予防



1. ネイティブアプリを起動、ログイン開始
2. アプリ内ブラウザを使いOAuth認可を要求（PKCEを併用）
3. ユーザー認証を実行
4. アクセストークンを取得するためのCodeを返却
5. アプリ内ブラウザからネイティブアプリにCodeを返却
6. Codeをアクセストークンに交換（PKCEを併用）
7. アクセストークンをつけてオープンAPIにアクセス
- 8.~9. アクセストークンの情報を確認（ユーザー、Scope、有効期限、など）
10. オープンAPIが情報を使ってネイティブアプリが動作

オープンAPIの振る舞いは、JSアプリ、ネイティブアプリで共通。

今後対応が必要となってくる仕様 (OAuth 2.0, OpenID Connect 関連)

金融API向けにOAuth実装仕様の策定が進行中

□ OpenID Foundation (Global) の Financial API WG が策定

➤ <https://openid.net/wg/fapi/>

□ OAuthの実装仕様について、以下の2つがすでに Implementer's Draft となり、公開されている。

➤ Part 1: Read Only API Security Profile (Implementer's Draft).

➤ Part 2: Read & Write API Security Profile (Implementer's Draft).

□ 以下の仕様についても検討が進行中。

➤ Client Initiated Backchannel Authentication Profile.

Part 1: Read Only API Security Profile ①

#	Authorization Server への要求事項	特に注目したい箇所
1	コンフィデンシャルクライアントをサポートすることが望ましい。(SHOULD)	—
2	パブリッククライアントをサポートすることが望ましい。(SHOULD)	—
3	(署名や暗号化に) 対象鍵を使う場合は [OIDC] の 16.19節 の要件に準拠した client_secret を提供する。(SHALL)	—
4	トークンエンドポイントでは以下のいずれかの方法を用いてコンフィデンシャルクライアントを認証する。(SHALL) 1. TLS相互認証 [MTLS] 2. [OIDC] の 9章 で指定されている client_secret もしくは秘密鍵を用いた JWS クライアントアサーション	□
5	クライアント認証にRSA暗号アルゴリズムを用いる場合は 2048ビット 以上の長さの鍵を用いる。(SHALL)	—
6	クライアント認証に楕円曲線暗号アルゴリズムを用いる場合は 160ビット 以上の長さの鍵を用いる。(SHALL)	—
7	パブリッククライアントをサポートする場合は、[RFC7636] の S256コードチャレンジ方式 をサポートする。(SHALL)	—
8	リダイレクトURIは事前登録する。(SHALL)	—
9	認可要求のパラメータに redirect_uri を必要とする。(SHALL)	—
10	(認可要求の) redirect_uri の値は、事前に登録されたリダイレクトURIの一つに完全一致しなければならない。(SHALL)	—
11	[X.1254] に定義された LoA 2 以上のユーザー認証を要求する。(SHALL)	□
12	以前に承認されていない scope の要求を承認する場合には、ユーザーの明示的な同意を必要とする。(SHALL)	—

□ OAuth と OpenID Connect を併用 (OIDCの仕様に基づきOAuthの認可要求を行なう)

□ Confidential Client の認証をより強度が高い方式の利用が指定される

➤ 事実上の OAuth 2.0 の追加仕様 RFC 7521, RFC 7523 への対応

□ アイデンティティ情報更新時の再認証など信頼性を確保する仕組みの導入

Part 1: Read Only API Security Profile ②

#	Authorization Server への要求事項	特に注目したい箇所
13	可能であれば、認可コード ([RFC6749] 1.3.1節) が以前に使用されていないことを検証する。(SHALL)	—
14	[RFC6749] 4.1.4節に沿ったトークンレスポンスを返す。(SHALL)	—
15	発行されたアクセストークンとともに許可されたスコープのリストを返す。(SHALL)	□
16	[RFC6819] 5.1.4.2.2節に定義されているように最少128ビットの不透明な推測できないアクセストークンを発行する。(SHALL)	—
17	[OIDC] 16.18節にあるとおり認可中に長期間の許可を与えようとしていることユーザーに明示することが望ましい。(SHOULD)	□
18	[OIDC] 16.18節にあるとおりエンドユーザーがクライアントに付与されたアクセストークンやリフレッシュトークンを失効する仕組みを提供することが望ましい。(SHOULD)	□
19	[OIDC] 3.1.2.1節にあるとおりの認証要求をサポートする。(SHALL)	—
20	[OIDC] 3.1.2.2節にあるとおりの認証要求の検証を行なう。(SHALL)	—
21	[OIDC] 3.1.2.2節および3.1.2.3節にあるとおりユーザーを認証する。(SHALL)	—
22	[OIDC] 3.1.2.4節および3.1.2.5節にあるとおり認証結果に応じて認証応答を行なう。(SHALL)	—
23	[OIDC] 3.1.3.2節にあるとおりトークン要求を検証する。(SHALL)	—
24	要求された scope に openid が含まれている場合は、[OIDC] 3.1.3.3節にあるとおり、トークン応答の中で認証されたユーザーに対応するsub値を持つ（オプションでacr値も含めた）IDトークンを発行して応答する。(SHALL)	—

- ❑ OAuth, OpenID Connect を併用（OIDCの仕様にに基づきOAuthの認可要求を行なう）
- ❑ 長時間有効な認可（Access Token, Refresh Token発行）には明示的な同意が必要に。
- ❑ 発行された Access Token の scope を明示的に応答することが必要に。

Part 2: Read & Write API Security Profile

#	Authorization Server への要求事項	特に注目したい箇所
1	[OIDC] 6章にあるとおり、request あるいは request_uri パラメータにJWS署名のJWTを使用する。(SHALL)	☐
2	response_type の値には、code id_token もしくは code id_token token を使用する。(SHALL)	☐
3	認可リクエストに対するレスポンスの分離署名として IDトークンを応答する。(SHALL)	—
4	state値を保護するため、IDトークンにはstateのハッシュ値 s_hash を含める。(SHALL)	☐
5	書き込み操作の用途の場合は、記名式 (Holder of Key) の認可コード、アクセストークン、リフレッシュトークンを発行する。(SHALL)	☐
6	記名式 (Holder of Key) の仕組みには [OAUTB] あるいは [MTLS] を用いる。(SHALL)	☐
7	[X.1254] に定義された LoA 3 以上のユーザー認証を要求する。(SHALL)	☐
8	署名付きのIDトークンをサポートする。(SHALL)	—
9	署名付きかつ暗号化されたIDトークンをサポートすることが望ましい。(SHOULD)	☐

❑ OpenID Connect のより深いレベルの仕様の実装

➤ JWTリクエストパラメーター、JWSとJWEによる署名と暗号化されたIDトークン

❑ IDトークンを使った認可レスポンスへの署名 (at_hash, c_hash, s_hashへの対応)

❑ 記名式トークン、記名式認可コード発行への対応

❑ OAuth の策定中仕様 (OAUTB, MTLS) の実装

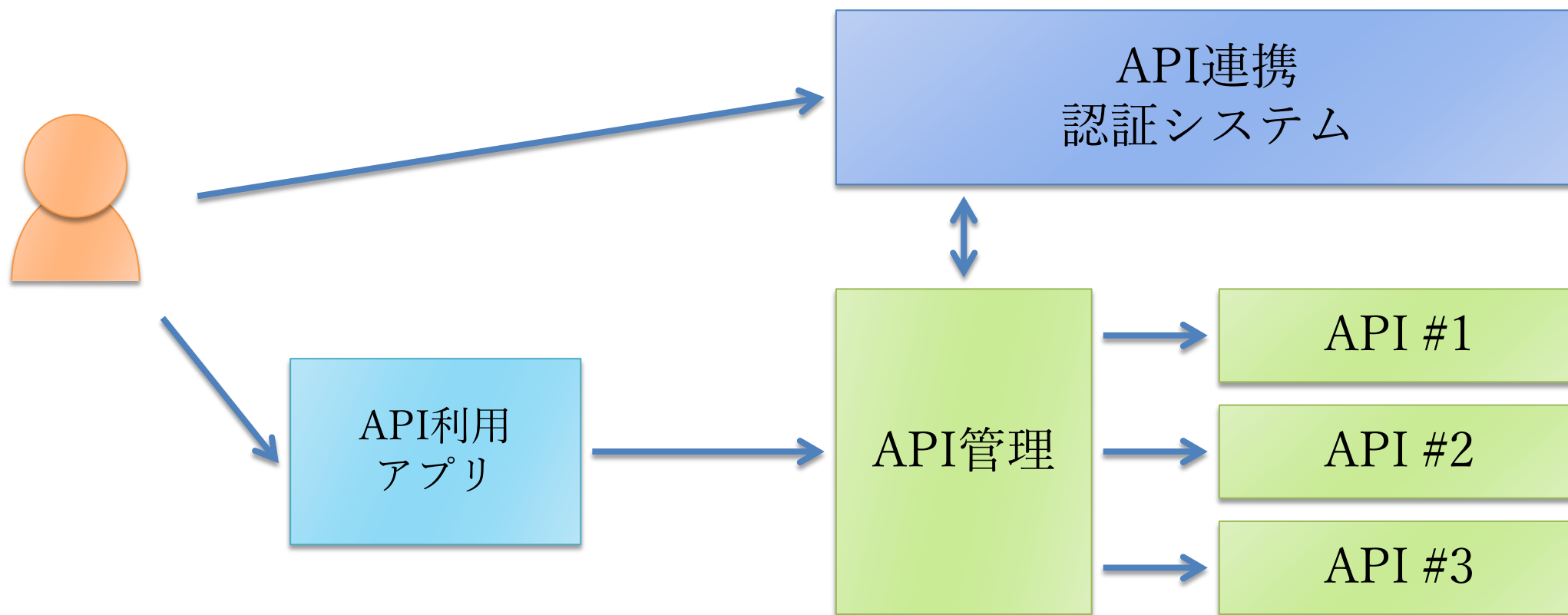
❑ 多要素認証への対応

現時点で対応を検討しておくべき仕様

#	仕様	AS 実装者	API 開発者	API利用者 アプリ開発者
①	OpenID Connect Core 1.0	○	—	○
.1	[OAUTB], [MTLS] を使った記名式トークン、認可コード発行への対応	○	○	○
.2	署名、暗号化されたIDトークンのサポート [Sec.2][Sec.16.14]	○	○	○
.3	長期間有効なトークン発行時の適切な同意画面の表示	○	—	—
②	OAuth 2.0 トークン発行時の scope クレームの応答への対応	○	—	○
③	Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants [RFC 7521]	○	—	○
④	JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants [RFC 7523]	○	—	○
⑤	OAuth 2.0 Token Binding [OAUTB] [draft-ietf-oauth-token-binding-05]	○	○	○
⑥	OAuth 2.0 Mutual TLS Client Authentication and Certificate Bound Access Tokens [MTLS] [draft-ietf-oauth-mtls-07]	○	○	○

API管理のソリューション の位置づけはより重要に

- 策定される新しい仕様への追従を個々の API で進めるのは困難
- API管理ソリューションを使った集中管理が必要になっていく



金融APIでなければ、金融向けのグレードである必要はない

□ FAPI の Profile は金融API向け

□ APIが提供する機能、情報のリスクを評価してどこまで対応するかを決める

□ 対応策は、FAPI Profile が参考になる

R/O Prof.
LoA 2

R/W Prof.
LoA 3

不便, 苦痛, または社会的地位やレピュテーションの毀損

経済的損失または機関の負債

組織や組織のプログラム、公共の利益への損害

許可されていないセンシティブ情報の公開

個人の安全

民事または刑事上の違反

Table 6-2 – Potential impact at each level of assurance

Possible consequences of authentication failure	Potential impact of authentication failure by LoA			
	1	2	3	4
Inconvenience, distress or damage to standing or reputation	Min*	Mod	Sub	High
Financial loss or agency liability	Min	Mod	Sub	High
Harm to the organization, its programs or public interests	N/A	Min	Mod	High
Unauthorized release of sensitive information	N/A	Mod	Sub	High
Personal safety	N/A	N/A	Min Mod	Sub High
Civil or criminal violations	N/A	Min	Sub	High

* Min=Minimum; Mod=Moderate; Sub=Substantial; High=High.

引用元: ITU-T Recommendation X.1254 Entity authentication assurance framework, p7 (2012年9月)

当社ソリューションのご紹介

統合認証ソリューション ThemisStruct を提供しています



ThemisStruct-WAM

シングルサインオン
認証基盤ソリューション

ThemisStruct-IDM

ID管理ソリューション

ThemisStruct-CM

電子証明書発行・管理
ソリューション

ワンタイムパスワードソリューション

ThemisStruct-OTP

システム監視ソリューション

ThemisStruct-MONITOR

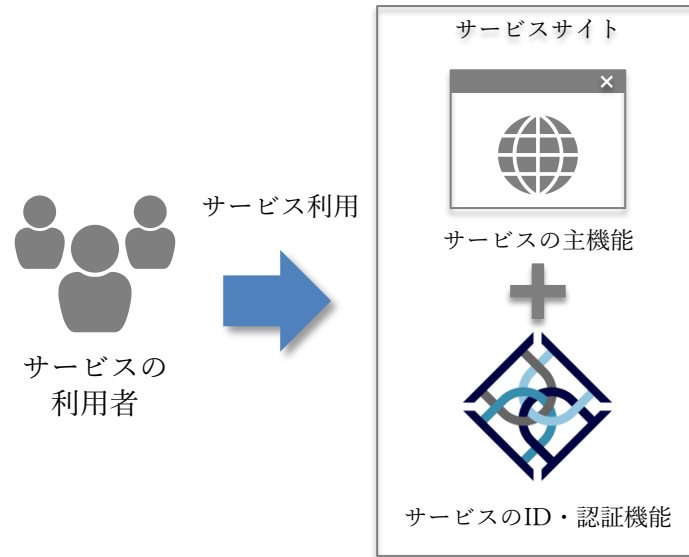
 ThemisStruct デモストラクト
Identity Platform

APIエコノミー時代の
統合認証パッケージ

統合認証パッケージ Themistruct Identity Platform

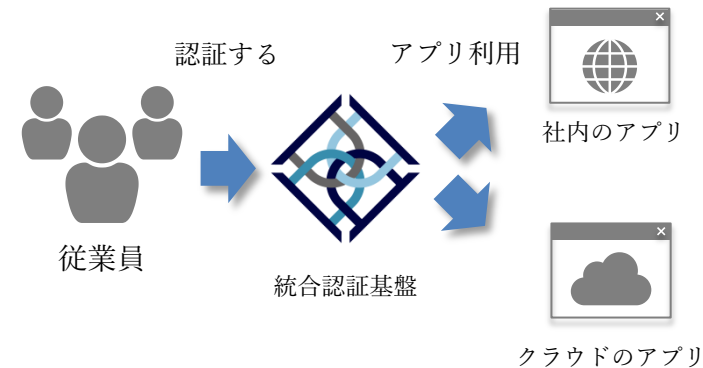
Themistruct Identity Platform は、ITシステム利用者のアイデンティティ管理と認証の機能を提供します。顧客向けにサービスを展開したり、従業員向けにアプリケーションを展開する際に必要となる、共通ID基盤の構築に活用いただけます。

顧客向けサイト領域に活用



サービスサイトの
ID、認証の共通機能として利用

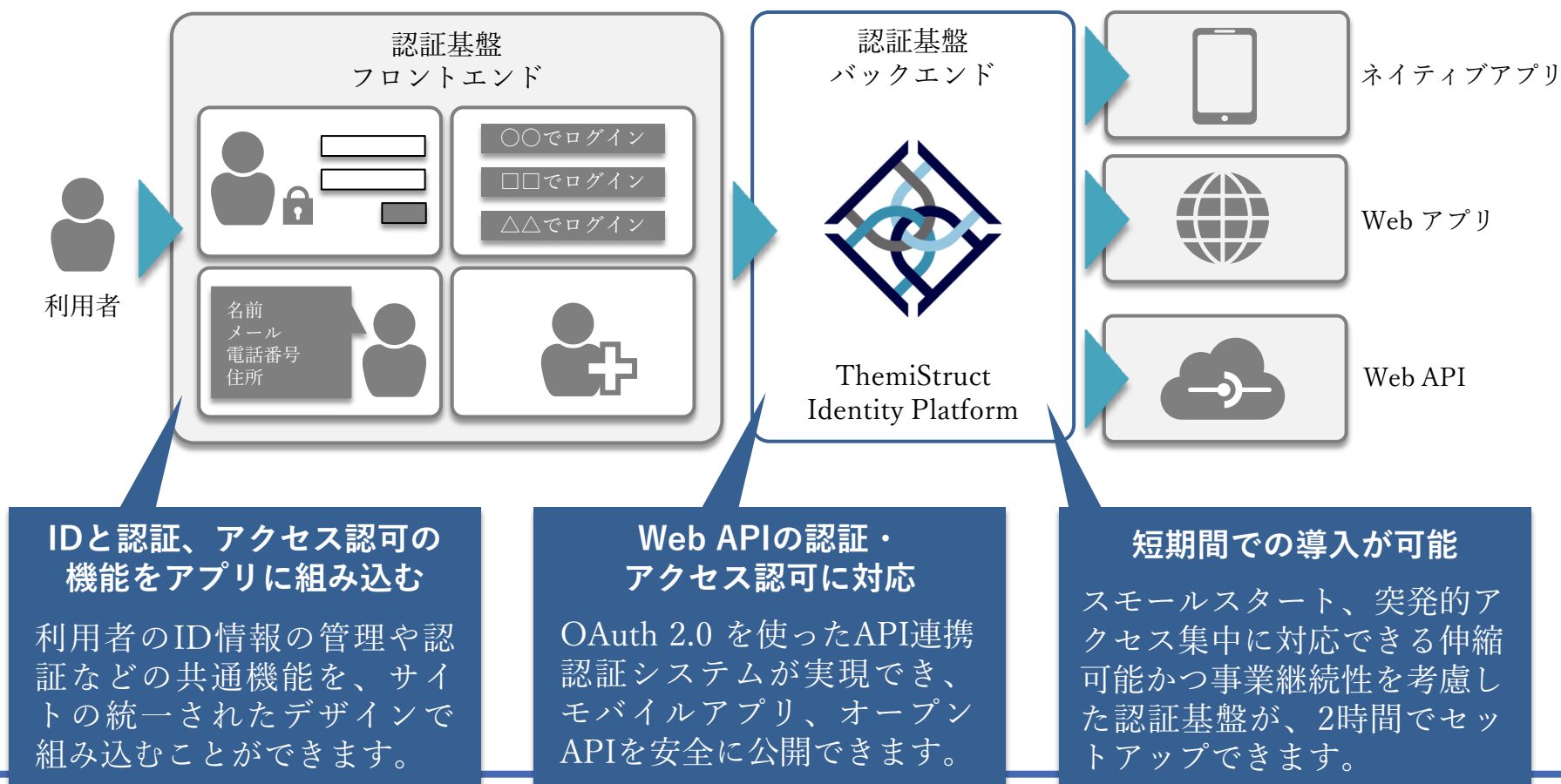
エンタープライズ領域に活用



エンタープライズアプリ群の
SSOやアクセス制御の基盤として利用

ThemiStruct Identity Platform を『顧客向けサイト』に活用

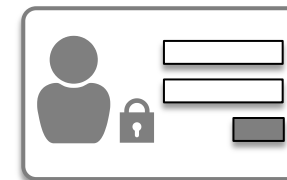
ThemiStruct Identity Platform を『顧客向けサイト』環境に適用した場合、サービスの利用者IDの管理と認証の機能を担うバックエンドサービスとして稼働します。これらの機能のUIにあたるアプリケーションの実装を支援し、実際のサービスアプリと接続するためのインタフェースを提供します。



『顧客向けサイト』に向けた3大特長

□ IDとユーザー認証、アクセス認可の機能をアプリに組み込む

- 利用者のID情報の管理や認証など利用者IDに関連する共通機能の実装を支援する『フレームワーク』と『Web API』を提供します。これらを活用し、貴社のサービスサイトに認証機能やパスワード変更機能、アプリケーションへのID連携機能などを組み込むことができます。



□ Web APIの認証・アクセス認可に対応

- OAuth 2.0 の技術仕様に基づいた認証・アクセス認可機能を提供します。OAuth 2.0 を使ったAPI連携認証システムが実現でき、モバイルアプリ、オープンAPIを安全に公開できます。



□ 短期間での導入が可能

- AWSマネージドサービスのコンポーネントを活用し、導入時におけるキャパシティやアベイラビリティプランニングから解放します。スモールスタート、突発的アクセス集中に対応できる伸縮可能かつ事業継続性を考慮した認証基盤が、2時間でセットアップできます。

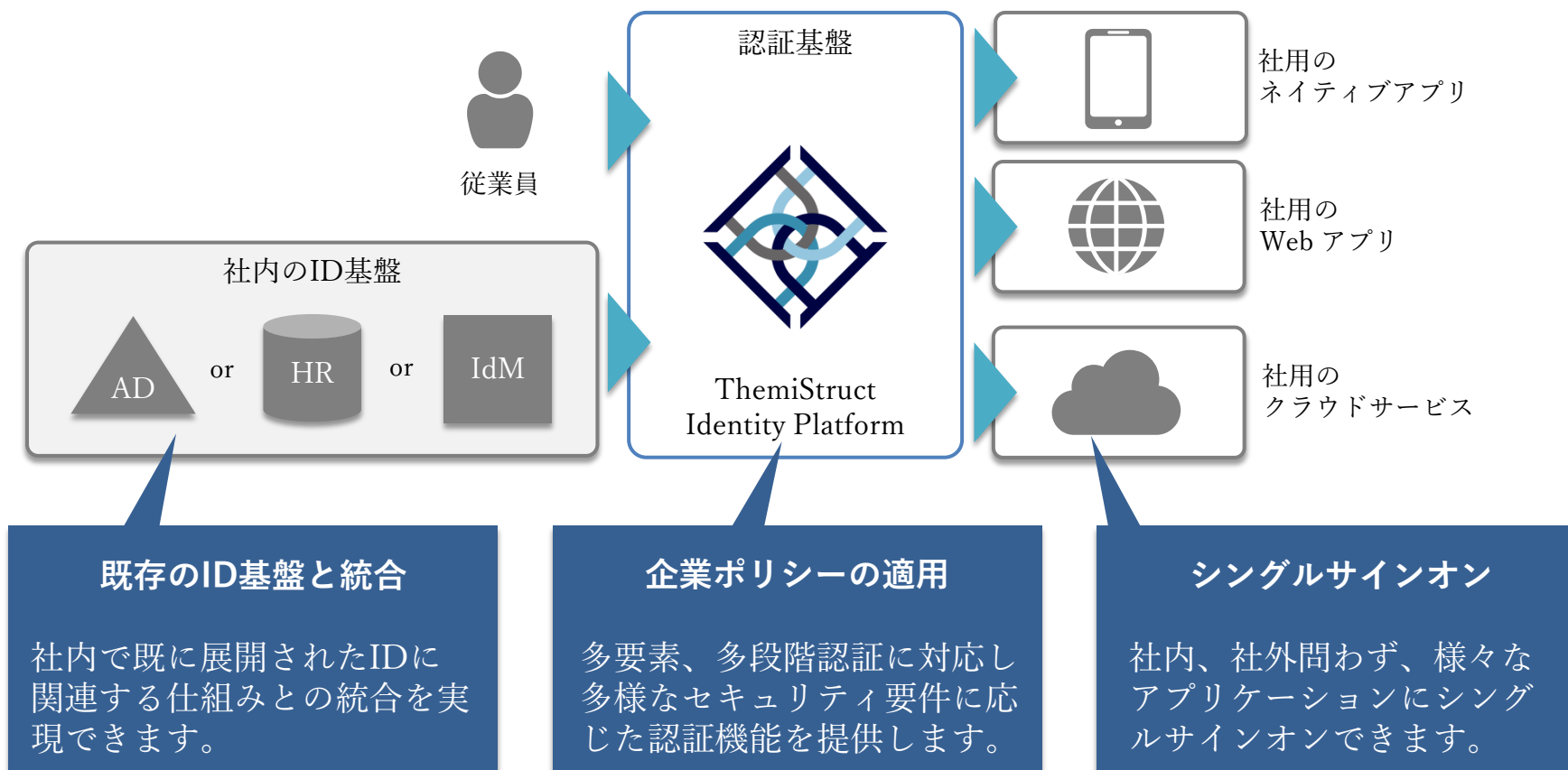


✓ High-Availability

✓ Scalability

ThemiStruct Identity Platform を『エンタープライズ』に活用

ThemiStruct Identity Platform を『エンタープライズ』環境に適用した場合、社内外のアプリケーションにシングルサインオン可能な認証基盤を提供できます。また、企業のポリシーにあった認証機能の設定や既存のIDとの統合をすることができます。



『エンタープライズ』に向けた3大特長

□ シングルサインオン

- OpenID Connect などの標準技術仕様を用いたシングルサインオンに対応しています。社内、社外問わず、様々なアプリケーションにシングルサインオンできます。



□ 企業ポリシーの適用

- ユーザーの属性や状態、利用するアプリケーションに応じて、柔軟な認証ポリシーをデザインすることができます。例えば、社内ネットワークからのアクセスの場合、IDとパスワードの認証を提供し、外出先からのアクセスの場合、追加でワンタイムパスワード認証を提供するといったポリシーを展開することができます。



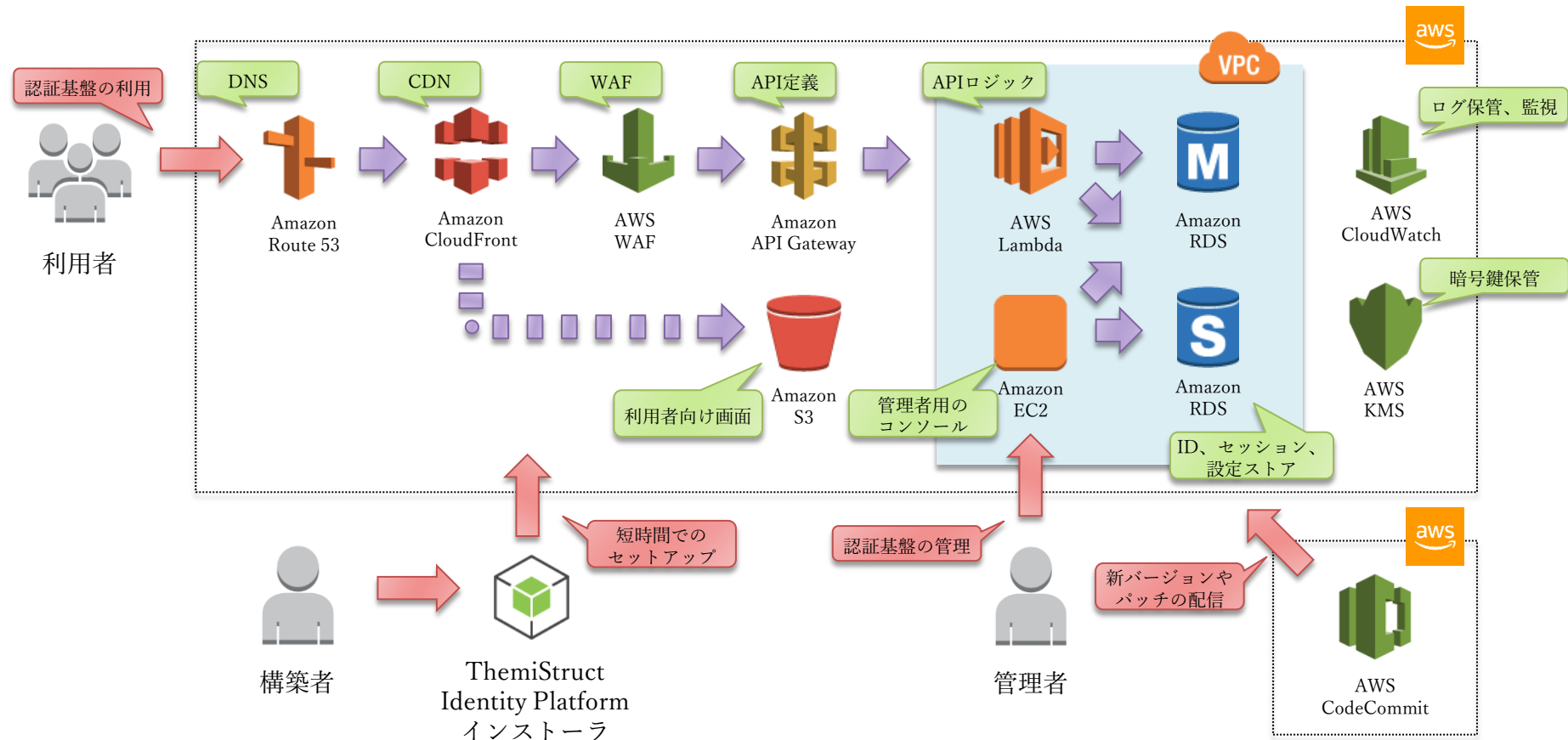
□ 既存のID基盤と統合

- 既存のID基盤と統合に向けたアカウント管理APIを提供しています。これにより、社内で既に展開されたIDに関連する仕組みとの統合を実現できます。



サーバーレスアーキテクチャを採用

ThemiStruct Identity Platform は Amazon Web Services のマネージドサービス上で稼働するため、一定のアベイラビリティ、スケーラビリティを実現することができます。また、以下の構成のセットアップを約2時間で完了できるインストーラを提供しています。



オージス総研のAPIソリューション

- オージス総研のクラウドとAPI開発・運用の知見を集めたサービス
 - FintechやIoTなどのデジタルビジネスに必要なAPI公開を実現
 - API導入のための技術調査・検討からAPIの公開・運用までフルカバー

■ API導入コンサルティングサービス

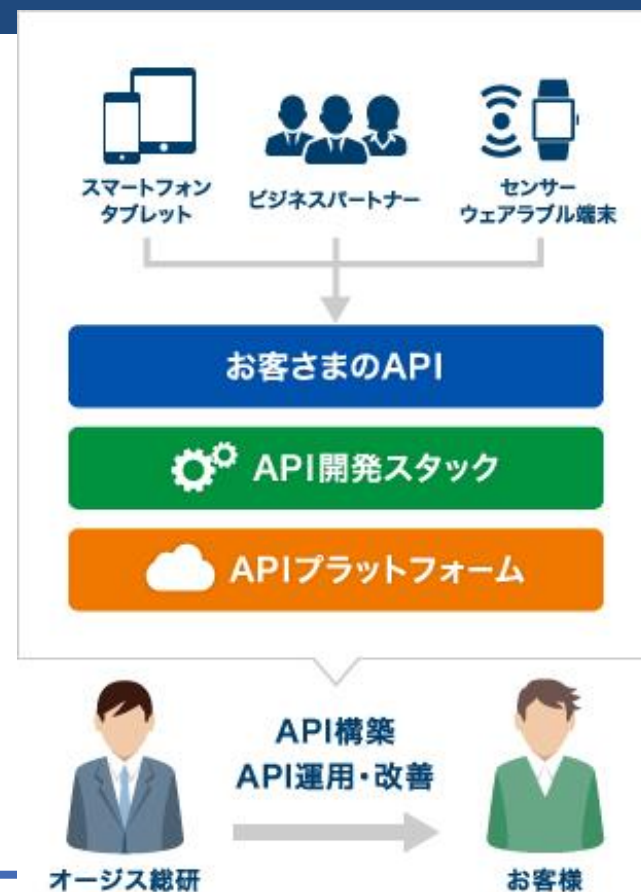
- ✓ ロードマップの策定援や、初期システムアーキテクチャ策定、API導入PoC等によりAPI導入を支援

■ API構築サービス

- ✓ ベストプラクティスを組み込んだAPI開発スタックを使い、お客様の環境に合ったAPIプラットフォーム、APIを迅速かつ効率的に開発

■ API運用サービス

- ✓ 継続的なAPIの改善・バージョンアップ
- ✓ 安定稼働のためのAPIプラットフォームの監視・障害対応
- ✓ API利用者向けサポートサービス



まとめ

まとめ

- 認証基盤のユースケースは、顧客向けサービス向け、オープンAPI向けへと広がり、重要性がより高まっている。
- 高い可用性、アクセスの増減、繁閑に対応できる認証基盤の実現が求められている。
- OpenID Connect, SAMLといったID連携の標準技術に加え、OAuth 2.0 の標準仕様群に継続的に対応していく必要がある。
- 当社では Themistruct Identity Platform をサーバーレスアーキテクチャの認証基盤を実現するパッケージとして提供。
- 共通ID基盤、API連携認証システム の構築ニーズに対応していく。

ご清聴ありがとうございました



ThemiStruct
テミストラクト

【お問い合わせ先】

株式会社オージス総研

TEL: 03-6712-1201 / 06-6871-8054

mail: info@ogis-ri.co.jp

