OpenID ファウンデーション・ジャパン

日本ネットワークセキュリティ協会

EIWG(Enterprise Identity Working Group)

OpenID Connect と SCIM の

エンタープライズ利用ガイドライン

作成日: 2013 年 12 月 20 日

リビジョン: 1.0

目次

はじめに　本書の目的と構成

本書は、エンタープライズ IT 向けにクラウドサービスを提供するクラウドサービス事業者に対して、コンシューマ IT で普及しているフェデレーション標準プロトコルである OpenID Connect と、現在標準化の検討が進んでいるアイデンティティ・プロビジョニング標準プロトコルである SCIM(System for Cross-domain Identity Management)を利用することの有用性と、その利用方法を説明したガイドラインである。なお、本書は OpenID ファウンデーション・ジャパンが主催する Enterprise Identity WG において、日本ネットワークセキュテリィ協会アイデンティティ管理 WG の協力にもとづいて作成した文書である。

1. エンタープライズ IT におけるフェデレーション標準プロトコルとアイデンティティ・プロビジョニング標準プロトコルの有用性

1.1 クラウドサービス事業者にとってのフェデレーション標準プロトコルの有用性

〔クラウドサービス事業者がクラウドビジネスを行う上での課題〕

クラウドサービス事業者がクラウドサービスを展開するにあたり、まず考慮しないといけないことは、クラウドサービスの利用を検討している企業が、サービス事業者に対して抱いているセキュリティ面の不安を払しょくすることである。

〔クラウドサービス利用者が抱くセキュリティ面の不安〕

クラウドサービスを利用する場合、クラウドサービス利用企業は情報セキュリティマネジメントの一部をクラウドサービス事業者(利用企業から観て外部組織)に依存することになる。多くの利用企業が内部統制(J-SOX)対策として、セキュリティポリシや IT システムの運用管理ポリシを策定した時代には、まだクラウドサービスは普及しておらず、自社内の IT リソースを利用することを前提としたポリシであり、クラウドサービスの利用は考慮されていなかった。

クラウドサービスの利用に際して、利用企業は、サービス事業者のセキュリティポリシや運用管理ポリシを考慮する必要が生じる。しかし、両者は別法人であるため、自社内と同様のコントロールを適用することは困難な場合が多い。

一方、サービス事業者は、サービス・レベル・アグリーメント(SLA)の中で、クラウドサービスの運用管理ポリシや技術面の脆弱性について十分に説明していない場合もあり、利用企業はクラウドサービスのコスト削減競争に伴う情報管理品質低下の懸念も含めて、サービス事業者に機密情報を預けることに対するセキュリティ面の不安を感じている場合が多い。機密情報の中でも特に、パスワード情報を含むアイデンティティ情報については、万が一漏えいした場合のセキュリティ被害の影響度の大きさから、利用企業の警戒感は特に強い。

〔コンシューマ IT 環境における情報漏えい事件の悪影響〕

コンシューマ IT 環境では、クレジットカード番号やセキュリティコード、パスワード情報等の漏えい事件が多発している。エンタープライズ IT においてクラウドサービスの利用を検討している企業は、このようなニュースを見聞きし、クラウドサービス事業者にアイデンティティ情報を預けることについて、さらに不安を募らせている。

〔クラウドサービス事業者の対応策～透明性確保と預かる情報の削減〕

クラウドサービス事業者がこれらの不安を払しょくするには、以下の 2 つの対策が考えられる。

1 つ目は、提供するクラウドサービスに関するセキュリティポリシや運用管理ポリシを策定し、これに沿った厳格な運用管理体制を構築・維持し、クラウドサービス利用企業に対してこれらの状況についての説明を行うことで、サービスの透明性を確保することである。

2 つ目は、利用企業から預かる情報をできる限り少なくすることである。

〔フェデレーションの利用による認証・認可〕

クラウドサービス事業者がクラウド利用企業から預かる情報をできる限り少なくするには、ユーザに対する認証・認可の方式について、サービス事業者が事前に必要なアイデンティティ情報をすべて保持し、サービス事業者側で認証・認可処理を行うローカル認証方式から、利用企業側に認証・認可処理を委譲するフェデレーション方式へと変更する方法が有効である。利用企業は自社内で認証処理を行い、その認証処理の結果を ID トークンとしてサービス事業者に引き渡す。これにより、サービス事業者は、アイデンティティ情報の中でも特に機密性の高いパスワード情報を預からずに、クラウドサービスを提供することができる。

1.2 クラウドサービス利用企業にとってのフェデレーション標準プロトコルの有用性

〔クラウドサービスに対するシングルサインオン〕

クラウドサービス利用企業にとっては、シングルサインオンの観点からフェデレーションは有用である。企業内のシステムに対するシングルサインオンは、対象システムが企業の管理下にあることを前提として、様々なシングルサインオン認証システム(パッケージソフト)を利用し、エージェント方式(管理対象にインストールするモジュール)やリバースプロキシ方式によって実現される。

一方、クラウドサービスは利用企業の管理下にはなく、クラウド利用企業が多岐にわたるマルチテナント環境において、クラウドサービス事業者が、利用企業の採用している様々なシステム(パッケージソフト)や方式のシングルサインオンに逐次対応することは困難である。

また、企業外からのアクセスも可能であるクラウドサービスの特性から、アクセス制御を実施する箇所を企業内のアクセス制御用サーバに一元化するリバースプロキシ方式の採用は、クラウドサービス利用に際しては現実的ではない。クラウドサービスに対してシングルサインオンを行うには、フェデレーションを利用することが最善策となる。

〔情報共有〕

親会社～子会社間や、製造業におけるサプライチェーンに参加する企業間で、クラウドサービスを利用して情報を共有する機会は益々増えてきている。複数の企業間で情報を共有する場合に利用するクラウドサービスの認証方式が、従来型のローカル認証システムである場合、パスワードを含むアイデンティティ情報は、利用するクラウドサービスの数だけ拡散されることになる。そこで、フェデレーションを利用することにより、クラウドサービスから認証処理を分離し、認証処理はクラウドサービス側ではなく情報を共有する各企業内で行うことで、ID 情報の拡散を防ぐことが可能になる。

クラウドサービスの契約主体＝共有する情報の所有者である企業にとっても、万が一情報が漏えいした場合の賠償責任リスクを考慮すると、他社のアイデンティティ情報の保持はできる限り少なくしたいはずで、ここでもフェデレーションは有用な手段となる。

〔認証処理の選択の自由〕

クラウドサービス利用企業は、フェデレーションの仕組みを構築することにより、自社で認証基盤を保有することになる。これにより、利用するクラウドサービスを拡張していく中で多要素認証等により認証手段を強化したい場合には、クラウドサービスの認証システムの制約を受けずに、自社で多要素認証に対応したシングルサインオンシステムを構築するなど、セキュリティポリシに沿った認証システムを自由に構築することができるようになる。

また、一方でクラウドサービス事業者としても、機密度の高いコンテンツを保有している場合に、アクセスしてきたユーザがフェデレーションを利用して認証済みであったとしても必要とされる認証レベルに達していないときには、利用企業に再度認証を求める処理を行うべきである。それに合わせて利用企業側でも、要求された認証レベルに応じた適切な認証方式を用いることが必要になる。

〔クラウド利用企業に有用な仕組みはクラウド事業者にとっての競争力となる〕

このように、フェデレーションプロトコルを利用することは、クラウドサービス利用企業にとって有用である。また、クラウドサービス事業者にとっても、利用企業の利益を確保する観点からフェデレーションを自社のクラウドサービスで採用することは、クラウドサービスの競争力の強化に結びつく価値があると言える。

〔OpenID Connect〕

このフェデレーションの仕組みとして今注目を浴びているのが、コンシューマ IT 向けクラウドサービスにおいて普及が進む OpenID Connect である。

1.3 クラウドサービス事業者にとってのアイデンティティ・プロビジョニング標準プロトコルの有用性

〔エンタープライズ IT でのフェデレーションとプロビジョニングの併用〕

エンタープライズ系システムにおけるアクセス制御は、単純に組織や役職によるだけではなく、業務上の役割や引き継ぎ期間等も考慮した複雑なロール管理を必要とする場合が多い。また、システムを複数の従業員で共有する場合、従業員同士を識別するための氏名、所属、役職といったアイデンティティ情報がアプリケーションデータとして必要とされる場合も多い。

このようなエンタープライズ系システムにおいても、フェデレーションを利用することで、クラウドサービス事業者が事前に保持するアイデンティティ情報をできる限り少なくすることは可能だが、アクセス制御処理やアプリケーションデータとしてのアイデンティティ情報の処理は、事前にプロビジョニング(配布)されたデータを元に実施される設計になっているため、フェデレーションでのやり取りを基にした処理に変更するには大きな工数とコストが必要となる。変更に要する多大なコストを考えると、フェデレーションを利用する場合であっても、プロビジョニングも併用することを選択する事業者が多いと考えられる。

日本のエンタープライズ IT の特性に起因するフェデレーションとプロビジョニングの併用については 4 章でさらに詳しく説明する。

〔ID 統合管理を実現するために必要なプロビジョニング標準プロトコル〕

企業内(オンプレミス)において既に ID 管理システムを構築している企業の場合、新たに利用するクラウドサービスのアイデンティティ情報も統合管理したいという要求は非常に強い。この要求に対応するためにクラウドサービス事業者は、アイデンティティ情報をメンテナンスするためのインタフェースとして、画面(Graphical User Interface)の提供だけでなく、プロビジョニング用インタフェースを用意し公開することが求められる。

〔アイデンティティ・プロビジョニング標準プロトコルの必要性〕

このように、エンタープライズ IT 向けのクラウドサービスにはアイデンティティ・プロビジョニング標準プロトコルへの対応が必要であり、クラウドサービス事業者にとっては、アイデンティティ・プロビジョニング・インタフェースを自ら開発するより、標準化されたものが存在し、これを利用することができれば、開発工数を大きく削減することが可能になる。

1.4 クラウドサービス利用企業にとってのアイデンティティ・プロビジョニング標準プロトコルの有用性

〔CSV の後始末問題に対する不安〕

もし、クラウドサービスがアイデンティティ情報の一括取り込み機能として CSV ファイルを利用する方式しか有していなかった場合、クラウドサービスへのアイデンティティ情報の取り込み処理が完了した後、CSV ファイルを即時に削除する仕組み(運用)になっているか否か、あるいは利用者が削除のタイミングを自ら設定できるかどうかは、利用企業のセキュリティの観点から非常に重要なポイントとなる。しかし、クラウドサービス事業者が CSV ファイルを後始末する仕組み(運用)について適切な説明を行っていない場合もあり、CSV ファイルの所有者であるクラウドサービス利用企業は、セキュリティ面から不安を感じることになる。

一方、アイデンティティ・プロビジョニング標準プロトコルがあれば、このような中間ファイルとしての CSV ファイルは必要が無くなる。

〔クラウドサービス利用拡張時のコスト〕

クラウドサービスにアイデンティティ・プロビジョニング・インタフェースがあったとしても、それが標準プロトコルではない場合、クラウドサービス利用企業が新たなサービスを利用するには、その都度、サービス個別に提供されたアイデンティティ・プロビジョニング・インタフェースに対する連携システムを開発したり、ID 管理ツールの連携用オプション部品を購入したりしなければならず、クラウド利用企業のコスト負担は大きくなる。

一方、アイデンティティ・プロビジョニング標準プロトコルがあれば、連携システムとしては一度開発したシステムの再利用が可能になり、ID 管理ツールの連携用オプション部品の購入にしても販売価格設定の低下を期待することができる。

〔SCIM〕

このアイデンティティ・プロビジョニングの仕組みとして標準化の検討が進んでいるのが、SCIM(System for Cross-domain Identity Management)である。

2. フェデレーション標準プロトコル ～ OpenID Connect 解説

2.1 はじめに

OpenID Connect 1.0（以下 OpenID Connect と表記）は、OpenID 2.0 の次期バージョンとなる予定の仕様群である[1]。同仕様は OAuth 2.0 プロトコル[2]上にシンプルな「アイデンティティ層」を定義するものであり、「クライアント」（以下、RP（リライイング・パーティ）と表記）が、「認可サーバ」（以下、OP（OpenID プロバイダ）の認証結果を基に、「エンドユーザ」のアイデンティティを検証することができるようになる。

想定される活用例としては、

- OP へのユーザ認証の一元化による、RP 間のシングルサインオン
- RP 側でユーザのクレデンシャル（パスワードなど）の管理が不要となることによる、セキュリティ向上と管理負荷の低減
- OP からのユーザ属性情報取得による、RP での新規ユーザ登録の容易化
- エンドユーザの認証と API アクセス認可の一体化によるユーザ利便性の向上

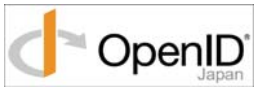
などがある。

2013 年 7 月現在、OpenID Connect 仕様はファイナライズされておらず、まだ実装者向けドラフトの段階であるものの、複数の製品・サービスにおいて、すでに実装されている。

以下に一例を示す（順不同）。

- Yahoo! JAPAN (YConnect)
- 日本経済新聞社（日経 ID）
- 東急電鉄
- Google
- PayPal (Log In with PayPal)
- 野村総合研究所（Uni-ID）

1 Connect | OpenID https://openid.net/connect/
2 The OAuth 2.0 Authorization Framework http://openid-foundation-japan.github.io/rfc6749.ja.html



- Ping Identity (PingFederate)
- Gluu (OX)
- Layer 7

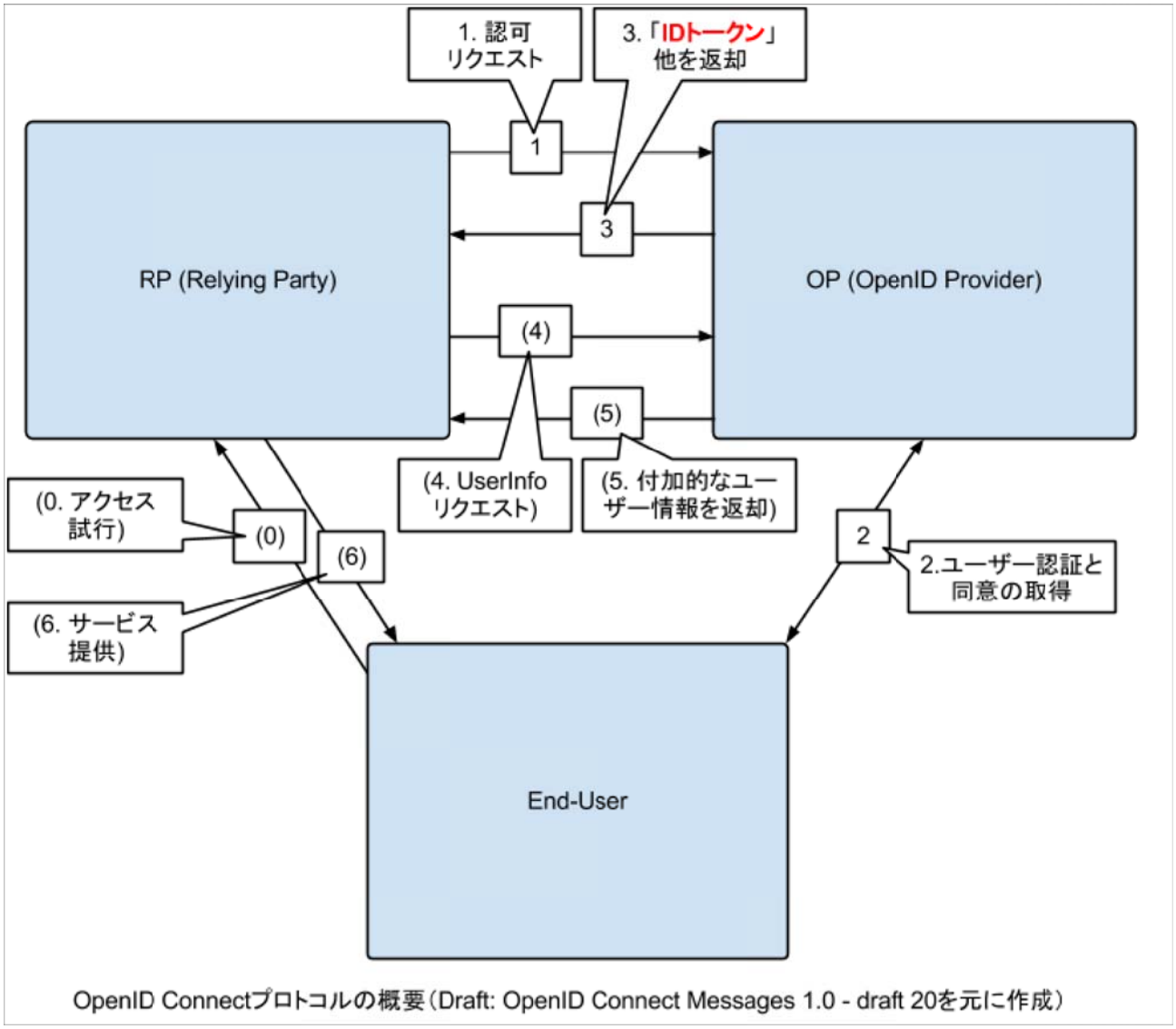
なお、仕様策定の背景は以下の文書を参照されたい。

- 情報セキュリティ技術動向調査(2011 年下期)[3]

2.2 OpenID Connect によるフェデレーション

OP と RP との間でのフェデレーションは、エンドユーザの関与の元、RP が OP に「ID トークン」というデータを要求し(認可リクエスト)、これを取得することによって実現される。

3 IPA 独立行政法人 情報処理推進機構:情報セキュリティ技術動向調査(2011 年下期) 3. OpenID Connect http://www.ipa.go.jp/security/fy23/reports/tech1-tg/b_03.html



OpenID Connectプロトコルの概要（Draft: OpenID Connect Messages 1.0 - draft 20を元に作成）

以下の節では、IDトークンの概要と、そのやり取り（プロトコル・フロー）について概要を示す。

2.2.1 IDトークン

「IDトークン」とは、OPにおけるユーザ認証イベントの情報である。OPはRPからの認可リクエストに応じ、エンドユーザの認証、およびOPからRPへの情報提供に関する同意を取得し、RPにIDトークンを返却する。RPはIDトークンに含まれるユーザ識別子を元にユーザを特定し、その他の認証イベントの情報に基づき、ユーザがRPにアクセスすることを認可する。

IDトークンはJSON Web Token (JWT) 形式であり、複数のクレーム（エンティティ、この場合にはユーザに関

し、OP が表明する情報）を含む。RP はユーザのアクセスの認可を行う上で、主に以下のクレームを利用することになると考えられる[4]。

クレーム名	説明	値の例
sub	サブジェクト（この場合はユーザ）を識別する値（識別子）であり、OP 内にて一意かつ再利用されない。	“24400320” “AItOawmwtWwcT0k51BayewNvutrJUqsvl6qs7A4”
exp	ID トークンの有効期限。1970-01-01T0:0:0Z からの秒数。RP はこの有効期限を過ぎた ID トークンを処理してはならない。	1311281970
auth_time	ユーザ認証を実施した日時。1970-01-01T0:0:0Z からの秒数。	1311280969
acr	認証コンテクスト・クラス・リファレンス（Authentication Context Class Reference）。OP の実施したユーザ認証が、どの認証コンテクスト・クラスに属するかを示す。	″0″ ″urn:mace:incommon:iap:silver″
amr	認証手段リファレンス（Authentication Methods References）。OP が実施したユーザ認証の手段を示す。	“otp”

なおセキュリティ上、実際に ID トークンを処理する際には、RP はこの他のクレームの値（例: iss（ID トークンの発行者）、aud（ID トークンの受け手）、iat（ID トークンの発行日時））を用いて ID トークンの真正性および有効性を検証する必要がある。また RP が ID トークンを OP との直接通信以外の経路（例: Web ブラウザのリダイレクト経由）で取得した場合には、RP は ID トークンに付与されている署名を検証する必要がある[5]。

RP は認可リクエストに際し、ID トークンに含めてほしいクレームや、ID トークン生成にあたりどのような認証イ

4 Draft: OpenID Connect Messages 1.0 – draft 20 2.1.2.1. ID Token http://openid.net/specs/openid-connect-messages-1_0-20.html#id_token

5 OpenID Connect Messages 1.0 – draft 20 4.2. ID Token Validation http://openid.net/specs/openid-connect-messages-1_0-20.html#id.token.validation

ベントを求めるかを指定する。OP はその指定を考慮した上で、ID トークンを返却する（RP の指定が必ずすべて満たされるものではないことに注意）。以下に指定可能な内容[6]の一部を示す。

パラメタ名	説明	値の例
scope	OAuth 2.0 (RFC 6749) の scope[7]。OpenID Connect では必須値として openid を指定する。加えて、クレームのセットを要求するための値を指定する（後述）。	openid profile email address phone offline_access
display	ウィンドウのポップアップや、タッチ・インターフェース向けのユーザ・インターフェース（UI）など、OP によるエンドユーザの認証および同意確認の際の UI を指定する。	page popup touch
prompt	OP によるエンドユーザの認証や同意の再実行の要否を指定する。	none login consent select_account
max_age	OP がエンドユーザを明示的に認証してからの経過時間として、RP が許容可能な値を指定する。	86400
ui_locales	OP のエンドユーザ向け UI のロケールを指定する。	fr-CA fr en
claims_locales	OP から提供されるクレームのロケールを指定する。	fr-CA fr en
id_token_hint	OP がエンドユーザを認証する際のヒント。認可リクエストの対象となるエンドユーザに関して RP が OP から前回提供を受けた ID トークンを指定する。	
login_hint	OP がエンドユーザを認証する際のヒント。エンドユーザが OP にてユーザ認証を受けるときに用いるであろう値を指定する。	（メールアドレスなど）
acr_values	OP によるエンドユーザの認証に関し、RP が求める認証コンテクス	″1″

6 Draft: OpenID Connect Messages 1.0 – draft 20 2.1.1.1. Request Parameters http://openid.net/specs/openid-connect-messages-1_0-20.html#RequestParameters

7 The OAuth 2.0 Authorization Framework 3.3. アクセストークンのスコープ http://openid-foundation-japan.github.io/rfc6749.ja.html#scope

	ト・クラス・リファレンスの値を指定する。	″urn:mace:incommon:iap:silver″

2.2.2 OpenID Connect プロトコル・フロー

「ID トークン」の要求・提供は、OAuth 2.0 仕様をベースとするプロトコル・フローによって行われる。OpenID Connect のプロトコル・フローは、ユースケースに応じ、「認可コードフロー」と「Implicit フロー」の二種類が規定されている[8]。ここではそれぞれのフローの概要を示す。

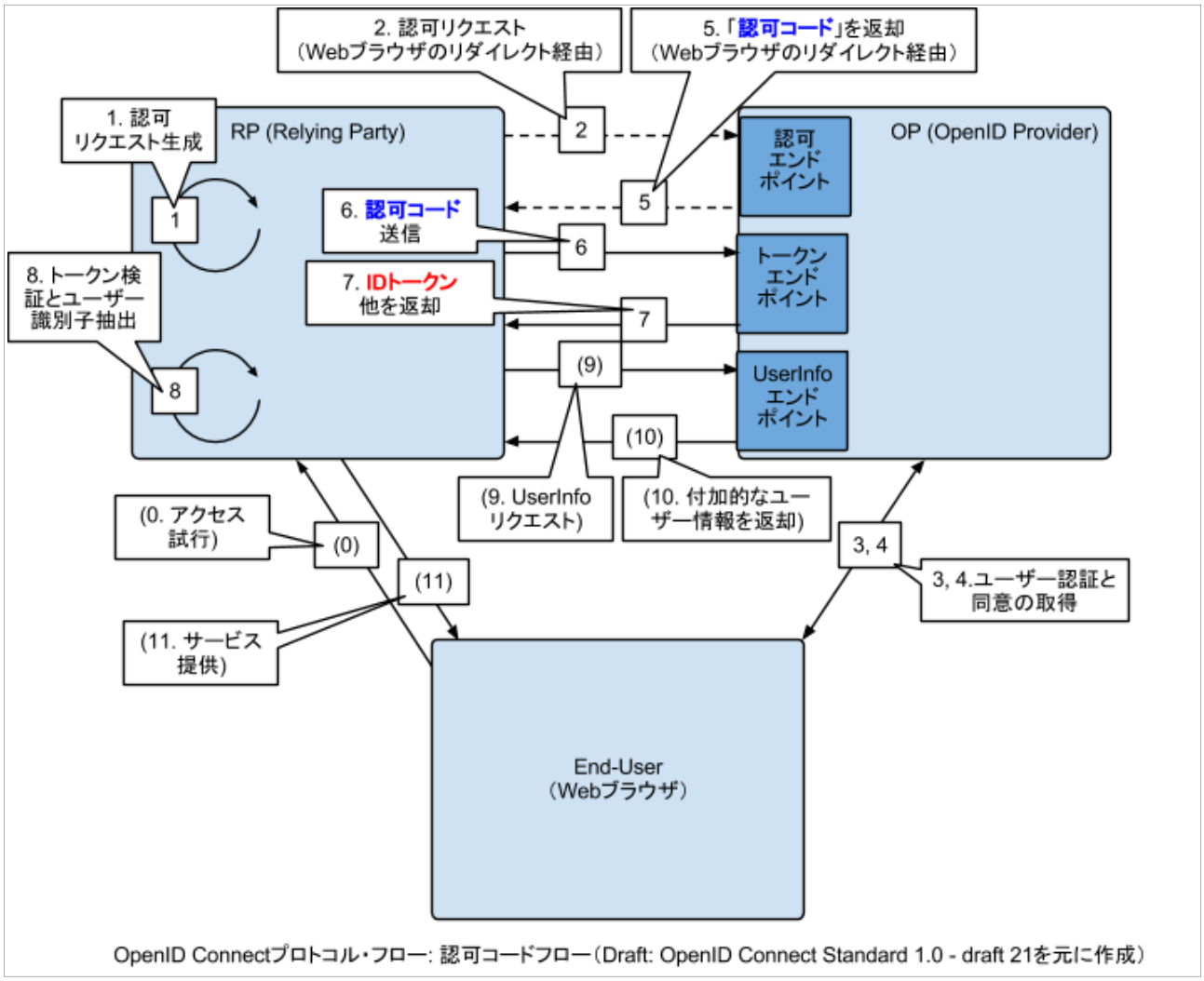
2.2.2.1 認可コードフロー

認可コードフローは、RP から OP へエンドユーザを経由して送信された認可リクエストに対して、OP が「認可コード」と呼ばれる値を同じくエンドユーザ経由で RP に返却し、そして RP がその認可コードを OP に送信して ID トークン（ならびにアクセストークン）を得るフローである。ID トークンのやり取り（トークン・リクエスト/レスポンス）が RP と OP との直接通信によって行われるため、OP による ID トークンへの署名は基本的には不要となり、RP 側での署名検証処理も発生しない。

OP は RP からのトークン・リクエストに際し、クライアント・シークレットにより RP を認証する。そのため RP は、Web サーバ型のアプリケーションのように、クライアント・シークレットをセキュアに管理できる構成でなくてはならない。

以下は認可コードフローのシーケンス概要である。

8 Draft: OpenID Connect Standard 1.0 - draft 21 2.1. Protocol Flows http://openid.net/specs/openid-connect-standard-1_0-21.html#protocol_flows

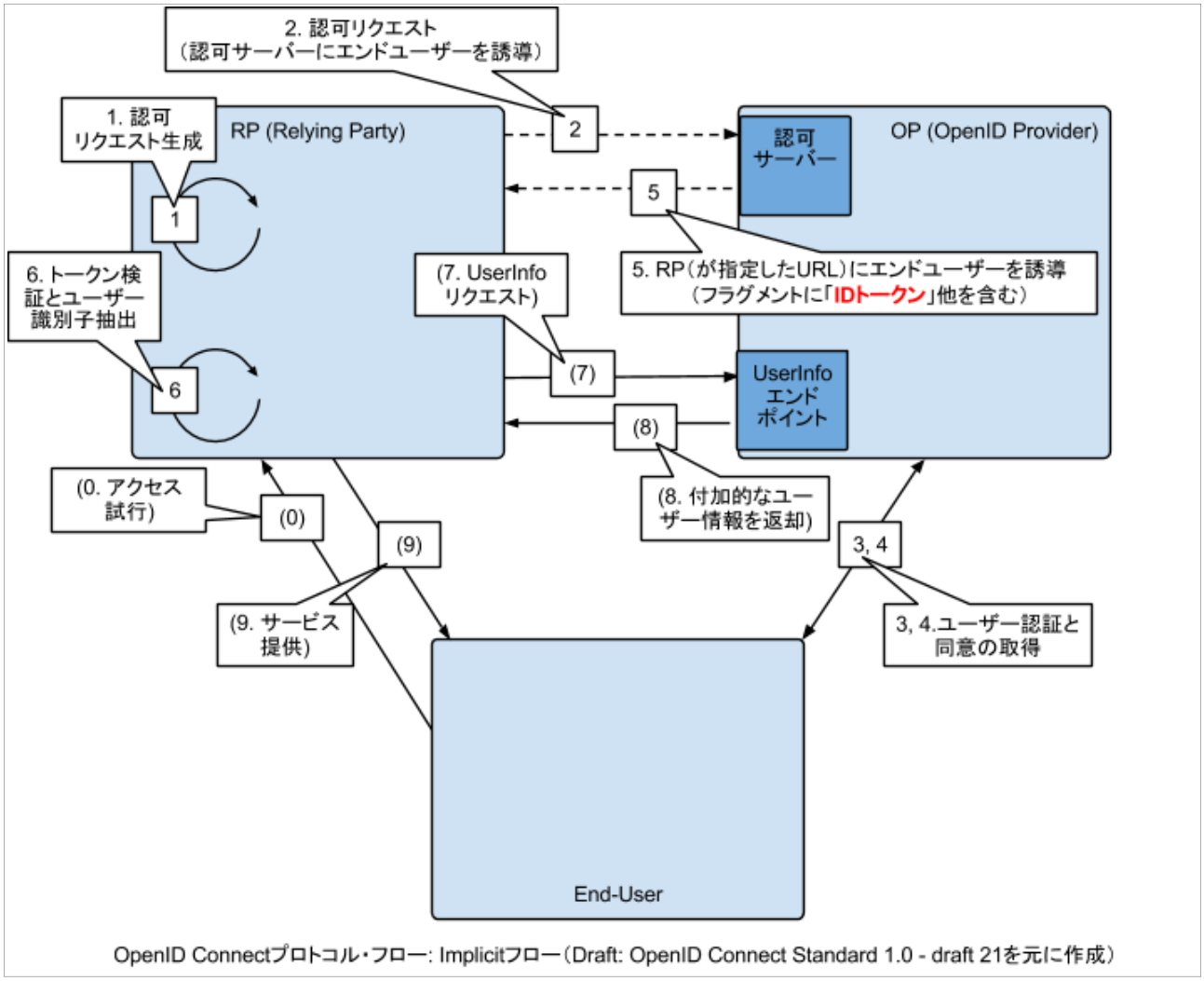


OpenID Connectプロトコル・フロー: 認可コードフロー（Draft: OpenID Connect Standard 1.0 - draft 21を元に作成）

2.2.2.2 Implicit フロー

Implicit フローは、認可コードフローと異なり、エンドユーザ経由で OP が ID トークン（ならびにアクセストークン）を RP に返却するフローである。ID トークンの授受に関し、RP から OP への通信が発生しないため、RP から OP への直接通信が行えない環境にも適用することが可能となっている。一方、OP による ID トークンへの署名、および RP 側での署名検証処理は必須となる。

以下は Implicit フローのシーケンス概要である。



OpenID Connectプロトコル・フロー: Implicitフロー（Draft: OpenID Connect Standard 1.0 - draft 21を元に作成）

このように、OP は ID トークンを URL フラグメントにエンコードして RP に返却するため、RP 側ではその URL フラグメントから ID トークンを抽出するための処理を実装する必要がある。典型的には、RP は Web ブラウザになんらかのスクリプトをダウンロードさせて[9]、そのスクリプトによってフラグメントから ID トークンを抽出し、Web サーバ・アプリケーションに送信させることとなる。

2.3 OpenID Connect によるユーザ属性のやり取り

RP は OP に対する認可リクエストの際、ユーザ認証イベントに加え、ユーザの属性情報も同時に要求することができる。以下の節では、RP から OP へのユーザ情報の要求、および OP から RP への提供の仕組みについて

9 OpenID Connect Implicit Client Profile 1.0 – draft 11 2.1.5.3. Example Redirect URI Response
http://openid.net/specs/openid-connect-implicit-1_0-11.html#implicit_callback

概要を示す。

2.3.1 RP から OP へのユーザ属性の要求

RP は認可リクエストのパラメタに、取得したいユーザ属性を指定する。指定方法は下記のいずれかとなる。

- scope パラメタに、OP が用意した「ユーザ属性のセット」を指定
- claims パラメタに、リクエストするユーザ属性名を個々に指定

本稿では、claims パラメタと比較して、実環境において広く採用されている、scope パラメタによる指定について紹介する。

2.3.1.1 scope パラメタによるユーザ属性のリクエスト

「2.2.1 ID トークン」にて述べたとおり、認可リクエストの scope パラメタは、OAuth 2.0 (RFC 6749) の scope であるが、OpenID Connect 仕様では要求するユーザ属性のセットの指定にも用いられる。ユーザ属性のセットとして、同仕様ではあらかじめ以下の値を定義している[10]。

値	説明
profile	既定のプロファイル。以下のユーザ属性を含む。 name（氏名）、family_name（名字）、given_name（名前）、middle_name（ミドルネーム）、nickname（ニックネーム）、preferred_username（希望するユーザ名）、profile（プロファイルページの URL）、picture（プロファイル写真の URL）、website（ユーザの Web サイトの URL）、gender（性別）、birthdate（誕生日）、zoneinfo（タイムゾーン）、locale（ロケール）、updated_at（最終更新日時）
email	メールアドレス。以下のユーザ属性を含む。 email（希望するメールアドレス）、email_verified（メールアドレスが検証済みかどうか）
address	住所。
phone	電話番号。以下のユーザ属性を含む。 phone_number（電話番号）、phone_number_verified（電話番号が検証済みかどうか）

これらあらかじめ定義されている値以外に、OP が独自に定義した「ユーザ属性のセット」を指定することも、OpenID Connect 仕様では許容している。たとえば PayPal 社の OpenID Connect API (Log In with PayPal) では、

10 Draft: OpenID Connect Messages 1.0 – draft 20 2.4. Scope Values http://openid.net/specs/openid-connect-messages-1_0.html#scopes

RP は認可リクエストの scope パラメタに、PayPal 社独自の「ユーザ属性のセット」である「https://uri.paypal.com/services/paypalattributes」という値を含めることにより、「年齢範囲」「アカウント状態（認証済みかどうか）」「アカウントタイプ」「アカウント作成日」などのクレームを要求することができる。[11]

2.3.2 OP から RP へのユーザ属性の提供

RP からの認可リクエストに対する、OP のユーザ属性の提供方法として、OpenID Connect では、RP に返却する ID トークンに含める方法と、RP がアクセス可能な UserInfo エンドポイントを用意する方法の二通りを定義している。

2.3.2.1 ID トークン

ID トークンには認証イベントに関する情報以外のクレームを含めることもできる。たとえば Google 社では、（ユーザのメールアドレス）や email_verified（そのメールアドレスが検証済みかどうか）などのクレームを ID トークンに付加している。例を以下に示す[12]。

```
{"iss":"accounts.google.com",
 "at_hash":"HK6E_P6Dh8Y93mRNtsDB1Q",
 "email_verified":"true",
 "sub":"10769150350006150715113082367",
 "azp":"1234987819200.apps.googleusercontent.com",
 "email":"jsmith@example.com",
 "aud":"1234987819200.apps.googleusercontent.com",
 "iat":1353601026,
 "exp":1353604926 }
```

2.3.2.2 UserInfo エンドポイント

「UserInfo エンドポイント」は、OpenID Connect 仕様に定義されている、OP が RP にユーザ情報を提供するための API である。OAuth 2.0 仕様の「保護されたリソース（Protected Resource）」であり[13]、RP は、認可リクエストの際に ID トークンと同時に OP から取得したアクセストークンを用いてアクセスする。OP はユーザ情報を、通常は JSON 形式にて RP に返却する。

---

11 Log In with PayPal Integration Details
https://developer.paypal.com/webapps/developer/docs/integration/direct/log-in-with-paypal/detailed/#parameters

12 Using OAuth 2.0 for Login - Google Accounts Authentication and Authorization ー Google Developers 5. Obtain user information from the ID Token
https://developers.google.com/accounts/docs/OAuth2Login#obtainuserinfo

13 The OAuth 2.0 Authorization Framework 7. 保護されたリソースへのアクセス http://openid-foundation-japan.github.io/rfc6749.ja.html#access-resource

以下は、OpenID Connect 仕様に記述されている、UserInfo エンドポイントのレスポンス例[14]である。

```
HTTP/1.1 200 OK
Content-Type: application/json

{
 "sub": "248289761001",
 "name": "Jane Doe",
 "given_name": "Jane",
 "family_name": "Doe",
 "email": "janedoe@example.com",
 "picture": "http://example.com/janedoe/me.jpg"
}
```

14 Draft: OpenID Connect Standard 1.0 - draft 21 4.2. UserInfo Response http://openid.net/specs/openid-connect-standard-1_0-21.html#userinfo

3. アイデンティティ・プロビジョニング標準プロトコル ～ SCIM 解説

3.1 はじめに

SCIM（System for Cross-domain Identity Management）は、クラウドサービスに対するアイデンティティ・プロビジョニングを実装するためのインタフェースを共通化する標準プロトコルである。

これまで、クラウドサービスに対するアイデンティティ・プロビジョニングの実装は、クラウドサービスごとに提供されている独自のインタフェース（独自 GUI、独自 API、独自フォーマットの CSV ファイルなど）を用いて実装しているのが現実である。しかし、今後ますますクラウドサービスの活用が普及し、数多くのクラウドサービスに対するアイデンティティ・プロビジョニングが必要となる状況下で、クラウドサービスごとに異なるインタフェースを用いてアイデンティティ・プロビジョニングを実装しなければならないのは大きな問題である。

様々なクラウドサービスに対するアイデンティティ・プロビジョニングを実装するための共通インタフェースの標準仕様を提供することにより、この問題を解決するのが、SCIM の目的である。

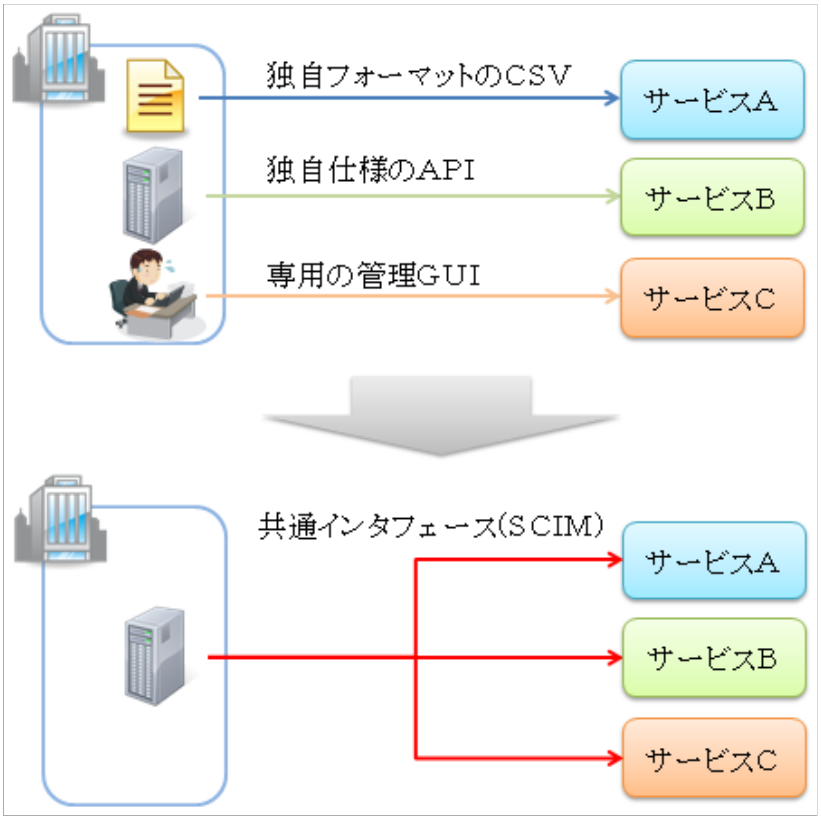


図 3.1: SCIM の目的

SCIM は、SCIM サービスコンシューマであるクラウドサービス利用者側から、SCIM サービスプロバイダであるクラウドサービス側へ、アイデンティティ情報のプロビジョニングと管理の操作を行うためのクライアントーサーバ型のプロトコルである。その構成は、両者間でやり取りする通信プロトコルに関する仕様と、プロトコル上でやり取りするデータフォーマットに関する仕様からなる。それぞれの仕様の中身は、後述する。

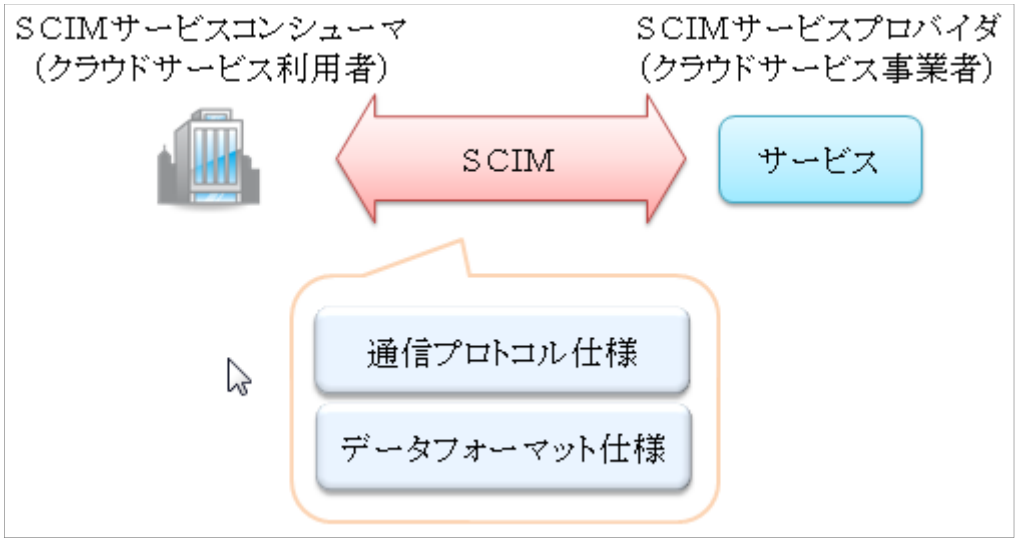


図 3.2: SCIM の位置付けと構成

SCIM 仕様の標準化は、2011 年 12 月にバージョン 1.0 がリリースされ、2012 年 7 月にバージョン 1.1 がリリースされている。2013 年 8 月時点での最新の標準はこのバージョン 1.1 である。IETF で次バージョン 2.0 の検討が進められている。

3.2 SCIM モデル

SCIM の仕様は、先に述べたとおり、SCIM のサービスコンシューマとサービスプロバイダの間でやり取りする通信プロトコルに関する仕様と、プロトコル上でやり取りするデータフォーマットに関する仕様からなる。

本節ではまず、データとプロトコルのモデルを説明する。

(1)データモデル

SCIM のデータモデルは、オブジェクトデータモデルをベースとしている。

「リソース」と呼ぶオブジェクトと、それを構成する「属性」および「属性値」の集合からなる。

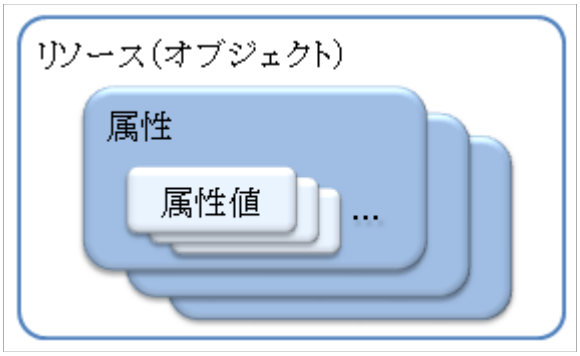


図 3.3: リソース、属性、属性値の関係

(2)プロトコルモデル

SCIM のプロトコルモデルは、SCIM のサービスコンシューマからサービスプロバイダへの HTTP ベースのクライアントーサーバ型プロトコルであり、REST API を提供している。

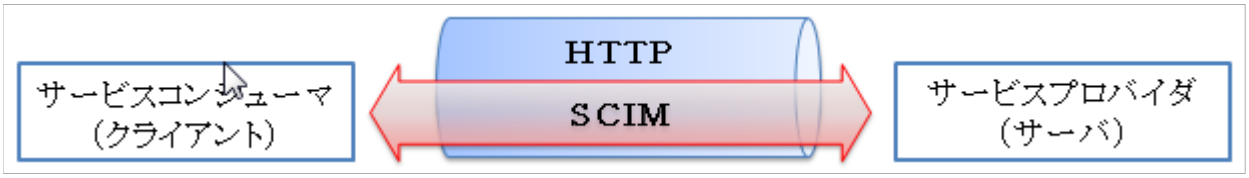


図 3.4: SCIM のプロトコルモデル

3.3 SCIM スキーマ仕様

本節では、SCIM データモデルの構成要素である「リソース」および「属性」の SCIM 標準スキーマ仕様や、スキーマの拡張方法、表記方法などを説明する。

3.3.1 スキーマ構造

SCIM では、管理対象のオブジェクトを「リソース」と呼び、オブジェクトが保持する個々の項目を「属性」と呼ぶ。

「属性」は Simple 属性か Complex 属性の2種類である。

Simple 属性は1つ以上の値(複数値)を持つことが可能である。

図 3.5: Simple 属性のイメージ(単一値)

図 3.6: Simple 属性のイメージ(複数値)

Complex 属性は1つ以上の Simple 属性をサブ属性として持つことが可能である。また、そのサブ属性をセットで持つことも可能である。

図 3.7: Complex 属性のイメージ

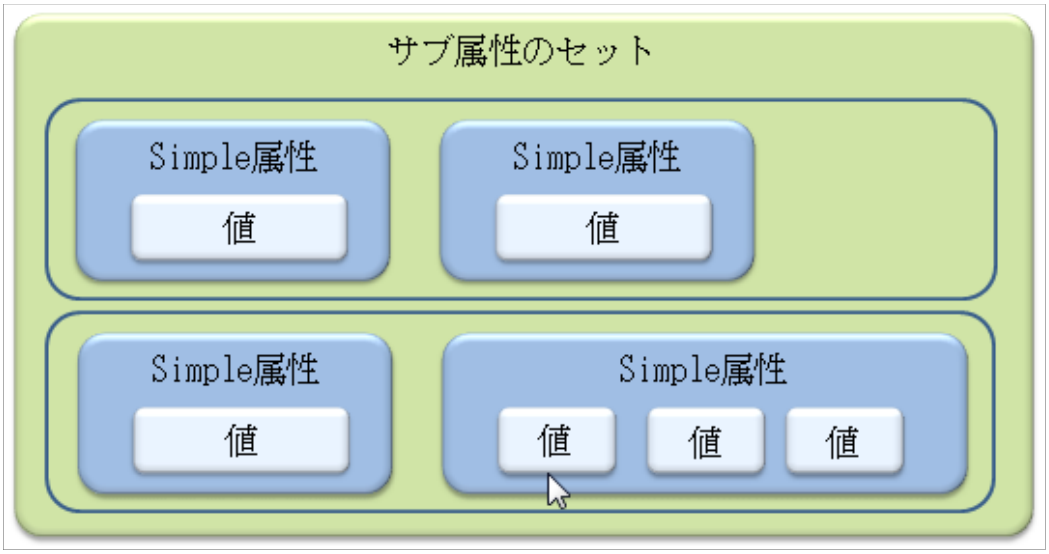


図 3.8: Complex 属性(サブ属性をセットで持つ場合)のイメージ

3.3.2 標準スキーマ仕様

3.3.2.1 共通属性

SCIM 標準仕様では、全てのリソースに含める共通属性が定義されている。

表 3.1: 共通属性の概要

属性名	説明
id	リソースを一意に識別する ID。
externalId	サービス利用企業側によって割り当てられる ID。
meta	Complex 型の属性。 メタ**データ**(作成日時・更新日時など)を格納する。
schemas	リソースに適用するスキーマの URI を格納する。

3.3.2.2 ユーザリソース、グループリソース

SCIM 標準仕様では、最小限のコアリソースとして、「ユーザリソース」と「グループリソース」を URI urn:scim:schemas:core:1.0 として定義している。

表 3.2: ユーザリソースのスキーマ定義概要

属性名	説明
userName	ユーザ名。
name	名前。Complex 型の属性。フルネーム・姓・名等を格納する。
displayName	表示名。
nickName	ニックネーム。
profileUrl	プロフィール URL。
title	タイトル。
userType	ユーザタイプ。
preferredLanguage	使用言語。
locale	地域。
timezone	タイムゾーン。
active	ユーザの何かしらの状態を表す真偽値。
password	パスワード。
emails	E メール。
phoneNumbers	電話番号。
ims	インスタントメッセージングアドレス。
photos	写真の URL。
addresses	住所。
groups	所属グループ。
entitlements	資格・権利。
roles	ロール。
x509Certificates	X.509 証明書。

表 3.3: グループリソースのスキーマ定義概要

属性名	説明
displayName	表示名。
members	所属メンバ。

3.3.2.3 エンタープライズユーザ拡張属性

SCIM 標準仕様には、ユーザ属性をエンタープライズ向けに拡張する"Enterprise User Schema Extension"という拡張仕様が存在し、URI urn:scim:schemas:extension:enterprise:1.0 として定義している。

表 3.4: Enterprise User Schema Extension で拡張される属性概要

属性名	説明
employeeNumber	従業員番号。
costCenter	コストセンター。(原価部門)
organization	組織。
division	組織内の部門。(企業によって意味は異なる)
department	組織内の部門。(企業によって意味は異なる)
manager	上司。

3.3.3 スキーマ拡張方法

SCIM スキーマは、LDAP で使用されるオブジェクトクラスに似た拡張モデルとなっている。しかし、オブジェクトクラスの継承モデルのような概念は無く、拡張は全て付加的に行うモデルとなっている。(補助オブジェクトクラスを付加していく考え方に似ている。)

スキーマ拡張を行う際には、標準で定義された属性の再定義を行ってはならない事に注意が必要である。

3.3.4 スキーマ定義表現

SCIM のスキーマ定義は、リソースごとにリソースの名前・URI・含める属性などを定義し、その属性はさらに、名前・型・単一値/複数値・説明・URI といった属性に関する定義を持つ。(Complex 型の属性の場合は、含めるサブ属性に関しても名前・型・単一値/複数値・説明などの定義を持つ。)

この定義情報は、/Schemas エンドポイントを参照する事で SCIM クライアントから参照可能である。

ユーザリソースのスキーマ定義の JSON 表現(抜粋)

```
{
  "id":"urn:scim:schemas:core:1.0:User",
  "name":"User",
  "description":"Core User",
  "schema":"urn:scim:schemas:core:1.0",
  "endpoint":"/Users",
  "attributes":[
    {
    "name":"id",
    "type":"string",
    "multiValued":false,
    "description":"Unique identifier for the SCIM resource ...",
    "schema":"urn:scim:schemas:core:1.0",
    "readOnly":true,
    "required":true,
    "caseExact":false
    },
    {
    "name":"name",
    "type":"complex",
    "multiValued":false,
    "description":"The components of the user's real name. ...",
    "schema":"urn:scim:schemas:core:1.0",
    "readOnly":false,
    "required":false,
    "caseExact":false,
    "subAttributes":[
    {
           "name":"formatted",
           "type":"string",
           "multiValued":false,
           "description":"The full name, ..." ,
           "readOnly":false,
           "required":false,
           "caseExact":false
    },
          ...
```

リソースに属性を追加する場合には、リソース定義内に追加する属性の定義を記載する。

ユーザリソースに employeeNumber 属性を記載した例

```
        ...
{
        "name":"employeeNumber",
        "type":"string",
        "multiValued":false,
        "description":"Numeric or alphanumeric identifier ..." ,
        "schema":"urn:scim:schemas:extension:enterprise:1.0",
        "readOnly":false,
        "required":false,
        "caseExact":false
},
        ...
```

3.4 SCIM プロトコル仕様

本節では、SCIM 標準プロトコルの仕様を説明する。

3.4.1 プロトコル概要

SCIM プロトコルは、HTTP 上でアイデンティティ情報を管理またはプロビジョニングするための RESTful なプロトコルである。

クラウドサービス事業者は、クラウドサービス利用企業(利用者)に対して、SCIM プロトコルでアクセスするための Base URL を公開する必要がある。この Base URL には、クエリ文字列を含んではならない事に注意が必要である。(SCIM クライアントが検索パラメタ等の付加的な情報を付与するために利用する可能性があるため。)

Base URL の例:https://example.com/scim/v1/

リソースに対する操作は、この Base URL 配下のリソースに応じたエンドポイントを基点として行う。

SCIM 標準では以下4つのエンドポイントを定義している。

表 3.5: SCIM 標準で定義されているエンドポイント

リソース	エンドポイント	備考

ユーザ	/Users	ユーザリソース操作用
グループ	/Groups	グループリソース操作用
サービスプロバイダ設定	/ServiceProviderConfigs	サーバの実装に関する情報参照用
スキーマ	/Schemas	スキーマリソース参照用

※オプションとして、一括操作を行うためのエンドポイントとして /Bulk も存在する。

3.4.2 認証・認可

SCIM プロトコルでは認証・認可に関する定義は無く、実装に応じて自由に選択して良いとされている。

プロトコル仕様では「OAuth 2.0 ベアラー・トークン」の利用が推奨されている。

(HTTP Basic 認証の利用も可能である。)

OAuth 2.0 Bearer Token を利用した HTTP ヘッダ例

```
GET /Users/12345678-1234-1234-1234-123456789012 HTTP/1.1
Host: example.com
Authorization: Bearer h480djs93hd8
```

3.4.3 操作

SCIM プロトコルでは、リソースに対する「追加」「取得(検索)」「更新」「削除」といった操作を提供しており、それぞれ HTTP メソッド(POST・GET・PUT(PATCH)・DELETE)に割り当てている。

結果は HTTP レスポンスコードで示し、詳細をレスポンスボディとして返却する。

エラー発生時には、以下のようなエラーを返却する。

表 3.6: エラーの例

レスポンスコード	操作	発生理由(例)
404 BAD REQUEST	GET,POST,PUT,PATCH,DELETE	リクエストを解釈不可/構文不正/スキーマ不正時
401	GET,POST,PUT,PATCH,DELETE	認証失敗時

UNAUTHORIZED		
403 FORBIDDEN	GET,POST,PUT,PATCH,DELETE	要求された操作をサーバがサポートしていない時
404 NOT FOUND	GET,PUT,PATCH,DELETE	指定されたリソースが存在しない時
409 CONFLICT	POST,PUT,PATCH,DELETE	指定されたバージョン番号がリソースの最新のバージョン番号と一致しない時(Bulk 時?)/リソースの重複時
412 PRECONDITION FAILED	PUT,PATCH,DELETE	IF-Match ヘッダで指定したバージョン番号がリソースの最新のバージョン番号と一致しない時
413 REQUEST ENTITY TOO LARGE	POST(Bulk)	最大操作数(maxOperations)超過時/最大ペイロードサイズ(maxPayload)超過時
500 INTERNAL SERVER ERROR	GET,POST,PUT,PATCH,DELETE	サーバ内部のエラー時
501 NOT IMPLEMENTED	GET,POST,PUT,PATCH,DELETE	要求された操作が未実装の時

エラー発生時のレスポンス例

```
HTTP/1.1 404 NOT FOUND

{
  "Errors":[
    {
    "description":"Resource 2819c223-7f76-453a-919d-413861904646 not found",
    "code":"404"
    }
  ]
}
```

以下に、各操作のリクエスト・レスポンスの例を記載する。

3.4.3.1 リソース追加操作

リソースを追加するには POST メソッドを使用する。

HTTP ヘッダで指定する URI は、追加するリソースのエンドポイント(/Users,/Groups など)を指定する必要がある。

HTTP ボディには、追加するリソースで利用するスキーマを schemas 属性で指定した上で、リソースに含める属性を指定する。

ユーザリソースの追加リクエスト例

```
POST /Users  HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Bearer h480djs93hd8
Content-Length: ...

{
  "schemas":["urn:scim:schemas:core:1.0"],
  "userName":"bjensen",
  "externalId":"bjensen",
  "name":{
    "formatted":"Ms. Barbara J Jensen III",
    "familyName":"Jensen",
    "givenName":"Barbara"
  }
}
```

リソースの追加に成功した場合には、HTTP レスポンスコード “201 Created” と共に、Location ヘッダに作成されたリソースの場所を示す URI を返却する。

ユーザリソースの追加レスポンス例

```
HTTP/1.1 201 Created
Content-Type: application/json
Location: https://example.com/v1/Users/2819c223-7f76-453a-919d-413861904646
ETag: W/"e180ee84f0671b1"
```

```
{
  "schemas":["urn:scim:schemas:core:1.0"],
  "id":"2819c223-7f76-453a-919d-413861904646",
  "externalId":"bjensen",
  "meta":{
    "resourceType":"User",
    "created":"2011-08-01T21:32:44.882Z",
    "lastModified":"2011-08-01T21:32:44.882Z",
    "location":"https://example.com/v1/Users/2819c223-7f76-453a-919d-413861904646",
    "version":"W¥/¥"e180ee84f0671b1¥""
  },
  "name":{
    "formatted":"Ms. Barbara J Jensen III",
    "familyName":"Jensen",
    "givenName":"Barbara"
  },
  "userName":"bjensen"
}
```

3.4.3.2 リソース取得操作

リソースの情報を取得するには GET メソッドを使用する。

URI にリソースの id を指定してリソースの情報を取得する方法と、URI に(クエリパラメタとして)検索フィルタを指定して検索結果を取得する方法(後述)がある。

以下はリソースの id を指定する場合の例。

ユーザリソースの取得リクエスト例

```
GET /Users/2819c223-7f76-453a-919d-413861904646
Host: example.com
Accept: application/json
Authorization: Bearer h480djs93hd8
```

ユーザリソースの取得レスポンス例

```
HTTP/1.1 200 OK
Content-Type: application/json
```

```
Location: https://example.com/v1/Users/2819c223-7f76-453a-919d-413861904646
ETag: W/"f250dd84f0671c3"

{
  "schemas":["urn:scim:schemas:core:1.0"],
  "id":"2819c223-7f76-453a-919d-413861904646,
  "externalId":"bjensen",
  "meta":{
    "resourceType":"User",
    "created":"2011-08-01T18:29:49.793Z",
    "lastModified":"2011-08-01T18:29:49.793Z",
    "location":"https://example.com/v1/Users/2819c223-7f76-453a-919d-413861904646",
    "version":"W¥/¥"f250dd84f0671c3¥""
  },
  "name":{
    "formatted":"Ms. Barbara J Jensen III",
    "familyName":"Jensen",
    "givenName":"Barbara"
  },
  "userName":"bjensen",
  "phoneNumbers":[
    {
      "value":"555-555-8377",
      "type":"work"
    }
  ],
  "emails":[
    {
      "value":"bjensen@example.com",
      "type":"work"
    }
  ]
}
```

URI に(クエリパラメタとして)返却属性を指定する事も可能である。この場合、パラメタ“attributes”の値として、属性名をカンマ区切りで指定する。

ユーザリソースの取得リクエスト例(返却属性を指定した場合)

```
GET /Users/2819c223-7f76-453a-919d-413861904646?attributes=userName
```

```
Host: example.com
Accept: application/json
Authorization: Bearer h480djs93hd8
```

ユーザリソースの取得レスポンス例(返却属性を指定した場合)

```
HTTP/1.1 200 OK
Content-Type: application/json
Location: https://example.com/v1/Users/2819c223-7f76-453a-919d-413861904646
ETag: W/"f250dd84f0671c3"

{
  "schemas":["urn:scim:schemas:core:1.0"],
  "id":"2819c223-7f76-453a-919d-413861904646,
  "meta":{
    "resourceType":"User",
    "created":"2011-08-01T18:29:49.793Z",
    "lastModified":"2011-08-01T18:29:49.793Z",
    "location":"https://example.com/v1/Users/2819c223-7f76-453a-919d-413861904646",
    "version":"W¥/¥"f250dd84f0671c3¥""
  },
  "userName":"bjensen"
}
```

3.4.3.3 リソース検索操作

GET メソッドの URI として、リソースのエンドポイントと共にクエリパラメタを指定する事で、一覧取得・フィルタ検索・ソート・ページングによるリソースの検索が可能である。(フィルタ検索・ソート・ページングの実装は必須ではなくオプション)

ユーザリソースの検索リクエスト例(userName 属性の一覧を取得)

```
GET /Users?attributes=userName
Host: example.com
Accept: application/json
Authorization: Bearer h480djs93hd8
```

ユーザリソースの検索レスポンス例(userName 属性の一覧を取得)

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "totalResults":2,
  "schemas":["urn:scim:schemas:core:1.0"],
  "Resources":[
    {
      "userName":"bjensen"
    },
    {
      "userName":"jsmith"
    }
  ]
}
```

フィルタ検索を行う場合には、パラメタ“filter”の値としてフィルタ式を指定する。

フィルタ式の例

```
filter=userName eq "bjensen"
filter=name.familyName co "O'Malley"
filter=userName sw "J"
filter=title pr
filter=meta.lastModified gt "2011-05-13T04:42:34Z"
filter=meta.lastModified ge "2011-05-13T04:42:34Z"
filter=meta.lastModified lt "2011-05-13T04:42:34Z"
filter=meta.lastModified le "2011-05-13T04:42:34Z"
filter=title pr and userType eq "Employee"
filter=title pr or userType eq "Intern"
filter=userType eq "Employee" and (emails co "example.com" or emailsco "example.org")
```

※実際には空白やマルチバイト文字はエンコードしてURIに指定する

ユーザリソースの検索リクエスト例(userNameがbjensenのリソースを検索)

```
GET /Users?filter=userName%20eq%20%22bjensen%22
Host: example.com
Accept: application/json
Authorization: Bearer h480djs93hd8
```

検索結果をソートしたい場合には、パラメタ “sortBy” の値にソートキーとして使用する属性名を指定し、パラメタ “sortOrder” の値にソート順(″ascending″ または ″descending″)を指定する。”sortOrder” の指定が無い場合は “ascending” としてソートする。

ユーザリソースの検索リクエスト例(userName で昇順ソート)

```
GET /Users?sortBy=userName&sortOrder=ascending
Host: example.com
Accept: application/json
Authorization: Bearer h480djs93hd8
```

検索結果をページ単位で取得したい場合には、パラメタ “startIndex” の値にインデックス(1 から始まるページ番号)を指定し、パラメタ “count” の値に 1 ページあたりの結果数を指定する。

ユーザリソースの検索リクエスト例(1 ページあたり 10 件で 1 ページ目を取得)

```
GET /Users?startIndex=1&count=10
Host: example.com
Accept: application/json
Authorization: Bearer h480djs93hd8
```

3.4.3.4 リソース更新操作

リソースを更新するには PUT メソッドまたは PATCH メソッドを使用する。

PUT はリソースの全属性(readOnly 属性は除く)を更新する(指定が無い属性は削除される)操作であり、PATCH はリソースの任意の属性を更新する(指定が無い属性は更新されない)操作である。

ユーザリソースの更新(PUT)リクエスト例

```
PUT /Users/2819c223-7f76-453a-919d-413861904646
Host: example.com
Accept: application/json
```

```
Content-Type: application/json
Authorization: Bearer h480djs93hd8
If-Match: W/"a330bc54f0671c9"

{
  "schemas":["urn:scim:schemas:core:1.0"],
  "id":"2819c223-7f76-453a-919d-413861904646",
  "userName":"bjensen",
  "externalId":"bjensen",
  "name":{
    "formatted":"Ms. Barbara J Jensen III",
    "familyName":"Jensen",
    "givenName":"Barbara",
    "middleName":"Jane"
  },
  "emails":[
    {
      "value":"bjensen@example.com"
    },
    {
      "value":"babs@jensen.org"
    }
  ]
}
```

ユーザリソースの更新(PUT)レスポンス例

```
HTTP/1.1 200 OK
Content-Type: application/json
ETag: W/"b431af54f0671a2"
Location:"https://example.com/v1/Users/2819c223-7f76-453a-919d-413861904646"
{
  "schemas":["urn:scim:schemas:core:1.0"],
  "id":"2819c223-7f76-453a-919d-413861904646",
  "userName":"bjensen",
  "externalId":"bjensen",
  "name":{
    "formatted":"Ms. Barbara J Jensen III",
    "familyName":"Jensen",
    "givenName":"Barbara",
    "middleName":"Jane"
  },
  "emails":[
```

```
    {
      "value":"bjensen@example.com"
    },
    {
      "value":"babs@jensen.org"
    }
  ],
  "meta": {
    "resourceType","User",
    "created":"2011-08-08T04:56:22Z",
    "lastModified":"2011-08-08T08:00:12Z",
    "location":"https://example.com/v1/Users/2819c223-7f76-453a-919d-413861904646",
    "version":"W¥/¥"b431af54f0671a2¥""
  }
}
```

ユーザリソースの更新(PATCH)リクエスト例

```
PATCH /Users/2819c223-7f76-453a-919d-413861904646
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Bearer h480djs93hd8
If-Match: W/"a330bc54f0671c9"

{
  "schemas": ["urn:scim:schemas:core:1.0"],
  "emails": [
    {
      "value": "bjensen@example.com",
      "primary": true
    }
  ]
}
```

ユーザリソースの更新(PATCH)レスポンス例

```
HTTP/1.1 200 OK
Content-Type: application/json
ETag: W/"b431af54f0671a2"
Location:"https://example.com/v1/Users/2819c223-7f76-453a-919d-413861904646"
{
```

```
  "schemas":["urn:scim:schemas:core:1.0"],
  "id":"2819c223-7f76-453a-919d-413861904646",
  "userName":"bjensen",
  "externalId":"bjensen",
  "name":{
    "formatted":"Ms. Barbara J Jensen III",
    "familyName":"Jensen",
    "givenName":"Barbara",
    "middleName":"Jane"
  },
  "emails":[
    {
      "value":"bjensen@example.com",
      "primary": true
    },
    {
      "value":"babs@jensen.org"
    }
  ],
  "meta": {
    "resourceType","User",
    "created":"2011-08-08T04:56:22Z",
    "lastModified":"2011-08-08T08:00:12Z",
    "location":"https://example.com/v1/Users/2819c223-7f76-453a-919d-413861904646",
    "version":"W¥/¥"b431af54f0671a2¥""
  }
}
```

3.4.3.5 リソース削除操作

リソースを削除するには DELETE メソッドを使用する。

URI に削除対象のリソースの id を指定する。

ユーザリソースの削除リクエスト例

```
DELETE /Users/2819c223-7f76-453a-919d-413861904646
Host: example.com
Authorization: Bearer h480djs93hd8
If-Match: W/"c310cd84f0281b7"
```

ユーザリソースの削除レスポンス例

```
HTTP/1.1 200 OK
```

3.4.3.6 一括(Bulk)操作

一括(Bulk)操作は、複数の SCIM 操作を1つの POST リクエストに含め、複数の操作の結果を1つのレスポンスに含めて返却する操作である。

URI に一括(Bulk)操作用のエンドポイントを指定する。リクエストボディには、一括操作を終了するエラー数を指定する“failOnErrors”属性と、複数の SCIM 操作を指定する“Operations”属性を指定する。

一括操作リクエスト例

```
POST /v1/Bulk
Host: example.com
Accept: application/json
Authorization: Bearer h480djs93hd8
Content-Length: …

{
  "schemas":[
    "urn:scim:schemas:core:1.0"
  ],
  "failOnErrors":1,
  "Operations":[
    {
      "method":"POST",
      "path":"/Users",
      "bulkId":"qwerty",
      "data":{
        "schemas":[
          "urn:scim:schemas:core:1.0"
        ],
        "userName":"Alice"
      }
    },
    {
      "method":"PUT",
      "path":"/Users/b7c14771-226c-4d05-8860-134711653041",
      "version":"W¥/¥"3694e05e9dff591¥"",
```

```
      "data":{
        "schemas":[
          "urn:scim:schemas:core:1.0"
        ],
        "id":"b7c14771-226c-4d05-8860-134711653041",
        "userName":"Bob"
      }
        },
        {
      "method":"PATCH",
      "path":"/Users/5d8d29d3-342c-4b5f-8683-a3cb6763ffcc",
      "version":"W¥/¥"edac3253e2c0ef2¥"",
      "data":{
        "schemas":[
          "urn:scim:schemas:core:1.0"
        ],
        "id":"5d8d29d3-342c-4b5f-8683-a3cb6763ffcc",
        "userName":"Dave",
        "meta":{
          "attributes":[
            "nickName"
          ]
        }
      }
        },
        {
      "method":"DELETE",
      "path":"/Users/e9025315-6bea-44e1-899c-1e07454e468b",
      "version":"W¥/¥"0ee8add0a938e1a¥""
        }
  ]
}
```

一括操作レスポンス例

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "schemas": [
    "urn:scim:schemas:core:1.0"
  ],
  "Operations": [
```

```
    {
      "location": "https://example.com/v1/Users/92b725cd-9465-4e7d-8c16-01f8e146b87a",
      "method": "POST",
      "bulkId": "qwerty",
      "version": "W¥/¥"oY4m4wn58tkVjJxK¥"",
      "status": {
        "code": "201"
      }
    },
    {
      "location": "https://example.com/v1/Users/b7c14771-226c-4d05-8860-134711653041",
      "method": "PUT",
      "version": "W¥/¥"huJj29dMNgu3WXPD¥"",
      "status": {
        "code": "200"
      }
    },
    {
      "location": "https://example.com/v1/Users/5d8d29d3-342c-4b5f-8683-a3cb6763ffcc",
      "method": "PATCH",
      "version": "W¥/¥"huJj29dMNgu3WXPD¥"",
      "status": {
        "code": "200"
      }
    },
    {
      "location": "https://example.com/v1/Users/e9025315-6bea-44e1-899c-1e07454e468b",
      "method": "DELETE",
      "status": {
        "code": "200"
      }
    }
  ]
}
```

4. フェデレーションとアイデンティティ・プロビジョニング標準プロトコルの日本エンタープライズ IT への適用

本章では、2～3 章で解説したクラウドサービスに対するフェデレーション標準プロトコル＝OpenID Connect とアイデンティティ・プロビジョニング標準プロトコル＝SCIM を、日本のエンタープライズ IT に適用する場合、どのようにすれば良いかを説明する。

OpenID Connect はコンシューマ IT での実績を積み上げてきたが、標準化作業が進む SCIM とともにエンタープライズ IT に適用しようとすると、コンシューマ IT とは異なるエンタープライズ IT ならではの新たな要件への対応が必要となり、従来の技術仕様をそのまま適用することでは対応できない点も出てくる。また、日本のエンタープライズ IT においては、欧米とは異なる要件への対応も必要となる。

そこで、まず 4.1 節では、OpenID Connect と SCIM を日本のエンタープライズ IT に適用する場合に考慮しなければならない、アイデンティティ情報と認証システムの特性を整理する。そして 4.2 節では、OpenID Connect を日本のエンタープライズ IT に適用する場合の具体策を、4.3 節では、SCIM を日本のエンタープライズ IT に適用する場合の具体策をそれぞれ、ガイドラインとして示す。

4.1 日本のエンタープライズ IT におけるアイデンティティ情報と認証システムの特性

クラウドサービスに対するフェデレーションとアイデンティティ・プロビジョニングの標準プロトコルを、エンタープライズ IT に適用する場合、コンシューマ IT の場合とは何が異なるのか。それは、アイデンティティ情報と認証システムの位置付けである。エンタープライズ IT に適用する場合には、アイデンティティ情報と認証システムの特性として、以下の 3 点を考慮しなければならない。

1. アイデンティティ情報の管理主体が企業であること
2. クラウドサービスの利用者が企業であること
3. アイデンティティ情報と認証システムの基盤が企業内（イントラネット内）に存在すること

また、日本のエンタープライズ IT に適用する場合には、欧米とは異なるアイデンティティ情報の特性への対応も必要になる。

4. アイデンティティ情報の管理対象として日本特有の属性が存在すること

以下の各節にて、これらを詳しく説明する。

4.1.1 アイデンティティ情報の管理主体が企業であること

エンタープライズ IT におけるクラウドサービス利用がコンシューマ IT と異なる 1 点目として、クラウドサービス事業者との利用契約の主体が個人ではなく企業であり、クラウドサービスで利用するデータの管理主体が個人ではなく企業であることが挙げられる。それゆえ、クラウドサービス利用者である従業員個人のアイデンティティ情報の管理主体も企業である。

この特性から、エンタープライズ IT におけるアイデンティティ情報の取り扱いは、以下に挙げる点でコンシューマ IT とは異なる。

〔企業のクラウドサービス利用契約に応じたプロビジョニングとフェデレーション〕

コンシューマ IT では、個人が自らのアイデンティティ情報をクラウドサービスに対して GUI により登録・管理する。

一方、エンタープライズ IT では、クラウドサービスを利用する企業(主に情報システム部門)が、利用契約に応じたライセンス数の範囲の中で、従業員個人のアイデンティティ情報をクラウドサービスに対して登録・管理する。

エンタープライズ IT の場合、フェデレーションにおいて用いるユーザ識別子は、このアイデンティティ情報のプロビジョニングで事前に割り当てることが多い。

クラウドサービス利用企業の中でクラウドサービスを利用することが認められた従業員は、企業を介して貸与されたユーザ識別子を用いて、フェデレーションによるクラウドサービスに対するユーザ認証とアイデンティティ情報の連携を行い、クラウドサービスを利用する。

〔クラウドサービスへのアイデンティティ情報の提供を必要最小限に絞るためのプロビジョニングとフェデレーションの併用〕

クラウドサービスの導入を考えている企業の懸念点の一つが、セキュリティである。これまで企業内にあった従業員アイデンティティ情報の一部をクラウド上に置くことになるからである。

セキュリティ対策の一つとして、クラウド上に置くアイデンティティ情報を必要最小限に絞るためには、クラウドサービス側で必要とされるアイデンティティ情報(ユーザ属性情報)をそれが必要になったときに逐次クラウドサービス側へ提供すること(＝Just In Time プロビジョニング)、つまりフェデレーションが有効である。しかし、それによるクラウド利用企業側のシステムの複雑化、コストの増大は極力避けなければならない。両者のバランスを検討し最適な解を見つける必要がある。

〔企業、クラウドサービス事業者、従業員の三者間の関係で考慮すべき内容〕

コンシューマ IT の場合、アイデンティティ情報の取り扱い（漏えい対策を含む）は、それを提供する個人と提供を受けるクラウドサービス事業者が共に責任をもって行うことが多い。

一方、エンタープライズ IT の場合、アイデンティティ情報の管理主体が企業であるため、アイデンティティ情報の取り扱いにおいては、企業～クラウドサービス事業者間、企業～従業員間、従業員～クラウドサービス事業者間それぞれの関係性を考慮する必要がある。

企業～クラウドサービス事業者間には、企業がクラウドサービスを利用するために必要な、企業内の多数のアイデンティティ情報（従業員属性情報）をクラウドサービスに対して提供するという関係が存在する。また、利用者である従業員がクラウドサービス上でアクセスするデータは企業の業務情報であり、企業秘密情報であることが多い。

そのため、クラウドサービス事業者側は、コンシューマ個人向けサービスとは異なる視点やレベルのリスク対策（認証強度を高める等）を講じる必要がある。もちろん、相手が個人の場合に発生するプライバシー侵害等の問題を軽視することはできないが、エンタープライズ向けクラウドサービスの企業秘密情報漏えいの場合の損害賠償などに係わるコストは、個人の場合に比較して莫大なものになる可能性は否定できないため、契約条項等は十分に練り上げる必要がある。

企業～従業員間には雇用契約が存在し、企業が統制上策定したルールやポリシを従業員は遵守しなければならないという関係が存在する。企業は業務にかかわるクラウドサービス利用に対しても例外なくルールやポリシを適用していく必要があり、従業員もそれを強く意識しておく必要がある。

よって従業員は、クラウドサービスの業務利用におけるアイデンティティ情報の取り扱いに際しても、企業のルールやポリシに従い適正にこれを管理しなければならない。

一方で、企業は従業員個人のアイデンティティ情報の取り扱いに際して、個人のプライバシー保護の観点からも留意を要する。EU など個人情報の取り扱いに特段の注意を払うことが求められる場合には、企業は個人情報取得・取り扱いに関する従業員本人からの事前同意や、従業員からの要請による個人情報の開示や不要になった個人情報の削除といった対応が必要になる。

従業員～クラウドサービス事業者間においては、クラウドサービス事業者は企業から提供を受けたアイデンティティ情報に従業員にとっての個人情報としての取り扱いを求められる情報が含まれていることを留意する必要がある。また、クラウドサービスの内容による個別の必要性に応じて、例えば、従業員個人のクレジットカード番号など、企業が管理していない個人情報のやり取りが発生する場合もある。

そのため、アイデンティティ情報の管理主体が企業であったとしても、クラウドサービス事業者は、個人のプライバシー保護の観点からコンシューマ IT 向けと同様の視点での個人情報取り扱いに関する考慮や対応が求められる。

〔アイデンティティのライフサイクル運用の源泉は人事システム〕

企業では、アイデンティティのライフサイクル上の運用が、組織改編、人事異動、職務変更、勤務地変更などの企業における組織と人の変動に応じて発生する。このうち人事異動には、入社、退社、昇格、所属変更、休職、復職、出向、復帰といったイベントが存在する。

このような企業における組織と人の変動に応じて、従業員個人に対する適切なアクセス権限の付与を維持管理するためには、フェデレーションやプロビジョニングで利用するアイデンティティ情報（いわゆるメタディレクトリに格納しているアイデンティティ情報）の適切なメンテナンスを ID 管理システム等により継続的に行うとともに、その源泉情報である人事データの適切なメンテナンスが継続的に行われていることが前提条件となる。つまり、企業としてのアイデンティティ情報が適切な状態に保たれるためには、アイデンティティのライフサイクル管理に関するポリシーやシステムが必要だが、それは IT 部門だけではなく人事部門の運用管理も含めて定義、構築される必要がある。

〔定期的な組織改編や一斉の人事異動への対応が必要〕

日本の企業では、定期的な組織改編や一斉の人事異動に対応したアイデンティティ・プロビジョニングが求められる。

定期的な組織改編や一斉の人事異動では、大量の更新処理が必要になる場合が多く、深夜の限られたメンテナンス時間帯の中で大量の更新を高速に処理できることが求められる。

また、発令日に先立って更新データを事前に入力しておき、発令日になると更新データが有効になるような仕組みや、発令後も当分は発令前の状態を併存させ異動猶予期間を設けるような仕組みなどが求められる場合もある。

4.1.2 クラウドサービスの利用者が企業であること

エンタープライズ IT におけるクラウドサービス利用がコンシューマ IT と異なる 2 点目として、クラウドサービス

の利用者が個人ではなく企業であることが挙げられる。

利用者が個人の場合、クラウドサービスに対するアイデンティティ情報の登録・管理は個人ひとり分のみで済むが、利用者が企業の場合、企業の組織に関する情報や、多数の従業員全員分のアイデンティティ情報の登録・管理が必要である。

また、企業が業務目的で利用するクラウドサービスは、スケジュール共有サービスや営業支援サービスのように複数の従業員の間で情報を共有するために、従業員個人を互いに識別する必要があり、認証してサービスを利用する従業員以外の従業員情報が予め必要となる。

さらに、従業員それぞれの業務上の権限に応じてサービスやデータに対するアクセス制御を実施する必要がある。そのため、エンタープライズ系システムは個人の識別に必要な属性情報（氏名、所属、メールアドレス等）やアクセス制御に必要な従業員個人の属性情報（所属、役職、ロール等）が、認証処理を実施する前に整備されていることが前提のロジックとなっている場合が多い。

この特性から、クラウドサービス利用企業はクラウドサービスに対して、サービス利用開始前にこれらのアイデンティティ情報をプロビジョニングしておく必要がある。

次にエンタープライズ IT において、プロビジョニングするアイデンティティ情報の内容について特徴的な点を説明する。

〔プロビジョニングの対象は企業の組織とそれに所属する多数の従業員である〕

エンタープライズ IT におけるクラウドサービスの場合、企業の組織そのものと組織に所属する多数の個人をアイデンティティ・プロビジョニングの対象とすることが求められる。

組織そのものの情報としては、組織コード、組織名称などに加えて、組織に所属する従業員をメンバとする各種グループ情報も求められる場合がある。

また、個人のアイデンティティ属性としては、社員番号、所属組織、役職、各種コードなど、組織人としての属性が求められる。

〔アイデンティティ情報を管理するために、多数の組織、グループ、個人の中から複雑な条件を指定して抽出する検索が必要〕

企業に存在する多数の組織、グループ、個人の中から、複雑な条件を指定して抽出する検索ができることが求められる。複雑な条件とは、組織、グループ、個人のアイデンティティ属性値の有無、同値、含有、順序に関する条件を論理演算（論理積、論理和、論理否定）により組み合わせたものである。

アイデンティティ・プロビジョニングの対象が多数存在するがゆえに、クラウドサービスに対するアイデンティティ・プロビジョニングの状態を様々な切り口から絞り込んで確認するために、このような検索機能が求められる。

4.1.3 アイデンティティ情報と認証システムの基盤が企業内（イントラネット内）に存在すること

エンタープライズ IT におけるクラウドサービス利用がコンシューマ IT と異なる 3 点目として、アイデンティティ情報と認証システムの基盤が企業内（イントラネット内）に存在することが挙げられる。

コンシューマ IT の場合、利用者個人が個人端末からクラウドサービスへアクセスする際の通信は、アイデンティティ・プロバイダとやり取りするための通信を含め、すべてインターネット上でやり取りされる。

一方、エンタープライズ IT の場合、利用者である従業員個人が企業端末からクラウドサービスへアクセスする際の通信は、企業内（イントラネット内）のネットワークからインターネットに接続され、やり取りされる。

（モバイルアクセスにおいても企業内（イントラネット内）に存在する ID 管理・認証システムの基盤を介してやり取りされる場合が多い。）

そして、この企業内(イントラネット内)に存在する ID 管理・認証システムとクラウドサービスの間にはファイアウォール等のネットワーク境界が存在し、データ通信の出入りが制限されている。特に外部（インターネット）側から企業内（イントラネット内）のデータやシステムへアクセスすることは、厳しく制限されているのが通常である。

〔利用企業側のファイアウォールによる制限のため、クラウドサービス側からアイデンティティ情報と認証システムの基盤へアクセスできない環境に対する対応が必要〕

このことは、クラウドサービスに対するフェデレーションを実現するためのプロトコルである OpenID Connect にとっては、クラウドサービス（RP）側から ID 管理・認証システム（OP）側へアクセスするためのプロトコルにおいて、上記制限に対する対応方法の検討が必要になることを意味する。

OpenID Connect をエンタープライズ IT におけるクラウドサービス利用に対して適用する場合、クラウドサービス（RP）側が ID トークンをはじめとするユーザ認証・認可データを要求し、それに対してクラウドサービス利用企業側が OpenID プロバイダ（OP）として応答するという構成を取るため、上記制限を回避する必要がある。RP から OP への直接的な通信が制限される前提で OpenID Connect の適用方法を設計することが重要なポイントになる。

詳しくは 4.2 節で説明する。

4.1.4 アイデンティティ情報の管理対象として日本特有の属性が存在すること

日本のエンタープライズ IT におけるクラウドサービス利用が欧米とは異なる点として挙げられるのは、アイデ

ンティティ情報の管理対象として日本特有の属性が存在することである。

日本のエンタープライズ IT の場合、組織階層の深さや実体を伴う兼務所属の多さ、ならびに多様な言語表現といった点で、欧米とは異なる。

この特性から、日本のエンタープライズ IT におけるアイデンティティ情報の取り扱いは、以下に挙げる点で欧米とは異なる。

〔深い組織階層や多数の兼務所属を扱えることが必要〕

日本の企業では、アイデンティティ・プロビジョニングの対象として、深い組織階層や多数の兼務所属を扱えることが求められる。

また、同一の個人であっても兼務所属ごとに役職や職務が異なり、それに応じて権限が異なる場合もある。

〔アイデンティティ属性の多様な言語表現が必要〕

日本では、アイデンティティ属性の多様な言語表現が求められる。例えば、組織、グループ、個人の名称や、役職、所在地などの文字列属性は、漢字、仮名（ひらがな、カタカナ）、ローマ字といった複数の文字や言語で表現した属性値を同時に保持できることが求められる。

また、現代のグローバル化したインターネット環境では、それぞれのローカルに応じた文字／言語表現が求められる場合もある。

4.2 OpenID Connect のエンタープライズ IT 適用ガイド

本節では、OpenID Connect をエンタープライズ IT に適用する場合に、どのように適用すれば良いかのガイドラインを示す。

OpenID Connect を用いて、エンタープライズ IT におけるクラウドサービスに対するフェデレーションを実現しようとすると、先に述べたとおり、コンシューマ IT とは異なるエンタープライズ IT ならではの要件への対応が必要となる。

本節では、まず、「4.1 日本のエンタープライズ IT におけるアイデンティティ情報と認証システムの特性」の内容を受け、日本のエンタープライズ IT におけるクラウドサービスに対するフェデレーションの要件をあらためて示す。次に、OpenID Connect 仕様をどのように利用すれば良いかのガイドラインを示す。

4.2.1 エンタープライズ IT におけるフェデレーション要件

「4.1 日本のエンタープライズ IT におけるアイデンティティ情報と認証システムの特性」で整理した日本のエン

タープライズ IT におけるアイデンティティ情報と認証システムの特性から、日本のエンタープライズ IT におけるクラウドサービスに対するフェデレーションの要件をあらためて抽出すると、以下に挙げる要件が存在する。

- 利用企業側のファイアウォールによる制限のため、クラウドサービス側からアイデンティティ情報と認証システムの基盤へアクセスできない環境に対する対応が必要

上記の要件はコンシューマ IT とは異なる要件である。

4.2.2 OpenID Connect の適用方法

OpenID Connect 仕様を具体的にどのように利用すれば良いかのガイドラインを示す。

OpenID Connect を利用するためには、以下の事項について検討し、クラウドサービス利用企業(OP)とクラウドサービス事業者(RP)からなる認証システムを実装する必要がある。

- 処理フローの選択
- ID トークンに含めるユーザ識別子の検討
- クラウドサービス事業者(RP)がクラウドサービス利用企業(OP)に要求する属性情報(クレーム)の検討
- サービス利用者向けインタフェース(ログオン画面)の設計
- 再認証処理の実現

次節以降で、上記の検討事項について解説する。

なお、「4.1 日本のエンタープライズ IT におけるアイデンティティ情報と認証システムの特性」で整理し前節であらためて抽出した、日本のエンタープライズ IT におけるクラウドサービスに対するフェデレーションの要件への対応方法については、次節「フローの選択」に示してある。

4.2.3 フローの選択

OpenID Connect を利用してフェデレーションを実現する場合、クラウドサービス事業者(RP)はクラウドサービス利用企業(OP)から、ID トークンとアクセストークンを受け取る。ID トークンにはサービス利用者のユーザ識別子が含まれており、クラウドサービス事業者(RP)側で利用者の特定に利用する。アクセストークンはクラウドサービス利用者の属性情報(クレーム。氏名、所属グループ、所属部署、連絡先など)を、UserInfo エンドポイントから取得するために利用する。

クラウドサービス利用企業(OP)から ID トークンとアクセストークンを受け取る処理のフローには、「2.2 OpenID Connect によるフェデレーション」で解説した以下の 2 種類のものがある。

〔認可コードフロー〕

クラウドサービス事業者(RP)はクラウドサービス利用企業(OP)に直接 HTTPS でアクセスして ID トークンとアクセストークンを取得する。クラウドサービス利用企業(OP)の認証システム(全エンドポイント)がインターネットからアクセス可能な DMZ などのネットワークに配置されている必要がある。

〔Implicit フロー〕

クラウドサービス事業者(RP)は、サービス利用者のブラウザを経由して(ブラウザのリダイレクトを利用)クラウドサービス利用企業(OP)から ID トークンとアクセストークンを取得する。

クラウドサービス利用企業(OP)の認証システムの一部、または全てが企業のファイアウォール内などインターネットからアクセス不可能なネットワークに配置されている場合にこのフローを利用する。

クラウドサービス利用企業(OP)のネットワーク構成などに応じて利用するフローを選択する。ここでは、フローの採用基準を示す。各フローの詳細は「2.2 OpenID Connect によるフェデレーション」や OpenID Connect の仕様本文を参照して頂きたい。なお、クラウドサービス利用企業(OP)の Authorization エンドポイントにはクラウドサービス利用者(のブラウザ)からアクセス可能であるものとする。

4.2.3.1 クラウドサービス利用企業の OP に対してインターネットからアクセス可能な場合

認可コードフローを採用する。コンシューマ環境に近い構成である。フローのシーケンスは以下の図のようになる。

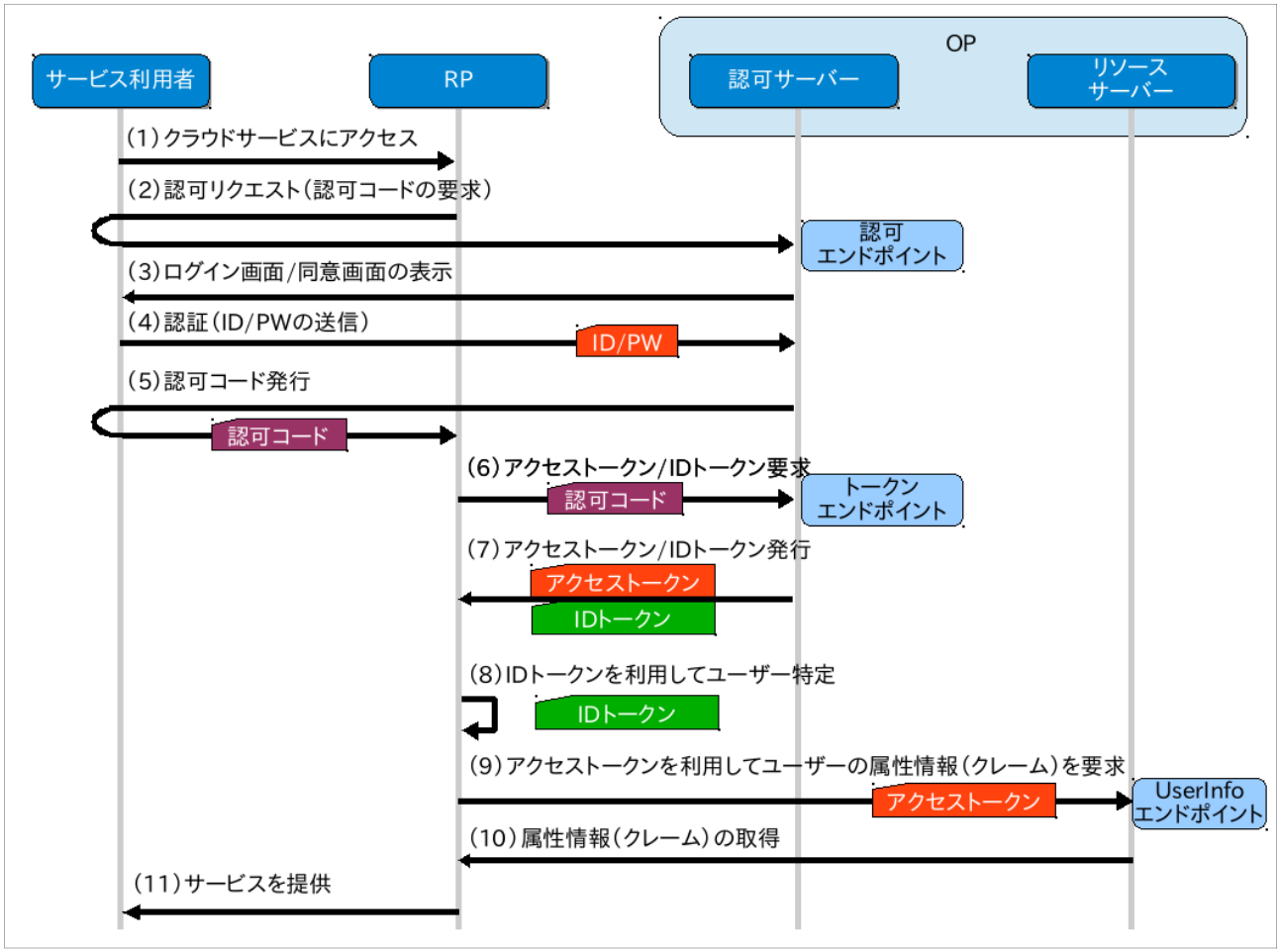


図 4.1: OpenID Connect 認可コードフロー

各リクエストとレスポンスについて解説する。

〔認可リクエスト〕

クラウドサービス事業者(RP)はクラウドサービス利用企業(OP)の Authorization エンドポイントに対して認可コードを要求する。リクエストの例を以下に示す。この例では、ブラウザのリダイレクト(HTTP ステータスコード 302)を利用してリクエストを送信することを想定している。

```
HTTP/1.1 302 Found
 Location: https://op.example.co.jp/authorize?
   response_type=code
   &client_id=jdDFd8fdl
```

```
  &redirect_uri=https%3A%2F%2Frp.example.jp%2Fcb
  &scope=openid%20profile%email
  &state=3ce334d8ka0
```

「response_type=code」が認可コードの要求を意味する。アクセストークンを利用して UserInfo エンドポイントから取得したい情報を scope パラメタにより指定する。レスポンスの例を以下に示す。

```
HTTP/1.1 302 Found
Location: https://rp.example.jp/cb?
  code=SplxlOBeZQQYbYS6WxSbIA
  &state=3ce334d8ka0
```

クラウドサービス事業者(RP)は、レスポンスから認可コード(code)を抽出し、アクセストークン取得リクエストを作成する。

〔アクセストークン取得リクエスト〕

クラウドサービス事業者(RP)はクラウドサービス利用企業(OP)の Token Endpoint に対してアクセストークンを要求する。リクエストの例を以下に示す。

```
POST /token HTTP/1.1
Host: op.example.co.jp
Content-Type: application/x-www-form-urlencoded
Authorization: Basic czZCaGRSa3FOMzpnWDFmQmFOM2JW

grant_type=authorization_code&code=SplxlOBeZQQYbYS6WxSbIA
  &redirect_uri=https%3A%2F%2Frp.example.jp%2Fcb
```

この例では HTTP POST メソッドを利用し、RP のクライアント認証の方法として HTTP Basic 認証を利用している。クラウドサービス事業者(RP)のクライアント ID とクライアント・シークレットは Authorization ヘッダに含まれる。レスポンスの例を以下に示す。

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache

{
 "access_token":"SlAV32hkKG",
 "token_type":"Bearer",
 "expires_in":3600,
 "refresh_token":"tGzv3JOkF0XG5Qx2TlKWIA",
 "id_token":"eyJ0 ... NiJ9.eyJ1c ... I6IjIifX0.DeWt4Qu ... ZXso"
}
```

クラウドサービス事業者(RP)は、レスポンスの ID トークン(id_token)からユーザ識別子を抽出してサービス利用者の特定を行う。また、アクセストークン(access_token)を利用して UserInfo エンドポイントにユーザ情報取得リクエストを送信し、サービス利用者の属性情報を取得する。

〔ユーザ情報取得リクエスト〕

クラウドサービス事業者(RP)はクラウドサービス利用企業(OP)の UserInfo エンドポイントに対してユーザの属性情報を要求する。リクエストの例を以下に示す。

```
GET /userinfo HTTP/1.1
Host: op.example.co.jp
Authorization: Bearer SlAV32hkKG
```

Token Endpoint から取得したアクセストークンを、Authorization ヘッダの値として「Bearer アクセストークン」という形式で指定する。レスポンスの例を以下に示す。

```
HTTP/1.1 200 OK
Content-Type: application/json

{
 "sub": "248289761001",
 "name": "NIPPON Taro",
 "given_name": "Taro",
```

```
  "family_name": "NIPPON",
  "email": "taro@op.example.co.jp"
}
```

HTTP のレスポンスボディに、属性情報が JSON 形式で入っている。

4.2.3.2 クラウドサービス利用企業の OP に対してインターネットからアクセス不可能な場合

クラウドサービス事業者(RP)はクラウドサービス利用企業(OP)の Authorization エンドポイントにしかアクセスできないため、Implicit フローを採用する。フローのシーケンスは以下の図のようになる。

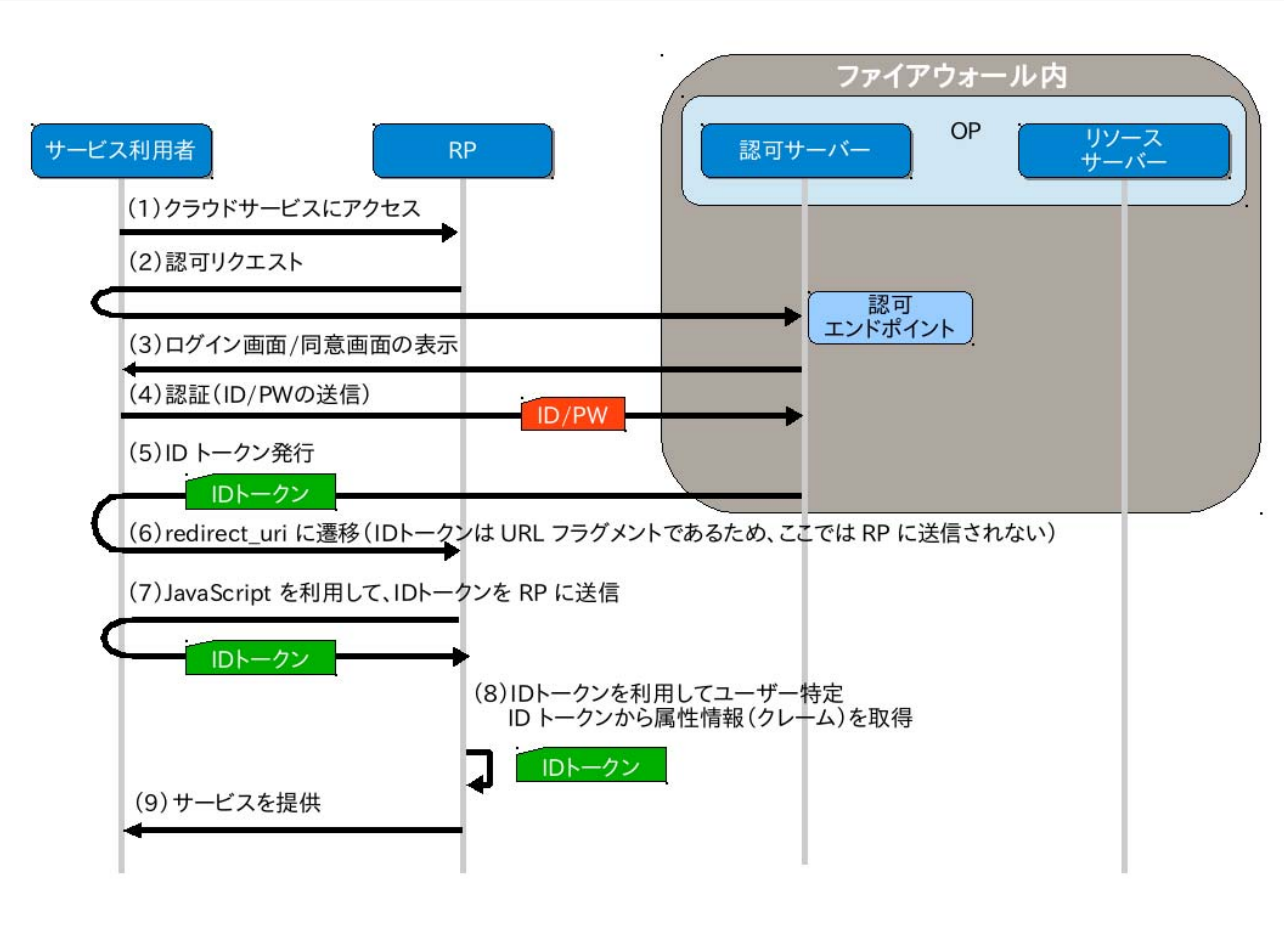


図 4.2: OpenID Connect Implicit フロー

UserInfo エンドポイントからの属性情報取得が不可能であるため、ユーザ識別子に加えて他の属性情報も必

要な場合は ID トークンに格納してクラウドサービス事業者(RP)に渡す。

各リクエストとレスポンスについて解説する。

〔認可リクエスト〕

クラウドサービス事業者(RP)はクラウドサービス利用企業(OP)の Authorization エンドポイントに対して ID トークンを要求する。リクエストの例を以下に示す。この例では、ブラウザのリダイレクト(HTTP ステータスコード 302)を利用してリクエストを送信することを想定している。

```
HTTP/1.1 302 Found
Location: https://op.example.co.jp/authorize?
  response_type=id_token
  &client_id=jdDFd8fdl
  &redirect_uri=https%3A%2F%2Frp.example.jp%2Fcb
  &scope=openid
  &state=3ce334d8ka0
```

「response_type=id_token」が ID トークンの要求を意味する。

```
HTTP/1.1 302 Found
Location: https://rp.example.jp/cb#
    id_token=eyJ0 ... NiJ9.eyJ1c ... I6IjIifX0.DeWt4Qu ... ZXso
    &expires_in=3600
    &state=3ce334d8ka0
```

クラウドサービス事業者(RP)は、レスポンスの ID トークン(id_token)からユーザ識別子を抽出してサービス利用者の特定を行う。また、必要に応じて ID トークン(id_token)からユーザ識別子以外の属性情報を抽出し利用することもできる。

認可コードフローの場合と異なり、Implicit フローの場合は ID トークンが URL フラグメントとして渡される(図 4.2 のステップ 6)。URL フラグメントはブラウザのリダイレクトではクラウドサービス事業者(RP)に渡らないため、JavaScript を利用して URL フラグメントから ID トークンを抽出し、クラウドサービス事業者(RP)に送信する必要がある(図 4.2 のステップ 7)。

4.2.4 Implicit フロー利用時の ID トークンの真正性と有効性の検証

Implicit フローを利用する場合、クラウドサービス事業者(RP)は ID トークンの有効性(有効期限)の確認に加えて、真正性を検証しなければならない。ID トークンには、真正性検証のためにデジタル署名、もしくは MAC(Message Authentication Code) 値が付加されているため、これらを利用して真正性の検証が可能である。デジタル署名を利用する場合は秘密鍵と公開鍵が必要となり、MAC を利用する場合は共有鍵が必要となる。ここでは、真正性の検証に必要となるこれらの鍵の運用案を提示する。真正性と有効性を検証するための具体的な方法は OpenID Connect の仕様本文を参照して頂きたい。

4.2.4.1 デジタル署名を利用する

デジタル署名を利用する場合、クラウドサービス利用企業(OP)は秘密鍵を利用して生成したデジタル署名を ID トークンに付加し、クラウドサービス事業者(RP)は公開鍵を利用してデジタル署名を検証する。そのため、クラウドサービス利用企業(OP)は何らかの方法で公開鍵をクラウドサービス事業者(RP)に提供する必要がある。また、クラウドサービス事業者(RP)はクラウドサービス利用企業(OP)毎に公開鍵を管理する必要がある。

クラウドサービス利用企業(OP)がクラウドサービス事業者(RP)に公開鍵を提供する方法として、以下のような方法が考えられる。

1. クラウドサービス事業者(RP)が公開鍵を登録するための GUI インタフェースを用意する。クラウドサービス利用企業(OP)の管理者は GUI インタフェースにアクセスし公開鍵を登録する。
2. クラウドサービス事業者(RP)が公開鍵を登録するための API を用意する。クラウドサービス利用企業(OP)は API を利用して公開鍵を登録するようなプログラムを作成し、クラウドサービス事業者(RP)への公開鍵登録作業を自動化する。
3. クラウドサービス利用企業(OP)は Web サーバなどの特定の場所に公開鍵を配置し、クラウドサービス事業者(RP)が適宜取得する。
4. クラウドサービス利用企業(OP)は OpenID Provider Metadata (メタデータ) を作成し、クラウドサービス事業者(RP)へ提供する。クラウドサービス事業者(RP)はメタデータの jwks_uri で指定された URI から JSON Web Key Set (JWK Set)を取得し、その中に含まれる公開鍵を取り出してデジタル署名の検証に利用する。クラウドサービス利用企業はメタデータの jwks_uri でした URI に JSON Web Key Set を配置しておく必要がある。

4.2.4.2 MAC を利用する

MAC を利用する場合は、クラウドサービス利用企業(OP)とクラウドサービス事業者(RP)が共通鍵をもつ必要

がある。共通鍵を生成・管理する方法として、以下のような方法が考えられる。

1. クライアント・シークレットを利用する。クライアント・シークレットは、クラウドサービス利用企業(OP)がクラウドサービス事業者(RP)に対して発行しているため、双方で共通したデータを保持することになる。そのため、共有鍵として利用可能である。ただし、この方法はクライアント・シークレットを安全に保持できるRP (たとえば Web サーバ上で稼働する Web アプリケーションなど)においてのみ利用可能な方法である。スマートフォンアプリケーションなどのように、リバースエンジニアリングなどによってクライアント・シークレットが漏洩する可能性のある RP ではクライアント・シークレットを安全に保持できないため、この方法は適さない。
2. クライアント・シークレット以外に、MAC 用の共通鍵を作成して共有する。この場合も、RP はクライアント・シークレットを安全に保持しなければならない。スマートフォンアプリケーションなどのように、リバースエンジニアリングなどによって共通鍵が漏洩する可能性のある RP ではこの方法は適さない。

4.2.5 ユーザ識別子(ID トークン)について

クラウドサービス利用者が OpenID Connect を利用してクラウドサービスにログオンする場合、クラウドサービス利用企業(OP)からクラウドサービス事業者(RP)に、利用者を識別するためのユーザ識別子を渡す必要がある。クラウドサービス利用企業(OP)はユーザ識別子を ID トークン内のパラメタである sub に格納してクラウドサービス事業者(RP)に渡し、クラウドサービス事業者(RP)側では ID トークン内の sub パラメタからユーザ識別子を抽出してサービス利用者を特定する。

クラウドサービス利用企業(OP)とクラウドサービス事業者(RP)は、ユーザ識別子として利用する情報について事前に合意しておく必要がある。既に多く利用されている SAML を利用したフェデレーションでは、クラウドサービス事業者(RP)が必要とするユーザ識別子を指定している事例が多い。クラウドサービス利用企業(OP)は、クラウドサービス事業者(RP)の仕様に合わせて、社内のユーザ情報データベースに保存されている情報をユーザ識別子としてクラウドサービス事業者(RP)に提供する。

ユーザ識別子には以下のような要件が求められる。クラウドサービス利用企業(OP)は、以下の要件を満たす識別子を用意する必要がある。

- ユーザ毎に一意な情報であること
- 再利用されないこと

- 退職などにより一度は無効化された識別子が別の従業員に割り当てられないこと
- 一つの識別子を複数の従業員間で使い回さないこと

ユーザ識別子として利用する情報は、上記の要件を満たしたうえで、「ユーザの属性情報」と「仮名情報」に分類される。

- ユーザーの属性情報

  クラウドサービス利用企業(OP)側で従業員に対して発行したユーザ ID、メールアドレス、社員番号などである。

- 仮名情報

  例えばランダムな文字列のように、その値だけでは個人を特定することができないような情報を意味する。このような情報を利用する目的の一つとして、クラウドサービス利用者のプライバシー保護がある。さらにインターネット上での意図しない名寄せを防止するためには、ユーザ識別子となる仮名情報には、「クラウドサービス利用者と、クラウドサービス利用企業(OP)と、クラウドサービス事業者(RP)の組み合わせで一意である」という要件が求められる。このようなユーザ識別子は「PPID(Pairwise Pseudonymous Identifier)」と呼ばれる。

次に、クラウドサービス事業者(RP)の ID トークンの取り扱いについて解説する。クラウドサービス事業者(RP)に対して ID プロビジョニングが可能な場合と不可能な場合の 2 通りに分けて考える。

〔プロビジョニングが可能な場合〕

クラウドサービス事業者(RP)に対して利用者アカウントのプロビジョニングが可能な場合、クラウドサービス利用企業(OP)は、クラウドサービスの利用を許可する従業員(クラウドサービス利用者)のアカウント情報を事前にプロビジョニングしておく。

クラウドサービス事業者(RP)はクラウドサービス利用者の ID トークンを受け取ると、ID トークンからユーザ識別子を抽出し、事前にプロビジョニングされたアカウントとの比較を行う。プロビジョニングによって既に登録されているアカウントの中に ID トークンから抽出したユーザ識別子と一致するアカウントが存在する場合は、そのアカウントの認証が成功したものとみなし、サービスを提供する。

ユーザ識別子として前述の「仮名情報(PPID)」を利用し、プロビジョニングのプロトコルとして SCIM を利用する場合、ユーザ識別子を挿入する SCIM の 属性として id もしくは externalId が適している。詳細は SCIM の仕様を参照して頂きたい。

〔プロビジョニングが不可能な場合〕

クラウドサービス事業者(RP)に対して利用者アカウントのプロビジョニングが不可能な場合は、サービス利用者の初回ログオン時に、ID トークン内のユーザ識別子をもとにしてアカウントを作成する。ユーザ識別子に加えて他の属性情報が必要な場合は以下のいずれかの手段で属性情報を取得する。

- 認可コードフローを利用する場合は、アクセストークンを利用して UserInfo エンドポイントから属性情報を取得する。
- Implicit フローを利用する場合は、ID トークンから属性情報(クレーム)を取得する

4.2.6 属性の受け渡し(個人と組織)

クラウドサービスで必要となるデータは、事前またはサービス利用時に企業もしくは従業員本人からクラウドサービス事業者に引き渡される。これらのデータは以下の3種類に大別可能である。

1. 所属する企業により付与された属性情報

   企業が従業員に付与する属性には従業員番号、所属部署(課、部)、連絡先(電話番号、住所)、職務名などがある。これらは、通常、人事データベース等を使い企業内で管理されている。

2. 所属する企業とは関係しない属性情報

   所属する企業とは関係のない従業員の属性情報には、氏名、年齢、性別などの従業員個人が元々持っている属性の他に、他の企業や組織から付与された属性(携帯電話番号、パスポート番号、クレジットカード情報等)や従業員の嗜好(通路側の席、禁煙席など)が含まれる。これらの属性には所属企業も把握していないものも多い。

3. 従業員が行う業務に関連した情報

   業務に関連した情報を OpenID Connect でやり取りすることは通常は行われないと思われる。例外として、クラウドサービスと社内システムが密接に連携して動作しているケースが挙げられる。具体的には、社内の承認ワークフローにクラウドサービスが組み込まれており、出張手続きなどの際に社内ワークフローにより付与された承認番号が必要となる場合などが考えられる。

クラウドサービス事業者へこれらのデータを引き渡すには以下の3つの方法がある。

A) プロビジョニングによる方法

B) フェデレーション（IDトークン、userinfoエンドポイント）を利用する方法

C) ユーザ（従業員）がフォームに直接入力する方法

このなかでは、A）のプロビジョニングによる方法が最も一般的である。SCIMやCSV等を使い予め必要となるデータをクラウドサービス側に設定しておくことにより、迅速にサービスが使えるようになるという利点は大きい。また利用企業側でのシステム変更も最小限で済むという利点もある。この方法は、特に上記1）のような企業が従業員に対して付与した属性情報について有効である。

問題点としては、必要とされていない属性情報も引渡してしまう可能性があることが挙げられる。プロビジョニング済みの全ての従業員がサービスを使うとは限らないうえ、サービスを使ったとしても、そのサービスが全ての属性を必要とするかは分からないためである。また上記2のように、企業側で把握していない属性や、把握していても、個人の同意を得ずにサービス事業者に渡すべきではない属性もあるかもしれない。

これに対して、B）のようにIDトークンを含めたり、userinfoエンドポイントに問合せたりする方法では、クラウドサービス事業者側から要求に応じてデータを渡す（Just In Time プロビジョニング）ことになるため、必要とされていない属性まで引き渡してしまうことは少なくなる。またOpenID Connectのフローに従い、個人の同意を得る処理や複数の値から選択する処理を含めることも可能になる。そのため、個人のプライバシー保護の点からは、好ましい方法と言えるが問題もある。利用企業側で既存システムに機能を追加する必要が発生し、コストがかかってしまう点である。

ユーザ（従業員）がクラウドサービス事業者のアプリケーションが表示するフォームに直接入力するC）の方法は従来から行われてきたものである。こちらも開発、改造のコストがかかるという点ではB）と同様であるが、そのコストが利用企業側ではなく、サービス事業者側に発生する点に違いがある。また、ユーザが入力するため、上記2）に分類される所属企業が把握していない属性もサービス側に渡すことが可能になる。特にユーザの好みのようなもの（例：通路側の席）にはこの方法が有効である。その反面、どんな値でも入力出来てしまうため、データの正確性は期待出来ない。例えば、上記1に分類される「役職」のような属性は従業員個人が入力すべきではないかもしれない。また、言うまでも無く、入力に手間がかかることが、この方法の最大の問題点である。

以下にOpenID Connectを使用する場合に推奨されるデータの受け渡し方法についてまとめる。

- 企業が従業員に対して付与した属性については、従業員がサービスを使用する前にSCIMやCSV等を使い予めサービス事業者側にプロビジョニングしておく。
- 企業が付与したものではない属性は、サービス側で必要になったときに、必要になった属性のみをOpenID Connectのフローに乗せることによりサービス事業者に伝えることが個人プライバシー保護の観点からは望ましい(Just In Time プロビジョニング)。また属性の種類によっては、そのフローのなかで従業員の同意を取り付ける必要があるケースも考えられる。
- 従業員の個人的な好みのようなものは企業側で把握していないため、サービス事業者が用意したフォームに企業の従業員が直接入力するという方法を採用する。

もちろん、全ての利用企業で上記の全ての方法が必要になるわけではない。それぞれの企業の状況に即して上記の方法をどのように組み合せるかを検討することになるはずである。

4.2.7 標準スキーマで定義されていない属性の取り扱いについて

UserInfo エンドポイントからのレスポンスや ID トークンに格納することができる属性情報(クレーム)は、氏名、メールアドレス、連絡先など一般的に利用される情報に関しては OpenID Connect の仕様で属性名やデータ型が定義されている。仕様で定義されていない属性に関してはスキーマを定義する必要がある。

OpenIDファウンデーション・ジャパンでは、日本企業において利用が予想される日本独自の属性の必要性について検討し、多くの企業で理想が予想される属性に関してはSCIMで利用するためのスキーマを定義している(「4.2.1.4 SCIM拡張仕様(共通)とその利用ガイド」を参照)。SCIMで利用されるデータフォーマットもJSON形式であるため、新たに定義したSCIM用の属性スキーマはOpenID Connectでも利用可能である。 そのため、OpenID Connectの標準スキーマで対応できない属性情報を利用したい場合は、本稿に記載されているSCIM用のスキーマを利用することで、スキーマ定義の工数を削減することができる。

4.2.8 ログオンページの構成

OpenID Connectの導入に際しては、それまでクラウドサービス側で行っていた認証がサービスを利用する企業側で行われるようになるため、ログオンの処理シークエンスに変更が生じることになる。ログオン・シークエンス

はどこを起点とするかにより以下の2つに分類できる。

1. クラウドサービス利用企業を起点とするシークエンス(OP 起点)

   クラウドサービス利用企業が起点となる場合は、企業内にポータルサーバがあることが前提となる。ポータル画面が表示された時点において認証は済んでいることを前提とする。クラウドサービスを利用する際には、ユーザ(従業員)は、まず、ポータルページ上のリンクをクリックすることによりクラウドサービス側に存在する認可リクエストの生成処理を起動する。これにより、認可リクエストが企業側の OP に送られ OpenID Connect のフローが開始される。

2. クラウドサービス事業者を起点とするシークエンス(RP 起点)

   認可リクエストを生成する際には、どの OP に対する認可リクエストであるかの情報が必要になる。ユーザ(従業員)がクラウドサービスへアクセスすることから始まるシークエンスにおいては、クラウドサービス事業者側で、どの企業に属する従業員のアクセスであるかを判別し対応する OP を特定する必要が生じる。サービスを提供している URL(ドメインやパラメタ)が企業毎に異なる場合には、それを手がかりにして対応する OP を特定することが可能である。そうした手がかりがない場合には、ユーザに自分が属する企業を選択させる方法もあるが、企業向けのサービスとしては適当ではないことが多い。その過程で、他にどのような企業がサービスを利用しているかが分かってしまうためである。

上記1と2は排他的ではない。どちらを起点としてもログオンが出来るように設定しておくことも可能である。企業内に既にポータルサーバがある場合には、そこにリンクを設けることにより、シームレスに外部のサービスが利用できるようになるため、上記1のように企業側を起点とする方法が推奨される。

既にサービスを提供、運営している事業者が OpenID Connect を採用する場合には、既存のログオンページとの整合性が重要になる。従来と同じく、上記2のように事業者側を起点とし、利用企業毎に用意された既存のログオンページに「OpenID でログオン」等のボタンを追加することで、ユーザの違和感を最低限に抑えた OpenID Connect の導入が可能になると思われる。

4.2.9 再認証を求めるには

クラウドサービス事業者側で必要とされる認証レベルが分かっている場合には、サービスを利用する企業側

(OP)に予め伝え合意しておくことにより、再認証処理を無くすことが可能である。しかし、サービス事業者が認証レベルの異なる様々なコンテンツを持ち、ユーザ（従業員）がどこにアクセスするか予想がつかない場合には再認証を求める処理は必須になる。

以下に再認証を行うために必要な処理を述べる。

〔クラウドサービス利用企業側で必要となる処理〕

認可サーバにおいて、認可リクエストに含まれる認証コンテクスト・クラスに基づいてユーザ認証を行い、実施された認証に対応する認証コンテクスト・クラスを ID トークンの acr パラメタに設定する。

〔クラウドサービス事業者側で必要となる処理〕

受け取った ID トークンに含まれる認証コンテクスト・クラス(acr パラメタ)をユーザセッションに保持しておく。ユーザ（従業員）がそれよりも厳密な認証を必要とするコンテンツにアクセスした際には、それに対応する認証コンテクスト・クラスを求める認可リクエストを生成し、認可サーバに送出する。認証コンテクスト・クラスは認可リクエストの acr_values パラメーターで指定する。

4.3 SCIM のエンタープライズ IT 適用ガイド

本節では、SCIM をエンタープライズ IT に適用する場合に、どのように適用すれば良いかのガイドラインを示す。

SCIM を用いて、エンタープライズ IT におけるクラウドサービスに対するアイデンティティ・プロビジョニングを実現しようとすると、先に述べたとおり、コンシューマ IT とは異なるエンタープライズ IT ならではの新たな要件への対応が必要となり、従来の SCIM 技術仕様をそのまま適用することでは対応できない点も出てくる。

本節では、まず、「4.1 日本のエンタープライズ IT におけるアイデンティティ情報と認証システムの特性」の内容を受け、日本のエンタープライズ IT におけるクラウドサービスに対するアイデンティティ・プロビジョニングの主な要件をあらためて整理する。次に、それら要件に対する SCIM の対応可否を整理する。次に、SCIM 標準仕様をどのように利用すれば良いかのガイドラインを示す。そして最後に、SCIM 拡張仕様（共通およびサービス個別）をどのように拡張定義し利用すれば良いかのガイドラインを示す。

4.3.1 エンタープライズ IT におけるアイデンティティ・プロビジョニング要件

「4.1 日本のエンタープライズ IT におけるアイデンティティ情報と認証システムの特性」で整理した日本のエンタープライズ IT におけるアイデンティティ情報と認証システムの特性から、日本のエンタープライズ IT におけるクラウドサービスに対するアイデンティティ・プロビジョニングの主な要件をあらためて抽出すると、以下に挙げる

要件が存在する。

(1) アイデンティティのライフサイクル上の運用が、企業における組織と人の変動に応じて発生すること

(2) 定期的な組織改編や一斉の人事異動への対応が必要

(3) プロビジョニングの対象が企業の組織とそれに所属する多数の従業員であること

(4) アイデンティティ情報を管理するために、多数の組織、グループ、個人の中から複雑な条件を指定して抽出する検索が必要

(5) 利用企業側のファイアウォールによる制限のため、クラウドサービス側からアイデンティティ情報と認証システムの基盤へアクセスできない環境に対する対応が必要

(6) 深い組織階層や多数の兼務所属を扱えることが必要

(7) アイデンティティ属性の多様な言語表現が必要

上記の要件のうち、(1)、(2)、(3)、(4)、(5)、(6)はコンシューマ IT とは異なる要件である。また、(2)、(6)、(7)は日本特有の要件であり、欧米とは異なる。(なお、(7)はマルチバイト文字を使用するアジア諸国等では共通の要件である。

4.3.2 エンタープライズ IT の要件に対する SCIM の対応可否

「4.1 日本のエンタープライズ IT におけるアイデンティティ情報と認証システムの特性」で整理し前節であらためて抽出した、日本のエンタープライズ IT におけるクラウドサービスに対するアイデンティティ・プロビジョニングの主な要件に対して、SCIM はどこまで対応可能なのか。

まず、SCIM 標準仕様により対応できるのはどこまでかを整理する。

「3 アイデンティティ・プロビジョニング標準プロトコル ～ SCIM 解説」で解説したとおり、SCIM 標準仕様は大きく分けて標準スキーマと標準プロトコルからなる。

標準スキーマは、ユーザとグループのリソースを定義しているが、ユーザの属性として従業員番号、名前、肩書き、連絡先、所属グループ、所属組織(organization, division, department の 3 階層のみ)、上司などの定義に留まっている(いずれもシングル言語表現のみ)。また、グループの属性として定義しているのは表示名、メンバのみである。

これら標準スキーマだけでは日本のエンタープライズ IT における要件に対応するには不十分であり、スキーマの拡張が必要である。(具体的には後述。)

一方、標準プロトコルは、SCIM のサービスコンシューマとサービスプロバイダ間でアイデンティティ・プロビジョニングを実現するために必要な、HTTP メソッドを用いた CRUD 操作(Create, Read/Retrieve, Update/Modify, Delete などの操作)を定義している。また、多数のリソースに対する操作を一括で実行するための Bulk 操作も定義している。

標準プロトコルは、基本的には、日本のエンタープライズ IT における要件に対応するのに必要な機能を備えている。(ごく一部の対応できない枝葉の部分に関しては、本節の最後に残課題として触れる。)

より具体的に、SCIM 標準仕様により対応できるのは下記のとおりである。

要件(1)〔アイデンティティのライフサイクル上の運用が、企業における組織と人の変動に応じて発生すること〕

本要件に対して、SCIM 標準仕様は、リソースの生成(Create)、読み出し・検索(Read/Retrieve)、更新・変更(Update/Modify)、削除(Delete)を実行するための操作(いわゆる CRUD 操作)を備えている。また、多数のリソースに対する操作を一括で実行するための Bulk 操作を備えている。

要件(2)〔定期的な組織改編や一斉の人事異動への対応が必要〕

本要件に対して、SCIM 標準仕様は、多数のリソースに対する操作を一括で実行するための Bulk 操作を備えている。

しかし、リソースの開始日や終了日を指定するための属性は、SCIM 標準仕様のスキーマとして定義していない。

要件(3)〔プロビジョニングの対象が企業の組織とそれに所属する多数の従業員であること〕

本要件に対して、SCIM 標準仕様は、組織に所属する個人をアイデンティティ・プロビジョニングの対象とすることが可能となっている。また、個人のアイデンティティ属性として、従業員番号、アカウント名、パスワード、電子メールアドレス、電話番号など、組織人としての属性のうち一部を定義しており、これらをアイデンティティ・プロビジョニングの対象とすることが可能となっている。

一方、日本特有の階層の深い所属組織、兼務所属ごとの役職、各種コードなどの属性は、SCIM 標準仕様のスキーマとして定義していない。また、組織や役職のリソースとしての定義もしていない。

上記の他に、多数の個人を一括でアイデンティティ・プロビジョニングの対象とすることが求められるが、SCIM 標準仕様は Bulk 操作を備えている。

要件(4)〔アイデンティティ情報を管理するために、多数の組織、グループ、個人の中から複雑な条件を指定して抽出する検索が必要〕

本要件に対して、SCIM 標準仕様は、属性値の有無、同値、含有、順序に関する条件や、条件の論理演算（論理積、論理和）による組合せを指定した検索が可能となっている。

しかし、条件の論理否定を指定することはできない。

また、検索結果の表示順を指定するための属性は、SCIM 標準仕様のスキーマとして定義していない。

要件(5)〔利用企業側のファイアウォールによる制限のため、クラウドサービス側からアイデンティティ情報と認証システムの基盤へアクセスできない環境に対する対応が必要〕

本要件に対して、SCIM 標準仕様は、HTTP メソッドを用いたプロトコルにより、企業イントラネットとインターネットの間でファイアウォールによる制限の影響を受けることなく必要な通信のやり取りが可能となっている。

要件(6)〔深い組織階層や多数の兼務所属を扱えることが必要〕

本要件に対して、SCIM 標準仕様は、組織のリソースとしての定義をしていない。また、階層の深い所属組織と兼務所属、兼務所属ごとの役職を表すための属性は定義していない。

要件(7)〔アイデンティティ属性の多様な言語表現が必要〕

本要件に対して、SCIM 標準仕様は、アイデンティティ属性をローカルのマルチ言語で表現するための属性を定義していない。

SCIM 標準仕様により対応できるのは上記のとおりであり、対応できない部分に対応するために SCIM 仕様の拡張（スキーマの拡張）が必要である。

そこで次に、SCIM 拡張仕様により対応できるのはどこまでかを整理する。

SCIM 仕様は拡張性を備えており、標準仕様では不十分な部分を SCIM サービスプロバイダが標準仕様の規定や慣例に従い相互接続性を損なわないように拡張できるようになっている。

プロトコル面では、REST の原則に従い、URL のパラメタを追加したり、リソースのエンドポイントを追加したりすることができる。

スキーマ面では、オブジェクト拡張モデルに従い、新しい属性を定義し既存のリソースに追加したり、新しいリソースを定義したりすることができる。

SCIM 仕様の拡張は、SCIM サービスプロバイダごとに独自に拡張することができる。しかし、相互接続性を損なわないように、多くの SCIM サービスプロバイダにとって同様に拡張が必要な部分は、サービス共通の拡張仕様として定義することが望ましい。そこで、本 WG にて日本のエンタープライズ領域に SCIM を適用する場合に必要となるサービス共通の拡張仕様を定義することにした。（具体的には後述。）それ以上に個々の SCIM サービスプロバイダ特有の拡張が必要な場合は、サービス個別の拡張仕様として定義していただくことになる。

より具体的に、SCIM 拡張仕様により対応できるのは下記のとおりである。

要件(2)〔定期的な組織改編や一斉の人事異動への対応が必要〕

本要件に対して、SCIM 標準仕様により対応できない部分、すなわち、リソースの開始日や終了日を指定するための属性は、拡張スキーマとして定義することにより対応できる。

要件(3)〔プロビジョニングの対象が企業の組織とそれに所属する多数の従業員であること〕

本要件に対して、SCIM 標準仕様により対応できない部分、すなわち、日本特有の階層の深い所属組織、兼務所属ごとの役職、各種コードなどの属性は、拡張スキーマとして定義することにより対応できる。また、組織や役職のリソースも、拡張スキーマとして対応できる。

要件(4)〔アイデンティティ情報を管理するために、多数の組織、グループ、個人の中から複雑な条件を指定して抽出する検索が必要〕

本要件に対して、SCIM 標準仕様により対応できない部分、すなわち、検索結果の表示順を指定するための属性は、拡張スキーマとして定義することにより対応できる。

一方、条件の論理否定の指定は、SCIM サービスプロバイダであるクラウドサービス事業者が任意で付加的にサポートすることにより対応できるが、サービス個別の対応になるため相互接続上の課題が残る。

要件(6)〔深い組織階層や多数の兼務所属を扱えることが必要〕

本要件に対して、SCIM 標準仕様により対応できない部分、すなわち組織のリソースは、拡張スキーマとして定義することにより対応できる。また、階層の深い所属組織と兼務所属、兼務所属ごとの役職を表すための属性も、拡張スキーマとして対応できる。

要件(7)〔アイデンティティ属性の多様な言語表現が必要〕

本要件に対して、SCIM 標準仕様により対応できない部分、すなわち、アイデンティティ属性をローカルのマルチ言語で表現するための属性は、拡張スキーマとして定義することにより対応できる。

SCIM 拡張仕様により対応できるのは上記のとおりであり、特に断り書きが無い部分はいずれもサービス共通の拡張仕様として定義し対応できるようにする。それでも対応できない部分は今後の課題ということになる。

最後に、SCIM により対応できない残課題は何かを整理しておく。

- 要件(4)に対する対応可否のところで述べたとおり、検索条件の論理否定の指定は SCIM 標準仕様では定義していない。SCIM サービスプロバイダであるクラウドサービス事業者が任意で付加的にサポートすることによりサービス個別の対応は可能になるが、相互接続上の課題が残る。
- 多数のリソースに対する操作を一括で実行するための Bulk 操作を SCIM 標準仕様では定義しているが、Bulk 操作全体をトランザクションとして実行する仕様は定義しておらず、その原子性（Atomicity）はリソース単位である。現時点では、SCIM のサービスコンシューマとサービスプロバイダが SCIM 仕様の外

側で必要に応じて対処するしかない。

表 4.1: 要件に対する SCIM の対応可否

要件	標準仕様による対応		拡張仕様による対応	残課題
	標準スキーマ	標準プロトコル		
①対象が企業の組織とそれに所属する多数の個人であること	✓アカウント名属性 ✓パスワード属性 ✓電子メールアドレス属性 ✓電話番号属性 ✓従業員番号属性 等	✓Bulk操作	✓組織リソース ✓役職リソース ✓所属組織・役職属性 ✓各種コード属性	✓Bulk操作のトランザクション処理
②深い組織階層や多数の兼務所属を扱えること			✓組織リソース ✓所属組織・役職属性	
③アイデンティティ属性の多様な言語表現が求められること			✓ローカル名(マルチ言語表現)属性	
④多数の組織、グループ、個人の中から複雑な条件を指定して抽出する検索ができること		✓属性値の有無、同値、含有、順序に関する条件 ✓条件の論理演算(論理積、論理和)による組合せ	✓表示順属性	✓条件の論理演算(論理否定)
⑤アイデンティティのライフサイクル上の運用が、企業における組織と人の変動に応じて発生すること		✓CRUD操作 ✓Bulk操作		
⑥定期的な組織改編や一斉の人事異動を扱えること		✓Bulk操作	✓開始日属性 ✓終了日属性	✓Bulk操作のトランザクション処理
⑦アイデンティティ・プロバイダが企業イントラネットの中に存在すること		✓HTTPメソッド		

次節以降で、日本のエンタープライズ領域における要件に対して、SCIM の標準仕様および拡張仕様により具体的にどのように対応すれば良いかを説明する。

4.3.3 SCIM 標準仕様の利用ガイド

本節では、日本のエンタープライズ IT におけるクラウドサービスに対するアイデンティティ・プロビジョニングの主な要件に対応するために、SCIM 標準仕様を具体的にどのように利用すれば良いかのガイドラインを示す。

4.3.3.1 SCIM 操作の利用

先に述べたとおり、SCIM 標準プロトコルは、SCIM のサービスコンシューマとサービスプロバイダ間でアイデンティティ・プロビジョニングを実現するために必要な、HTTP メソッドを用いた CRUD 操作（Create, Read/Retrieve, Update/Modify, Delete などの操作）を定義している。また、多数のリソースに対する操作を一括で実行するための Bulk 操作も定義している。

これらの操作を用いて、ユーザやグループなどのリソースの生成(Create)、読み出し・検索(Read/Retrieve)、更新・変更(Update/Modify)、削除(Delete)を実行することができる。

それぞれの操作の機能と利用場面を以下の表に整理する。

表 4.2: SCIM 操作の機能と利用場面

SCIM操作 (HTTPメソッド)	機能	利用場面	備考
Retrieve - Read (GET)	単一のリソースを読み出す	特定のリソースの内容を読み出して確認したい場合	
Retrieve - Search (GET)	複数のリソースを検索する	条件に合致するリソースを検索して一覧で確認したい場合	
Create (POST)	単一のリソースを生成する	対話的に即座に、単一のリソースを更新したい場合	
Modify - Replace (PUT)	単一のリソースを置換する（全部）		
Modify - Update (PATCH)	単一のリソースを変更する（部分）		✓実装オプション ✓パラメタの指定方法が複雑
Delete (DELETE)	単一のリソースを削除する		
Bulk (POST)	複数のリソースを一括で更新する（＝単一リソースの生成、置換、変更、削除の組合せ）	定期的にバッチ処理で一括して、複数（大量）のリソースを更新したい場合	✓実装オプション ✓CSV→JSON変換要 ✓複数リソースにわたるトランザクション処理には未対応

日本のエンタープライズ IT において SCIM を適用する場合、先に述べたとおり、アイデンティティのライフサイクル上の運用が企業における組織と人の変動に応じて発生するという要件や、定期的な組織改編や一斉の人事異動を扱えるという要件に対応することが求められる。

そのため、企業イントラネットの中に存在する人事データベースやアイデンティティ管理システムと連動して SCIM 操作を実行するアイデンティティ・プロビジョニングの仕組みを実装するのが望ましい。

その中で SCIM 操作の利用パターンは、1 回の操作で対象となるリソース数の規模と、求められるリアルタイム性によって大きく 2 つに分けられる。

- リアルタイム性を要する情報を即座にクラウドサービスへ連携するパターン
- クラウドサービス利用企業側で累積した情報を一定間隔でクラウドサービスへ連携するパターン

前者は、クラウドサービス利用者のパスワード変更や突発的に発生した人事異動への対応時などが考えられる。このような場合、操作対象のリソースは少なく、1操作で1リソースを対象に SCIM 操作を発行し、操作が複数リソースにわたる場合はこれを繰り返す。エラーは即座にクラウドサービス利用者(利用企業)に通知されるのが望ましい。

後者は、定期的な一斉の人事異動や組織改編時などが考えられ、このような場合、操作対象のリソースは多数になる。1操作1リソースを対象に順番に SCIM 操作を発行する方法も考えられるが、SCIM の標準プロトコルはオプション(実装が必須ではない)機能として Bulk(一括)操作を定義しており、それを用いる方法が一般的である。

Bulk 操作は1操作で複数リソースを対象に処理を可能とする操作で、一括処理を行う場合に用いることを想定しているが、トランザクションの概念は無く、処理途中でエラーが発生してもロールバックする機能は無い。(レスポンスには個々のリソースの処理結果が列挙されている。) そのため、エラー発生時に即座に対処を要する処理での利用には向かない事に注意する必要がある。

4.3.3.2 SCIM 標準スキーマの利用

先に述べたとおり、SCIM 標準スキーマは、ユーザとグループのリソースを定義している。ユーザの属性として従業員番号、名前、肩書き、連絡先、所属グループ、所属組織(organization, division, department の 3 階層のみ)、上司などを定義している(いずれもシングル言語表現のみ)。また、グループの属性として表示名、メンバを定義している。

これらの標準スキーマを用いて、ユーザとグループのリソースおよび属性のうちの一部を表現することができる。

それぞれのリソースおよび属性と用途(例)を以下の表に整理する。

表 4.3: SCIM 標準スキーマ(リソースおよび属性)と用途(例)

標準スキーマ	用途（例）	標準スキーマ	用途（例）
ユーザ(User)リソース	ユーザ	addresses属性	住所（会社、自宅等）
userName属性	アカウント名	groups属性	所属グループのid属性値のリスト
name.familyName属性	姓（ローマ字）	エンタープライズユーザ(Enterprise User)リソース	企業ユーザ
name.givenName属性	名（ローマ字）	employeeNumber属性	従業員番号
displayName属性	表示名（ローマ字）	organization属性	会社名（英語）
title属性	肩書き、役職名（英語）	division属性	事業部名（英語）
userType属性	雇用形態（英語）	department属性	部名、課名（英語）
active属性	アカウントの有効／無効	manager.managerId属性	上司のid属性値
password属性	パスワード	グループ(Group)リソース	グループ
emails属性	電子メールアドレス（会社、自宅等）	displayName属性	表示名（英語）
phoneNumbers属性	電話番号（会社、自宅、モバイル等）	members属性	メンバ(ユーザ、グループ)のid属性値のリスト

日本のエンタープライズ IT において SCIM を適用する場合、先に述べたとおり、深い組織階層を扱えることや、アイデンティティ属性を多様な言語で表現できることが求められる。

それに対して、標準スキーマはユーザの所属組織を表すための属性として 3 階層のみの定義となっているため、標準スキーマとして定義している属性には組織階層の要となる要素（会社、事業部、部あるいは課など）のみの値を持たせることになる。3 階層の属性名が organization, division, department であることから、グローバル表記の場合にこれら 3 種類の組織要素に対応する所属組織の名称を英語表記で持たせるのが良いだろう。

また、標準スキーマはシングル言語表現のみの対応となっているため、グローバルなインターネット環境での相互接続性を考慮し、標準スキーマとして定義している属性にはローマ字や英語表記の値を持たせるのが良いと考える。

4.3.4 SCIM 拡張仕様（共通）とその利用ガイド

本節では、日本のエンタープライズ IT におけるクラウドサービスに対するアイデンティティ・プロビジョニングの主な要件に対応するために、SCIM 拡張仕様（共通およびサービス個別）をどのように拡張定義し利用すれば良いかのガイドラインを示す。

4.3.4.1 SCIM 拡張スキーマ（共通）の定義と利用

先に述べたとおり、日本のエンタープライズ IT における要件に対応するためには、SCIM 標準スキーマのみでは不十分であり、スキーマの拡張が必要である。

そして、SCIM サービスプロバイダごとに独自に拡張することにより相互接続性を損なうことの無いように、多くの SCIM サービスプロバイダにとって同様に拡張が必要な部分は、本 WG にて日本のエンタープライズ IT に SCIM を適用する場合に必要となるサービス共通の拡張スキーマとして定義することにした。

以下にサービス共通の拡張スキーマとその利用方法を説明する。

標準スキーマにより対応できない部分は、先に述べたとおり、以下に挙げるリソースや属性の表現である。

- 組織や役職のリソース
- 階層の深い所属組織と兼務所属、兼務所属ごとの役職、各種コードなどを表現するための属性
- アイデンティティ属性をローカルのマルチ言語で表現するための属性
- 検索結果の表示順を指定するための属性
- リソースの開始日や終了日を指定するための属性

これらのリソースや属性は、多くの SCIM サービスプロバイダにとって同様に拡張が必要であると考えられ、本 WG にてサービス共通の拡張スキーマとして定義する。

サービス共通の拡張スキーマとその用途（例）を以下の表に整理する。

表 4.4: サービス共通の拡張スキーマ(組織リソース、役職リソース)と用途(例)

拡張スキーマ(共通)	用途(例)	拡張スキーマ(共通)	用途(例)
組織(OrganizationalUnit)リソース	組織	役職(Title)リソース	役職
orgUnitCode属性	組織コード	titleCode属性	役職コード
localNames属性	組織名（漢字、仮名、ローマ字、英語等）	localNames属性	役職名（漢字、仮名、ローマ字、英語等）
parentOrgUnitCode属性	上位組織コード	displayOrder属性	表示順
level属性	組織レベル	startDate属性	開始日
type属性	組織種別（business unit, division, department等）	endDate属性	終了日
displayOrder属性	表示順	description属性	説明
startDate属性	開始日		
endDate属性	終了日		
description属性	説明		
members属性	所属メンバのid属性値のリスト		

表 4.5: サービス共通の拡張スキーマ(ユーザリソース、グループリソース)と用途(例)

拡張スキーマ(共通)	用途(例)	拡張スキーマ(共通)	用途(例)
日本エンタープライズユーザ(JapanEnterpriseUser)リソース	日本企業ユーザ	日本エンタープライズグループ(JapanEnterpriseGroup)リソース	日本企業グループ
localNames属性	姓名（漢字、仮名、ローマ字、英語等）	type属性	グループ種別
position属性	職位	filter属性	動的グループの生成フィルタ記述
registrationDate	入社日	displayOrder属性	表示順
displayOrder属性	表示順	startDate属性	開始日
startDate属性	開始日	endDate属性	終了日
endDate属性	終了日	description属性	説明
description属性	説明		
organizationalUnits属性	所属組織・役職		

サービス共通の拡張スキーマは、大きく分けて以下に挙げる 4 つがある。それぞれの特徴は以下のとおりである。

1. 組織リソース

   属性として組織コード、上位組織コード、組織レベル、組織種別を定義しており、階層の深い組織を表現できるようになっている。また、ローカルのマルチ言語で表現できる組織名属性を定義している。

2. 役職リソース

   属性として役職コードや、ローカルのマルチ言語で表現できる役職名属性を定義している。

3. ユーザリソースへの追加属性

   ローカルのマルチ言語で表現できる姓名属性を定義している。また、所属組織・役職属性を定義し、階層の深い所属組織と兼務所属、ならびに兼務所属ごとの役職を表現できるようになっている。

4. グループリソースへの追加属性

   グループ種別属性や、動的グループの生成フィルタを記述するための属性を定義している。

以下にそれぞれのスキーマ定義をより詳しく具体的に示す。

スキーマの識別には URI を用いるが、本 WG にて定義する拡張スキーマを識別するための URI として、以下に示すものを用いる。

```
urn:oidfj:scim:schemas:extention:enterprisejp:1.0
```

表 4.6: 組織(OrganizationalUnit)リソースの属性一覧※1

名称	属性名	備考
組織コード	orgUnitCode	組織を一意に識別するコード。※2
組織名	localNames	組織名。 complex 型の属性とし、サブ属性で地域・言語・文字種毎に名称を表現可能とする。※3
親組織	parentOrgUnitCode	上位組織。組織コードや組織名を格納する等、実装や運用に応じた利用を想定。
組織レベル	level	組織階層におけるレベル(位置)。数値や記号等、実装や運用に応じた利用を想定。

組織種別	type	組織の種別。(business unit/division/ department 等)
表示順	displayOrder	表示順。
有効期限(開始)	startDate	組織改編等の運用イベントに備え、実装や運用に応じた利用を想定。
有効期限(終了)	endDate	(同上)
説明	description	組織の説明や組織に関する任意の情報を格納。実装や運用に応じた利用を想定。
所属メンバ	members	組織に所属するユーザ情報。ReadOnly の complex 型の属性とし、オプション(実装は必須ではない)として、実装や運用に応じた利用を想定。

※1:上記に加え SCIM 標準で定義される共通属性(id,externalId,meta)を持つ
※2:組織コードの形式として、単一のコードで組織そのものを一意に識別する体系や、複数のコードを組み合わせて値が組織階層そのものをも表す体系などが考えられる。
例えば、前者は"001"といった単一のコードで一意性を保つ方法で、後者は"001-011-111"といった複数のコードが組織階層を成すといった場合である。
場合によっては複数の体系の組織コードを1リソースに格納する事も想定される。その場合 orgUnitCode だけでなく、externalId 属性や description 属性の併用の検討を推奨する。
※3:具体的には以下のような JSON 表記である。

```
"urn:oidfj:scim:schemas:extention:enterprisejp:1.0":{
        "localNames": [
          {
            "value": "営業部",
            "locale": "ja_JP",
            "type": "kanji",
            "primary": true
          },
          {
            "value": "Sales Division",
            "locale": "en_US",
            "type": "english"
          }
        ]
      }
```

上記のように、名称に関係する属性は地域・言語・文字種を表現する形で共通化することで

相互接続性の確保を目指している。

表 4.7: 役職(Title)リソースの属性一覧※4

名称	属性名	備考
役職コード	titleCode	組織を一意に識別するコード。
役職名	localNames	役職名。※3
表示順	displayOrder	表示順。
有効期限(開始)	startDate	組織改編等の運用イベントに備え、実装や運用に応じた利用を想定。
有効期限(終了)	endDate	(同上)
説明	description	役職の説明や役職に関する任意の情報を格納。実装や運用に応じた利用を想定。

※4:上記に加え SCIM 標準で定義される共通属性(id,externalId,meta)を持つ

表 4.8: ユーザリソースの拡張属性一覧

名称	属性名	備考
組織ユーザ名	localNames	ユーザ名。 complex 型の属性とし、サブ属性で地域・言語・文字種毎に名称を表現可能とする。※3
入社日	registrationDate	入社日。
職位	position	組織や役職におけるユーザの位置付けを示す。実装や運用に応じた利用を想定。
表示順	displayOrder	表示順。
有効期限(開始)	startDate	組織改編等の運用イベントに備え、実装や運用に応じた利用を想定。
有効期限(終了)	endDate	(同上)
説明	description	ユーザの説明や役職に関する任意の情報を格納。実装や運用に応じた利用を想定。
所属情報	organizationalUnits	ユーザの所属組織および組織における役職を示す。 complex 型の属性とし、サブ属性で所属組織と役職に関する情報を表現可能とする。※5

※5:具体的には以下のような JSON 表記である。

```
"urn:oidfj:scim:schemas:extention:enterprisejp:1.0":{
        "organizationUnits": [
          {
            "orgUnitId": "0001-0011-0111",
            "orgUnitCode": "0111",
            "orgUnitLocalNameValue": "営業部",
```

```
            "orgUnitLocalNameLocale": "ja_JP",
            "orgUnitLocalNameType": "kanji",
            "parentOrgUnitCode": "0011",
            "orgUnitLevel": "3",
            "orgUnitType": "Division",
            "orgUnitDisplayOrder": "1",
            "orgUnitStartDate": "2013/07/04",
            "orgUnitEndDate": "2023/07/03",
            "orgUnitDescription": "",
            "titleId": "0002-0022-0222"
           "titleCode": "002",
            "titleLocalNameValue": "部長",
            "titleLocalNameLocale": "ja_JP",
            "titleLocalNameType": "kanji",
            "titleDisplayOrder": "1",
            "titleStartDate": "2013/07/04",
            "titleEndDate": "2023/07/03",
            "titleDescription": ""
            "primary": true
          }
        ]
      }
```


全てのサブ属性に値を格納する必要は無く、実装や運用に応じて必要となるサブ属性のみを利用する想定である。
organizationalUnits は複数値を格納可能にすることで、兼務の表現も可能とする。

表 4.9: グループリソースの拡張属性一覧

名称	属性名	備考
グループ種別	type	静的グループ・動的グループなどグループの種類を示す。実装や運用に応じた利用を想定。
条件	filter	グループのメンバを動的に抽出する条件文。 実装や運用に応じた利用を想定。
表示順	displayOrder	表示順。
有効期限(開始)	startDate	組織改編等の運用イベントに備え、実装や運用に応じた利用を想定。
有効期限(終了)	endDate	(同上)
説明	description	グループの説明や役職に関する任意の情報を格納。実装や運用に応じた利用を想定。

4.3.4.2 SCIM 拡張仕様（サービス個別）の拡張方法ガイド

前節で説明したとおり、本 WG にてサービス共通の拡張スキーマを定義したが、それ以上に個々の SCIM サービスプロバイダ特有の拡張が必要な場合は、サービス個別の拡張スキーマとして定義していただくことになる。

サービス個別の拡張スキーマをどのように拡張定義すれば良いのか。その基本方針と拡張手順を以下に示す。

- 基本方針

  1. SCIM: Protocol 1.1, Core Schema 1.1 で定義されている標準仕様（リソースや属性など）の定義には手を加えない。
  2. できるだけ既定の仕様（属性やサブ属性など）をそのまま活用する。
  3. 他のサービスでも共通に必要となる仕様は、OpenIDFJ/EIdWG で拡張仕様（共通）として拡張定義する。
  4. 相互接続性を損なうことの無いよう、十分に配慮する。

- 拡張手順

  1. 拡張仕様（サービス個別）を識別する URI を定義する。
  2. 標準仕様の規定や慣例に従い、プロトコル（パラメタ）、リソース、エンドポイント、属性などの必要な仕様を拡張定義する。
  3. スキーマ拡張定義に必要な項目や形式は、SCIM: Core Schema 1.1 の 10. Resouce Schema を参照。

5. OpenID Connect/SCIM ユースケース

本章では、OpenID Connect と SCIM をエンタープライズ IT に適用する場合のユースケースを説明する。

（本章は、OpenID ファウンデーション・ジャパン 会員企業限定のコンテンツです）

6. 関連技術/概念

6.1 トラストフレームワーク

6.1.1 トラストフレームワークの概要

〔フェデレーションはトラストが前提〕

フェデレーションを利用し、分散環境でサービスを提供する場合、クラウドサービス事業者がクラウドサービス利用企業に認証処理を委譲することは、利用企業の認証処理の結果を信頼することにより成り立つ。信頼することができる適切な認証処理結果は、利用企業が ID に関して ID ライフサイクルの適切な運用を実施していることが前提条件となる。具体的には利用企業が ID 運用管理をポリシとして明文化し、それを遵守していることの説明を行い、事業者はこの説明を元に利用企業の ID 情報の品質について評価を行い、信頼に足る場合、サービスを提供することになる。

〔信頼関係をとりもつフレームワーク〕

事業者が、利用企業の認証処理結果や ID 管理運用について個別に信頼をしていくのは大変に手間がかかる。また、利用企業にとっても事業者に対して個別に適切な運用を実施していることを主張していくことは同様に非常に手間がかかる。

そこで、事業者でも利用企業でもない、第三者機関が中心となり、利用企業が遵守する ID 管理運用ポリシを作成し、利用企業が遵守していることを監査することで、利用企業と事業者の信頼関係を効率良く取り持つ仕組みが考案され、構築された。これがトラストフレームワークであり、このトラストフレームワークを運用する第三者機関がトラストフレームワーク・プロバイダである。

トラストフレームワークは、元々は電子政府を推進する多くの国、地域で策定された「電子政府認証ガイドライン」の中で、サービス利用者が各種手続きサービスを利用するにあたり、必要な仕組みとして説明された。

各種手続きサービスのリスク評価の結果から求められる認証の保証レベル(LoA：Levels of Assurance)が定義されており、利用企業はこの保証レベルに合った適切な認証方式を選択することで、トラストが維持され、サービスを安全・安心に利用することができる。

6.1.2 ケーススタディー(学認)

学術分野で国立情報学研究所(NII)が中心となり構築されているトラストフレームワークが学認(学術認証フェデレーション)である。フェデレーションプロトコルとして SAML をベースとして、ディスカバリーサービスやシング

ルサインオンのやり取りを定義した Shibboleth を利用している。

NII が一年に一度行う ID 運用管理に関する学認アンケートについて、クラウドサービス利用者である大学は自組織の運用状況を回答する。厳格な監査ではないものの、学術分野においてはこれが事業者と利用組織の信頼関係を「効率良く」取り持つ仕組みとして成立していることは興味深い。

学認アンケートのポイントは以下の3点である、

1. governance
    - IdP の運用が組織によっオーソライズされていること。
    - 組織がセキュリティについてのポリシ、運用規則を定め、文書化されそれに従った運用をしている。
2. Privacy
    - 個人情報について法令を遵守していること。
    - より良いのは、『学内外にプライバシーポリシを開示している。』、『新たな SP のサービスを利用するときには、書面またはオンラインで利用者許諾を得ないと利用できないように運用している。』
3. Technical
    - ID 情報の源泉データは、人事 DB、教務 DB 等、組織の中で人事業務を担当する部署により管理されている Authoritative Source を源泉として作成されること。
    - Authoritative Source から作ることができない ID 情報は、作成にあたってワークフロー等で管理部局である人事または学務において適切に管理され、IDのライフサイクルもその一環として管理されていること。
    - 利用者が組織を去った場合、担当部局によって失効作業が行われる体制になっていること。
    - IDとパスワードの配布は、職員証・学生証を用いて本人確認を行った上で、書面で行っていること。承認行為があるなど責任の所在が明確であること。

6.1.3 エンタープライズ適用における課題と考察

エンタープライズ IT においては、学術分野における学認のようなトラストフレームワーク・プロバイダが現在のところ存在しない。現実問題としてクラウド上で情報を共有する場合の情報/システムのオーナがポリシを策定し、情報/システムを共有する企業(クラウドサビス利用者)がこのポリシに基づいた ID ライフサイクル管理を行うことにより、フェデレーションで委譲された認証処理が保証されることになる。この場合の情報システムのオーナは、製

造業におけるサプライチェーンの中で最も力のある企業の場合が多い。

つまりLoAは、力のある情報オーナから情報共有者に対する評価基準となり、情報共有者にとっては、複数のサプライチェーンに参加する場合、複数の情報オーナから異なる運用管理ポリシの遵守を要求される事態が発生する。

情報共有者がこのような事態に柔軟に対応するには、多くの情報オーナが納得するIDライフサイクルに関するポイントを網羅した運用ポリシを予め策定しておく必要がある。実際の運用例から、IDライフサイクルに関する運用ポリシで重要なポイントは、Authoritative Sourceの存在、ID管理ツール利用による運用の恣意性の排除、ID情報の適時更新、IDの厳格な引き渡し、そしてこの運用を維持し続けることに集約される。

情報共有者は自社のIDライフサイクルに関する運用ポリシの策定や見直しを行う時に、このあたりに注意すれば、力にある情報オーナから提示されるポリシにも比較的容易に対応することができるはずである。

また一方で、クラウドサービス利用者がクラウドサービス事業者を選択する局面がある。この場合、いわゆるLoP(LoP:Levels of Protection)を意識した評価選定が必要になる。事業者の市場が学術分野に閉じている、ないし公共色の強いクラウドサービス事業者である学術分野とは異なり、ビジネスとしてのクラウドサービスが中心のエンタープライズ市場では、様々な事業者が存在するため、LoPはより重要な考え方と言える。

6.2 権限委譲の適用と課題

エンタープライズにおける認証認可の問題のひとつに権限委譲がある。OpenID ConnectとSCIMのエンタープライズ利用を進めていけば、この問題は避けては通れない。しかしながら、従来は企業自身のオンプレミスシステムにおいて曖昧な運用によって対処してきた権限委譲の問題を、クラウドサービス事業者という第三者との関係の中で解決していくには、機能整理と役割分担について検討していく必要がある。

本節では、こうした検討に必要な課題整理をまず行い、実現方式の検討については今後の作業とする。

6.2.1 権限委譲が進まなかった背景

コンシューマビジネスでは権限委譲の範囲が局所的(家族や後見人など)であるのに対して、エンタープライズでは組織構造や業務に応じて多様な権限委譲が行われている。権限者が明確に代行者を任命するケースもあれば、一定の条件(特定の業務、期間、職級、権限者の不在など)を満たす場合には自動的に代行者に権限委譲が許可されるケースもあり、また、こうした権限委譲のルールが明文化されていない組織では、俗人的あるいは俗部署的に運用される(都度ルールが変わる)ことも少なくない。

このようにエンタープライズにおける権限委譲の要件は複雑かつ多様であり、事前にルールを定義し難いこと

もあり、システムには実装されずに代行権限者が一時的にアカウントを共有するなど、いわゆる運用でカバーするということが多かった。しかし、アカウント共有は安全管理の面からはできる限り避けるべきであり、また、ID 管理が進めば動的な権限委譲なども実装しやすくなると期待されている。

6.2.2 実現すべき機能の検討

OpenID Connect や SCIM を利用する、つまり認証認可をアプリケーションから独立させて連携させることを想定する場合、図 1 に示すようなアーキテクチャが考えられる。これを実現する機能は、既存の OpenID Connect と SCIM に加えて、以下の 3 つである。

(機能 1)委譲ポリシの記述

委譲ポリシそのものは当然ながらクラウドサービス利用企業自身が規定すべきものである。仮に、後述の(機能 2)を利用企業以外の第三者に委託する場合には、これを明文化して第三者と共有するために、その記述ルールを明確にする必要がある。記述ルールの規定および第三者との共有方法については課題 1 および課題 2 で後述する。

(機能 2)も含めて利用企業自身が提供する場合は、この機能は内部仕様に留まる。

(機能 2)属性情報と委譲ポリシにもとづく認可の判断

もっとも単純には(機能 1)を内包する形でクラウドサービス利用企業自身が提供するケースも考えられるが、典型的な例はアプリケーションを提供するクラウドサービス事業者(リライイング・パーティ自身)が実装するケースだろう。そして、後述の課題が解決されれば、将来的にはリライイング・パーティ以外のクラウドサービス事業者が提供する可能性も考えられる。

(機能 3)判断結果にもとづくアクセス制御

(機能 2)を受けてリライイング・パーティとなるクラウドサービス事業者が提供すべき機能である。(機能 2)をリライイング・パーティ自身が提供する限りは、この機能は内部仕様に留まる。

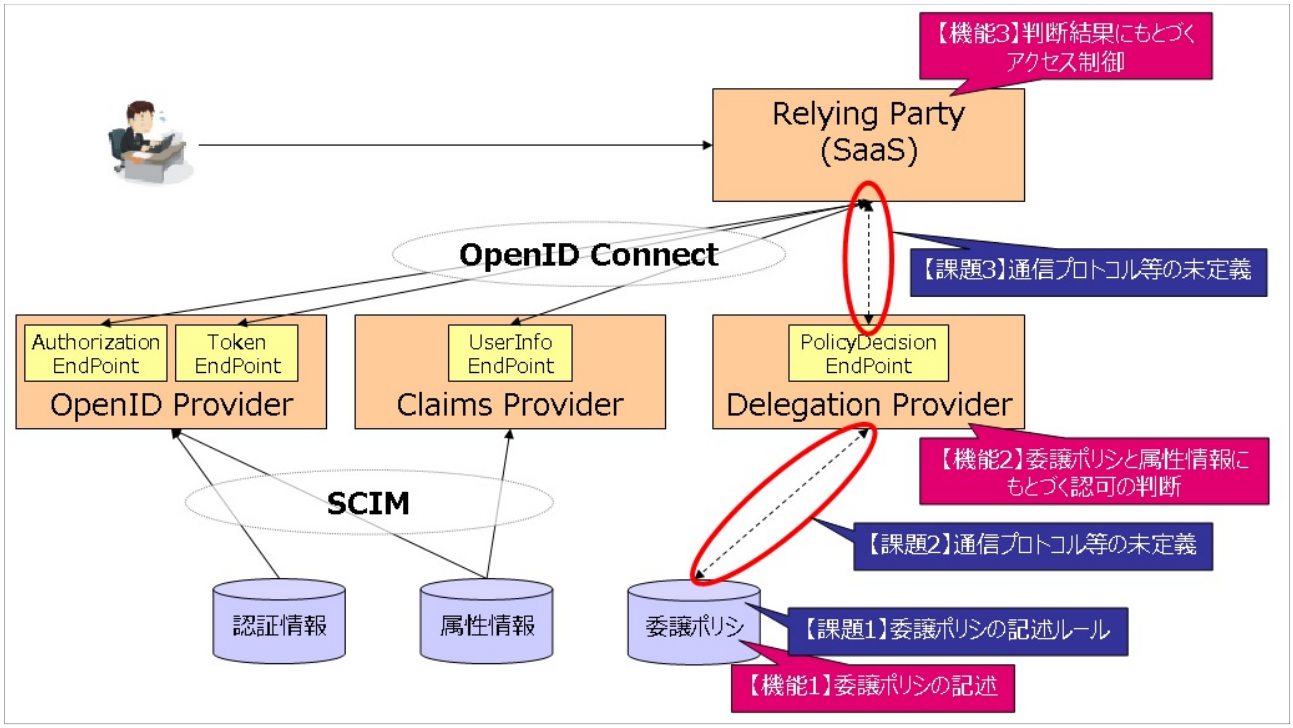


図 6.1: クラウドサービスにおける権限委譲のコンセプト

6.2.3 機能を実現する際の課題

前述の 3 つの機能を提供実現するにあたっては、提供方法次第で以下の課題が想定される。

(課題 1) (機能 1)の記述ルール定義

(機能 2)をクラウドサービス利用企業が実装するのでない限りは避けられない課題である。単純な記述でも十分であれば、前述のようにクラウドサービス事業者の独自仕様でも機能するかもしれないが、高度な記述が必要であれば十分に検討された記述言語が必要になるだろう。また、様々なクラウドサービス事業者を使い分けるクラウドサービス利用企業が増えれば、クラウドサービス事業者間で仕様の共通化が求められるようになるかも知れない。なお、(機能 2)を提供するクラウドサービス事業者が、(機能 3)を提供するクラウドサービス事業者と異なる場合には、本課題は避けて通れない。

(課題 2) (機能 1)と(機能 2)の間の通信プロトコル、データフォーマット等の規定

これも当然ながら(機能 2)をクラウドサービス利用企業が実装するのでない限りは避けられない課題である。

これは、クラウドサービス利用企業からクラウドサービス事業者に送信する情報となるため、暫定的な解決策と

しては利用企業から管理可能なWebインタフェースや何らかのAPIをクラウドサービス事業者(または機能2を提供する第三者)が提供することなどが考えられる。理想的にはSCIMに統合できることが望ましいと考えられるが、現時点では(課題1)の詳細検討に踏み込んでいないため、SCIMへの統合が効果的ではない可能性もあると付け加えておく。

(課題3) (機能2)と(機能3)の間の通信プロトコル、データフォーマット等の規定

これは、(機能3)を提供するクラウドサービス事業者自身が(機能2)を提供するのでない限りは避けられない課題である。

これも(課題2)と同様の観点から、OpenID Connectに統合できることが望ましいだろう。

6.2.4 権限委譲の実現がもたらす効用

エンタープライズにおける権限委譲は複雑かつ多様であり、本節で示した課題はあくまでも大項目に過ぎない。しかし、今後クラウドサービスのエンタープライズ利用が進めば、権限委譲を曖昧なままにしておくことはクラウドサービス利用企業にとってもリライイング・パーティとなるクラウドサービス事業者にとっても様々なリスクあるいはコストを抱え込むことになる。

権限委譲の課題を整理し、いくつかの実現方式を検討していくことで、利用企業およびクラウドサービス事業者に多様性を提供することが期待できる。本節でも少し言及しているが、(機能2)を誰が提供するか、は大きなポイントになると考えられる。クラウドサービス利用企業、リライイング・パーティであるクラウドサービス事業者、あるいは第三者的なクラウドサービス事業者が可能性として考えられる。それぞれが提供する場合のメリット・デメリットを分析することにより、各プレイヤーの選択肢が広がるとともに新たなビジネス機会にもつながることが期待できる。

6.2.5 クラウドサービス利用企業にとってクラウドサービスは多くのアプリケーションのひとつであること

〔フェデレーション機能はクラウドサービス利用企業側に設置されたポータルサーバへ組み込めると良い〕

多くの企業は、社内にポータルサーバを持ち様々な業務アプリケーションへの入り口としている。クラウドサービス事業者が提供するサービスへもこのポータルサーバからアクセスできるようになれば、企業内で既に利用している社内システムとの統合認証環境が構築され、社外システムのちがいを意識せずに済むため、使い勝手を向上する。これを実現するためには、社内で認証済みのユーザが外部のクラウドサービスにアクセスする際のリンクをポータルサーバ上に用意する必要がある。

〔シングルサインオフが実現できること〕

更に、エンタープライズでは単一のサービスだけで業務が完結することは少ないため、利用者は複数のサービスを同時に利用することとなる。その場合に利便性と安全性を向上するためにシングルサインオン／シングルサインオフについても重要な要件となる。

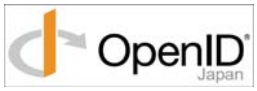
近い将来、OpenID Connect が普及し、多くのクラウドサービス事業者に採用されるようになると、ひとりのユーザ（従業員）がいくつかのサービスに OpenID Connect を使って同時にサインオンしているという状況が起こるはずである。このような場合にはシングルサインオフを使って複数のサービスから同時にサインオフ出来ればユーザの利便性が増し、安全性も向上する。しかし、本書では将来に向けての課題とするに留め詳述はしない。

6.2.6 認証システムの更新は各企業の都合で行われること

エンタープライズ市場では現在のところ、Yahoo や Google のような商用 IdP サービスが立ち上がっておらず、フェデレーションを利用できるか否かはクラウドサービスを利用する企業が IdP を構築済みか否かによる。つまり、クラウドサービス事業者から見て、全顧客企業がすべてフェデレーション対応できている‘純フェデレーション環境’な状況はあり得ない。

〔各企業が個別に持つ認証サーバとの認証連携ができること。認証連携が可能な企業とローカル認証が必要な企業が混在できること〕

多くの企業に対してサービスを提供している事業者が OpenID Connect を採用する場合、全ての企業が同時に導入することを前提にするのは難しい。まず少数の企業が導入し、その後、徐々に導入企業が増えると想定すべきであろう。そのため、導入済みと導入済みではない企業が混在する状況に対する考慮が必要になる。具体的には、アクセスして来たユーザ（従業員）が所属する企業を特定し、その企業が OpenID Connect を導入済みである場合には、企業内におかれた認可サーバまでユーザを導く方法が重要になる。既存のログオンページを変更し、これらの処理に対応させる必要がある。

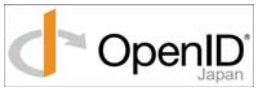


【著者一覧】 （氏名 50 音順）

浅賀 功次	サイボウズ株式会社
岩片 靖	オープンソース・ソリューションテクノロジー株式会社
上田　尊教	エクスジェン・ネットワークス株式会社
江川　淳一	エクスジェン・ネットワークス株式会社
工藤　達雄	株式会社野村総合研究所
桑田　雅彦	日本電気株式会社 （JNSA IDM-WG※）
佐々木　晃法	富士通関西中部ネットテック株式会社 （JNSA IDM-WG※）
島岡 政基	セコム株式会社 IS 研究所
鈴木 恭介	株式会社ジェイティービービジネストラベルソリューションズ
野村 健太郎	オープンソース・ソリューション・テクノロジ株式会社
富士榮　尚寛	伊藤忠テクノソリューションズ株式会社
和田 健	株式会社ウェイズジャパン
渡辺　龍	KDDI 株式会社

ほか

※JNSA IDM-WG：日本ネットワークセキュテリィ協会アイデンティティ管理 WG



【Enterprise Identity Working Group 参加者一覧】 （氏名　50 音順）

浅賀 功次	サイボウズ株式会社
芦田　剛	株式会社野村総合研究所
岩片 靖	オープンソース・ソリューション・テクノロジ株式会社
上田　尊教	エクスジェン・ネットワークス株式会社
江川　淳一	エクスジェン・ネットワークス株式会社
小田切 耕司	オープンソース・ソリューション・テクノロジ株式会社
工藤　達雄	株式会社野村総合研究所
桑田　雅彦	日本電気株式会社 （JNSA IDM-WG※）
作田　宗臣	NTT ソフトウェア株式会社
佐々木　晃法	富士通関西中部ネットテック株式会社 （JNSA IDM-WG※）
島岡 政基	セコム株式会社 IS 研究所
鈴木 恭介	株式会社ジェイティービービジネストラベルソリューションズ
関森信之	株式会社アグレックス
田中 亮	株式会社ウェイズジャパン
手塚　由起子	日本電気株式会社
永野　一郎	NTT ソフトウェア株式会社
野村 健太郎	オープンソース・ソリューション・テクノロジ株式会社
富士榮　尚寛	伊藤忠テクノソリューションズ株式会社
和田 健	株式会社ウェイズジャパン
渡辺　龍	KDDI 株式会社
ほか	

※JNSA IDM-WG：日本ネットワークセキュテリィ協会アイデンティティ管理 WG