WAS V8.5.5 OpenID Connect 認証構成ガイド

Disclaimer

- この資料は日本アイ・ビー・エム株式会社ならびに日本アイ・ビー・エム　システムズ・エンジニアリング株式会社の正式なレビューを受けておりません。
- 当資料は、資料内で説明されている製品の仕様を保証するものではありません。
- 資料の内容には正確を期するよう注意しておりますが、この資料の内容は2016年1月現在の情報であり、製品の新しいリリース、PTFなどによって動作、仕様が変わる可能性があるのでご注意下さい。
- 今後国内で提供されるリリース情報は、対応する発表レターなどでご確認ください。
- ＩＢＭ、ＩＢＭロゴおよびibm.comは、世界の多くの国で登録されたInternational Business Machines Corporationの商標です。他の製品名およびサービス名等は、それぞれＩＢＭまたは各社の商標である場合があります。現時点でのＩＢＭの商標リストについては、www.ibm.com/legal/copytrade.shtmlをご覧ください。
- 当資料をコピー等で複製することは、日本アイ・ビー・エム株式会社ならびに日本アイ・ビー・エム　システムズ・エンジニアリング株式会社の承諾なしではできません。
- 当資料に記載された製品名または会社名はそれぞれの各社の商標または登録商標です。
- JavaおよびすべてのJava関連の商標およびロゴは Oracleやその関連会社の米国およびその他の国における商標または登録商標です。
- Microsoft, Windows および Windowsロゴは、Microsoft Corporationの米国およびその他の国における商標です。
- Linuxは、Linus Torvaldsの米国およびその他の国における登録商標です。
- UNIXはThe Open Groupの米国およびその他の国における登録商標です。

目次

1. OpenID Connectで実現できること

ユースケース：ユーザー認証および属性情報の取得

- 認証が必要なアプリケーションで「○○のIDでログイン」することができ、ユーザー属性情報を取得することができる

（例）GoogleのIDでログインするケース

1. 利用したいアプリケーションのログイン方法をエンドユーザーが選択する（外部IDでログインする選択肢が提供されている）
2. GoogleのIDでログインしたいため、「Sign in with Google」のボタンをクリック
3. Googleのログイン画面にリダイレクトされてユーザーIDおよびパスワードを入力してログインする
4. アプリケーションがGoogleのユーザー情報にアクセスすることを許可するかどうかの同意画面が表示されるため、同意をクリックする
5. エンドユーザーはアプリケーションにアクセスできる
6. アプリケーションはGoogleからユーザー属性情報を入手することが可能になる

ユースケース：利用するメリットは？

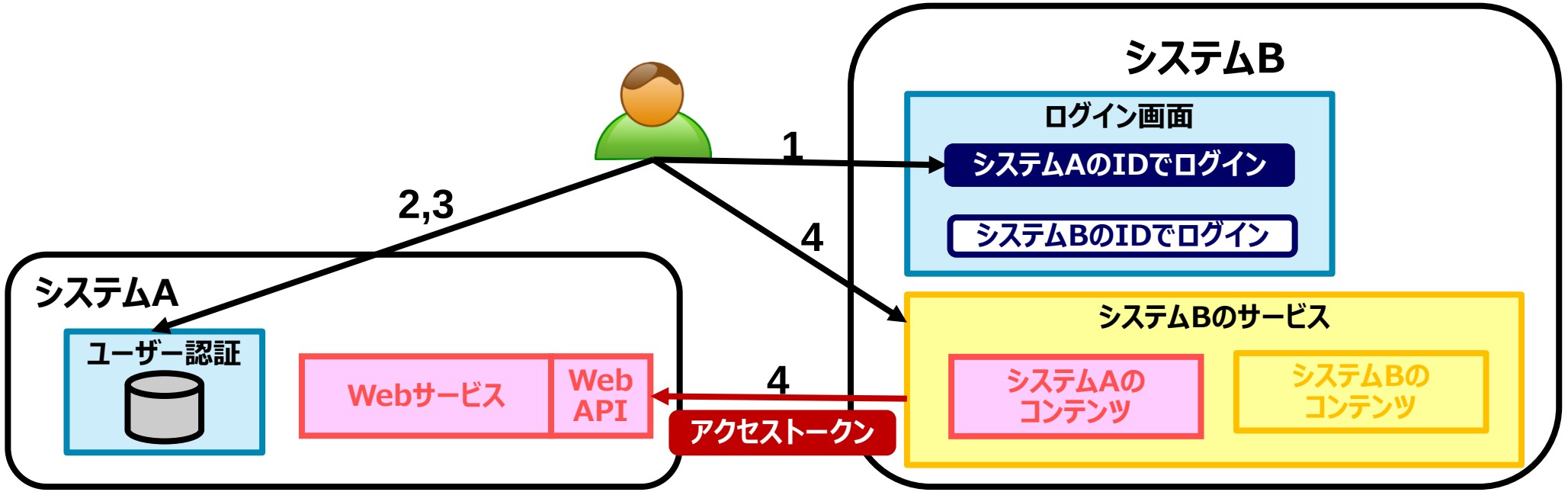
- エンドユーザーの観点
  - 認証プロバイダーのユーザーIDでサービスが利用できるため、サイト毎のパスワードを覚える必要がない
  - OpenID Connectに対応するWebサイト間でシングルサインオンが可能となる
  - Webサイトが認証プロバイダーからユーザー情報を取得することが可能になるため、 Webサイト上でユーザー登録をする際に自動的にユーザー情報を補完することも可能になるため、エンドユーザーによる入力の手間が省ける

- アプリケーション開発者およびサービス提供者の観点
  - 認証機能のアウトソースすることによる実装コストの低減
    - ユーザーのパスワード管理が不要となる
    - 認証プロバイダーで提供されている高度な認証機能（2要素認証など)を簡単に利用することができる
  - エンドユーザーの利便性向上に伴うサービスの満足度向上が見込める
  - OAuth認証を実装したアプリケーションであれば、OpenID Connectに移行させることで、アクセストークン置き換え攻撃やリプレイアタックを防止することが可能となり、セキュリティーレベルが向上する

(オプション)従来のOAuthの機能を利用したWeb APIのアクセス認可

- Web APIのアクセス認可を利用したWebコンテンツのマッシュアップが可能になる
  - OpenID ConnectはOAuthをベースとした仕様(*)であるため、従来のOAuthの機能であるWebAPIのアクセス認可を利用することができる

1. エンドユーザーがシステムBにアクセスして「システムAのIDでログイン」をクリックする
2. システムAでユーザー認証する
3. 同意処理によりシステムAが提供するWeb APIへのアクセス権限がシステムBに付与される
4. システムBはシステムAのWebAPIにアクセス可能となり、システムAとBのWebコンテンツをマッシュアップしたサービスを提供できる

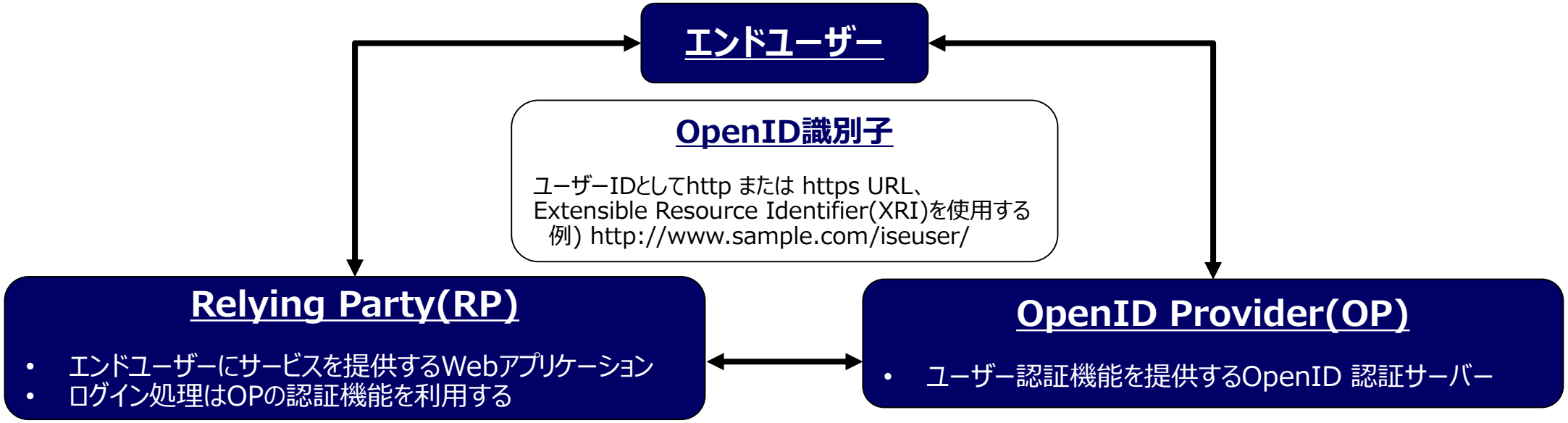


(*)OpenID ConnectとOAuthは別の仕様として定義されており、WebAPIのアクセス認可はOAuth側の仕様に含まれる

2. OpenID Connectの概要

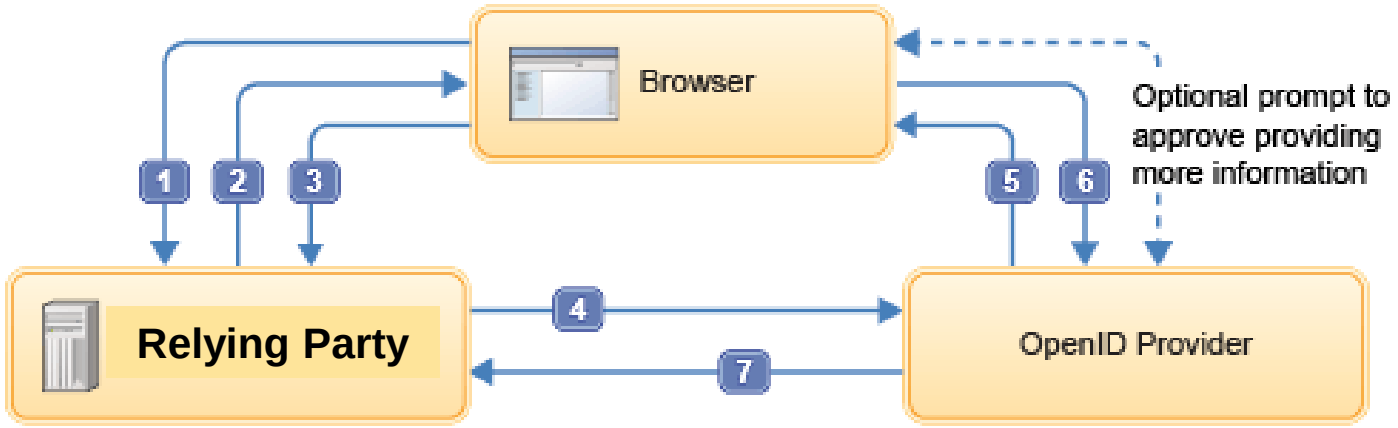
背景: OpenID Connectのベースとなる仕様について　～OpenID～

- ユーザー中心の分散型デジタルアイデンティティーフレームワークの仕様
  - OpenID Authentication 2.0 (http://openid.net/specs/openid-authentication-2_0.html)として仕様策定
  - 「○○のIDでログイン」
  - エンドユーザーは自身のIDをOpenIDに対応した複数の認証サービスの中から自由に選択することが可能
  - エンドユーザーはWebサイト毎にID/パスワードを登録する必要がなくなり、ユーザビリティーが向上する
  - Webサイト側はパスワードやemailアドレスなどのユーザー情報にアクセスする必要がなくなる
  - Webブラウザーベースのアプリケーションにのみ対応している



(参考情報)OpenIDによる認証フロー

- 一般的な OpenID 認証フローは次のとおり

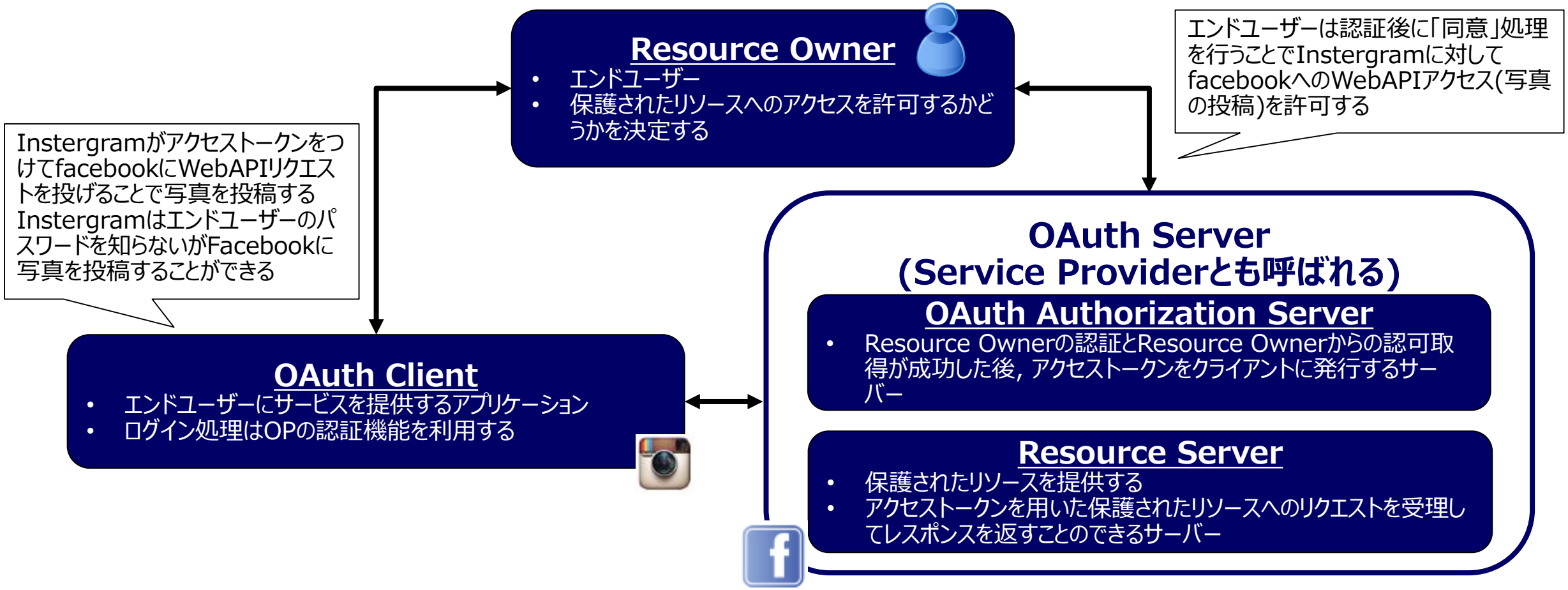


1. ユーザーは、保護リソース (例:Web ページ) にアクセスを試みる
2. RPは、保護リソース用のフォーム・ログイン・ページを提示する
3. ユーザーは、OpenID 識別子を入力する
4. RP は、ユーザー ID を取得し、適切な OpenID プロバイダーにユーザーを転送する
5. OpenID プロバイダーは、資格情報を入力するようユーザーにプロンプトを出す
6. ユーザーは、OpenID プロバイダーと関連付けられたアカウントの資格情報を入力する
7. OpenID プロバイダーは、ユーザーを認証する。さらに、オプションで、RP へのユーザー情報の提供を承認または拒否するようユーザーにプロンプトを出す。 次に、認証結果と共にユーザーを RP に返す。

OpenID プロバイダー認証が成功した場合、RP はユーザーの認可を試みる。認可が成功すると、RP はユーザーとの認証済みセッションを確立する。

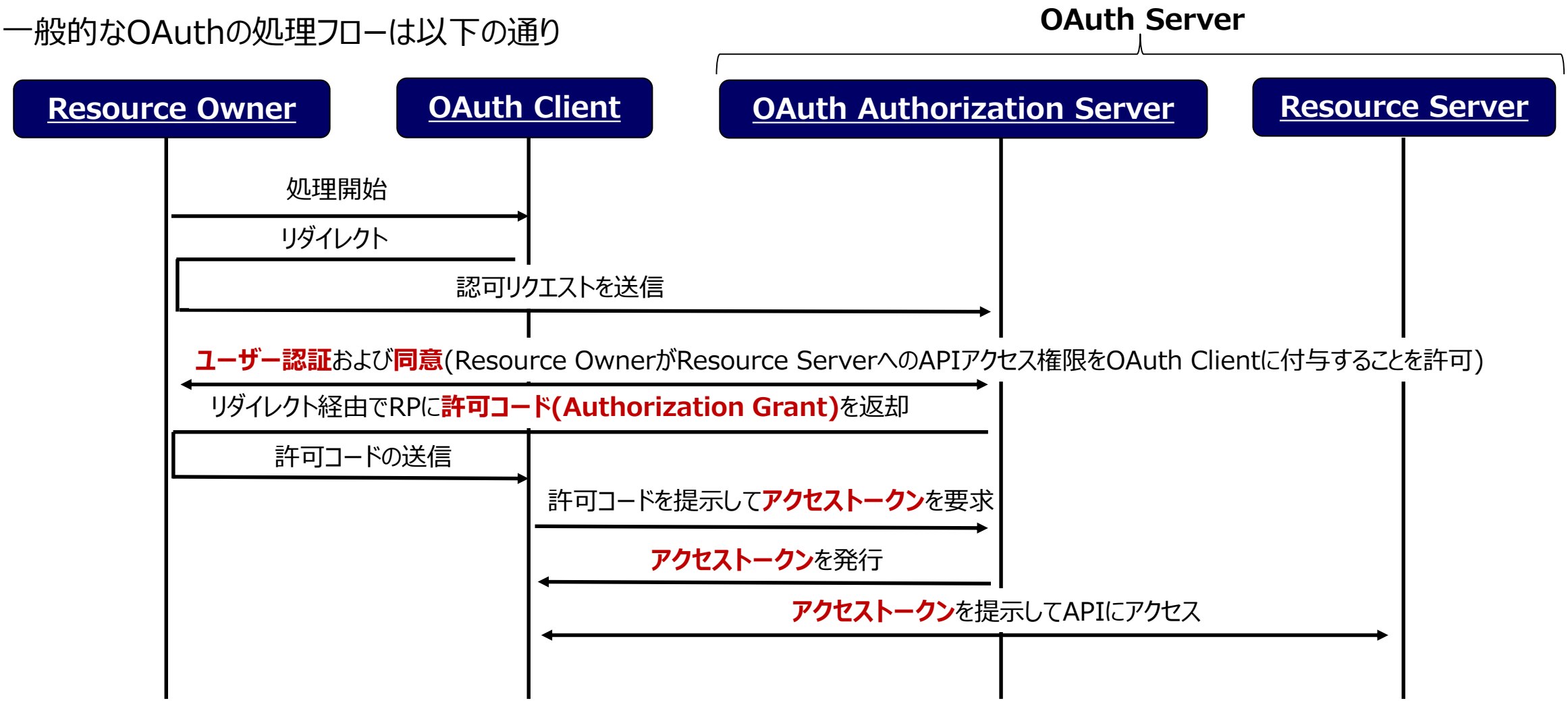
背景: OpenID Connectのベースとなる仕様について　～OAuth～

- Web APIのアクセス認可を行うためのプロトコル
  - RFC 6749 The OAuth 2.0 Authorization Frameworkとして仕様化されている
  - 例)エンドユーザーがInstergramに投稿した写真をInstagramがFacebookやtwitterなどに同時に投稿する



(参考情報)OAuthの処理フロー

- 一般的なOAuthの処理フローは以下の通り



OpenID Connectとは (1)

- OpenID Foundationが策定したOAuth 2.0をベースとするシンプルなアイデンティティ連携プロトコル
- OAuth 2.0およびOpenID 2.0を統合し、両者の機能を継承および拡張している
- OpenID Connect 1.0 > OAuth 2.0 + OpenID 2.0

エンドユーザー

- RPが提供するアプリケーションに対して「○○のIDでログイン」できる
- RP間のシングルサインオンが可能
- アプリケーション毎のユーザーID&パスワード登録が不要
- OAuthの「同意」処理により、ユーザー属性情報を取得するためのAPIアクセス権限を委譲する

(1)RPにアクセス
(4)許可コードを送信
(7)コンテンツを提供

(2)ユーザー認証
(3)同意

Relying Party(RP)

- OpenID Providerにエンドユーザーの認証とユーザー属性情報を要求するOAuth 2.0 Client
- ログイン処理はOPの認証機能を利用する
- OPに対してアクセストークンを提示してユーザー属性情報を取得する

(5)IDトークンおよびアクセストークンの発行
(6)ユーザー属性情報の提供

OpenID Provider(OP)

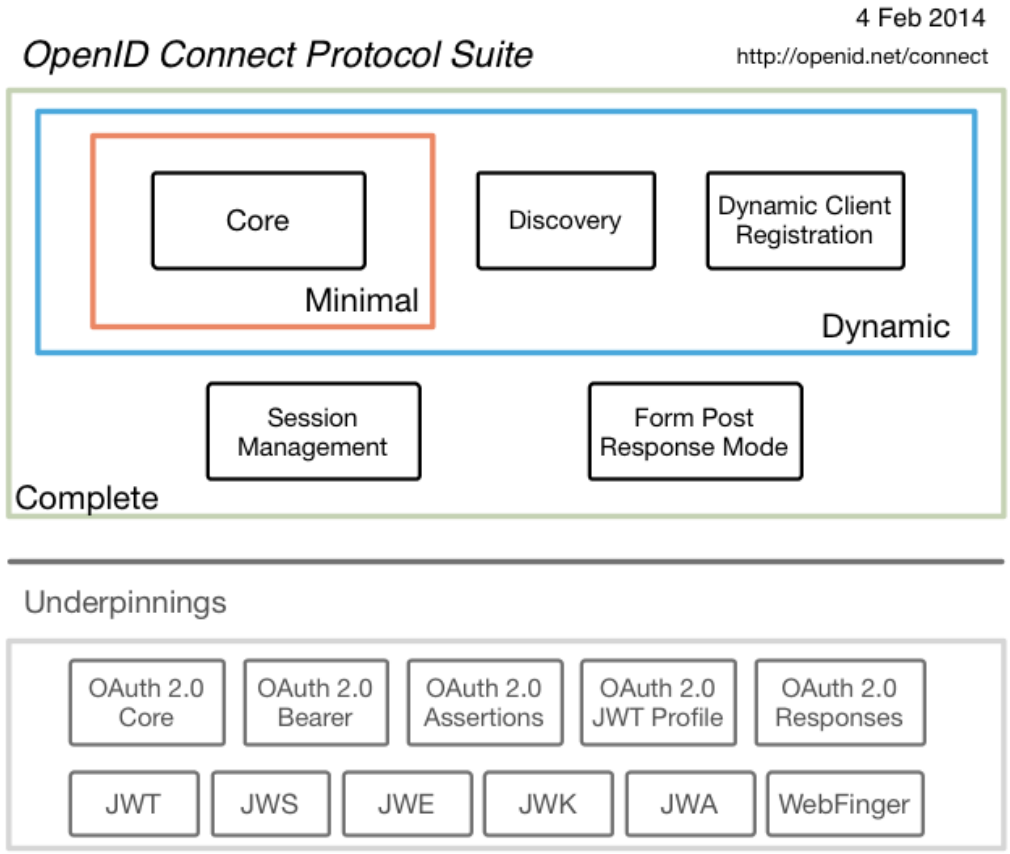
- OpenID ConnectをサポートするOAuth 2.0 Authorization Server
- ユーザー認証機能を提供し、IDトークンやアクセストークンの発行やユーザー属性情報の提供を行う

OpenID Connectとは (2)

- 認証トークン
  - 認証トークンとしてIDトークンを使用する
  - OPによる認証後にIDトークンが発行される
  - IDトークンにはユーザー識別子(sub)や発行者(iss)、発行時間(iat)などの情報が含まれている
  - IDトークンはJSON Web Token(JWT)形式であり、JSON Web Signature(JWS)で署名されている
- ユーザー属性情報の取得
  - アクセストークンを使用してUserInfoエンドポイントから取得する、あるいは、IDトークンに含める
  - JSON形式で表現される
- 「ユーザーセントリック」という設計思想
  - 利用者が中心となって自身のID情報を管理する
  - ユーザーが任意に選択したID提供者とサービス提供者の間で信頼関係を確立する
    - Federated SSOの主要プロトコルの1つであるSAMLと比較した場合
      - SAMLの設計思想は「事業者間で事前に信頼関係を確立する」ため、OpenID ConnectとSAMLは設計思想が異なる
- REST/JSONベースのプロトコル仕様
  - RPはOPが提供するエンドポイントにRESTアクセスすることでJSONベースのIDトークンを取得することができる
  - JSONベースであるためモバイルアプリケーションとの親和性が高い
- Webブラウザーだけではなくネイティブアプリケーション(モバイル/デスクトップアプリ)にも対応している

OpenID Connect 1.0 仕様について

- OpenID Connect 1.0の仕様は2014年2月に最終承認
  - http://openid.net/connect/
- ６つのドキュメントと2つの開発者向けガイドが公開されている
  - Core – Defines the core OpenID Connect functionality: authentication built on top of OAuth 2.0 and the use of Claims to communicate information about the End-User
  - http://openid.net/specs/openid-connect-core-1_0.html
  - Discovery – (Optional) Defines how Clients dynamically discover information about OpenID Providers
  - http://openid.net/specs/openid-connect-discovery-1_0.html
  - Dynamic Registration – (Optional) Defines how clients dynamically register with OpenID Providers
  - http://openid.net/specs/openid-connect-registration-1_0.html
  - Session Management – (Optional) Defines how to manage OpenID Connect sessions, including logout functionality
  - http://openid.net/specs/openid-connect-session-1_0.html
  - OAuth 2.0 Multiple Response Types – Defines several specific new OAuth 2.0 response types
  - http://openid.net/specs/oauth-v2-multiple-response-types-1_0.html
  - OAuth 2.0 Form Post Response Mode – (Optional) Defines how to return OAuth 2.0 Authorization Response parameters (including OpenID Connect Authentication Response parameters) using HTML form values that are auto-submitted by the User Agent using HTTP POST
  - http://openid.net/specs/oauth-v2-form-post-response-mode-1_0.html
  - Basic Client Implementer's Guide – Simple subset of the Core functionality for a web-based Relying Party using the OAuth code flow
  - http://openid.net/specs/openid-connect-basic-1_0.html
  - Implicit Client Implementer's Guide – Simple subset of the Core functionality for a web-based Relying Party using the OAuth implicit flow
  - http://openid.net/specs/openid-connect-implicit-1_0.html



出典:OpenID Connect Protocol Suite http://openid.net/connect/

OpenID Connectへの対応状況

- Google / Microsoft / Salesforce / Paypal / Ping Identity / Yahoo! Japan / mixi などがOpenID Connectに対応している
- IBMはWebSphere Application ServerがOpenID Connectをサポート(詳細は後述)
- その他IBM製品でもサポートが追加されている

- クラウド環境におけるOpenID Connectの活用について
  - Open Standards in Cloud Foundry Identity Services
    - Cloud Foundryがアイデンティティーサービスのオープン・スタンダードとしてOpenID Connectを選定している
    - http://blog.cloudfoundry.org/2013/02/19/open-standards-in-cloud-foundry-identity-services/

3. OpenID Connectの処理フロー

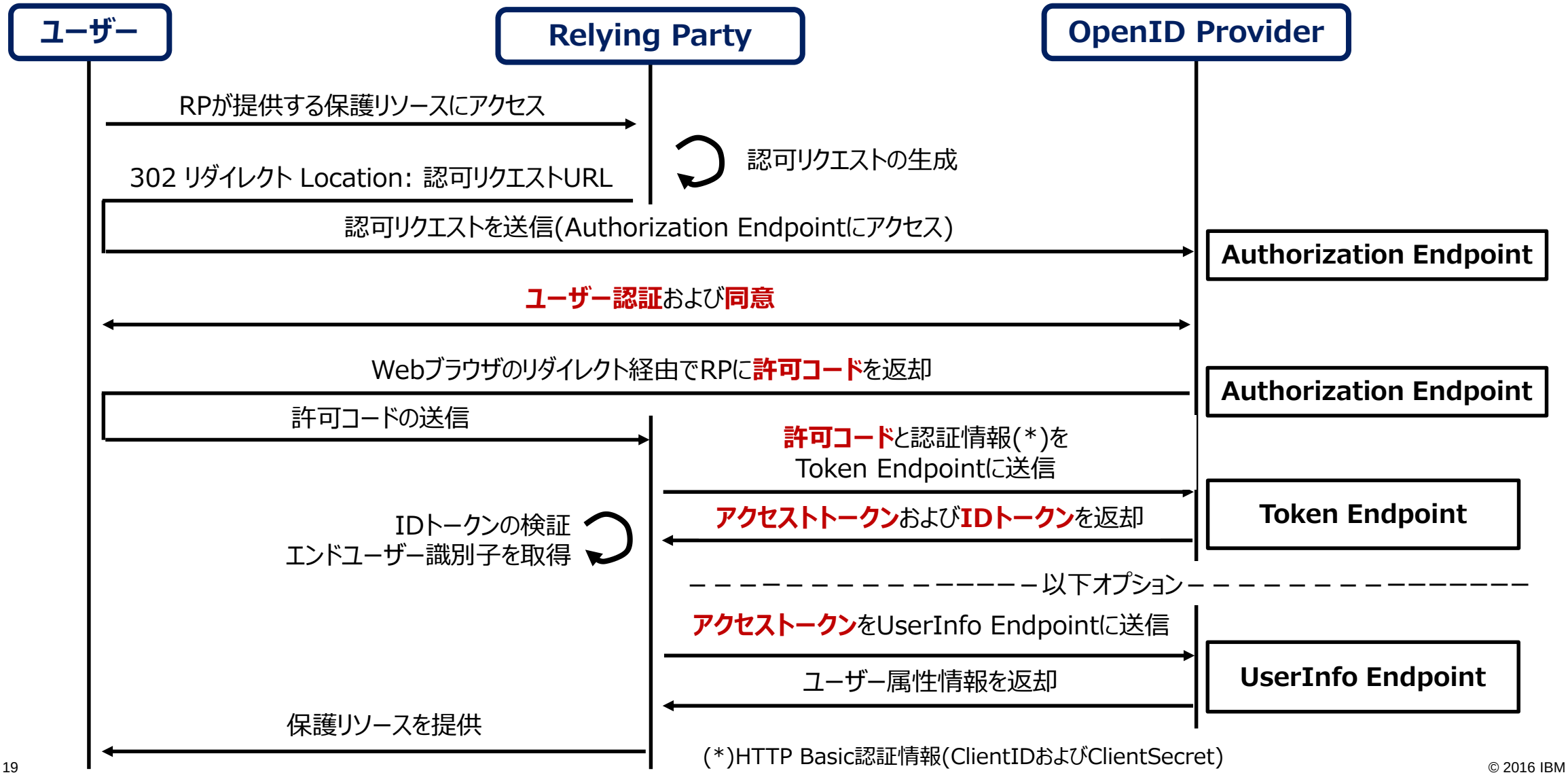
OpenID Connect処理フローの種類

- OpenID Connectには２種類の処理フローが提供されている

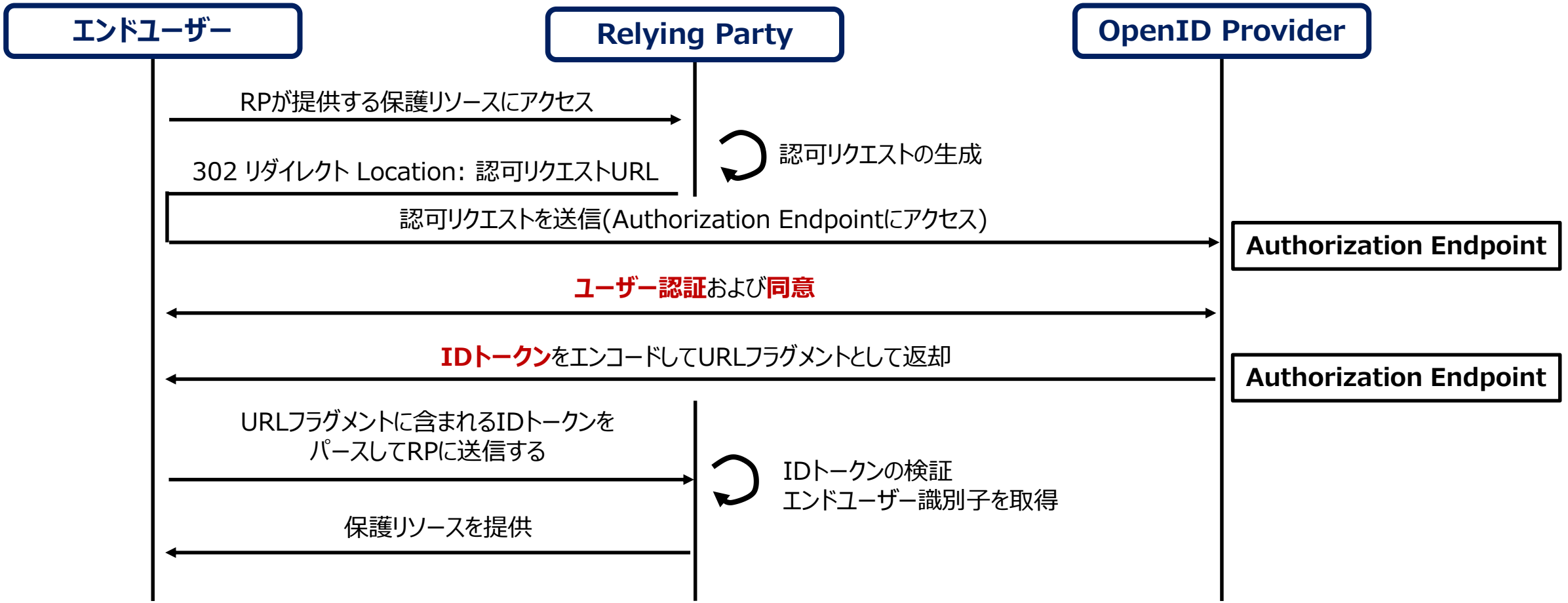
  – 許可コードフロー
    - 許可コードフローはクライアントに許可コードを返し、直接IDトークンおよびアクセストークンと交換することができる
    - User AgentおよびUser Agentにアクセスできる悪意のあるアプリケーションにトークンが渡ることがないというメリットがある
    - 許可コードフローはClient Secretをクライアントと Authorizationサーバー間で安全に保持できるクライアントに適している
    - OPとRPが直接通信を行う
  – Implicitフロー
    - Implicit フローはトークンエンドポイントは使用されず、全てのトークンはAuthorizationエンドポイントから返却される
    - 主にスクリプト言語を用いてブラウザーで実装されたクライアントによって使用される
    - アクセストークンと ID トークンはクライアントに直接返されるため、エンドユーザーおよびUser Agentにアクセスするアプリケーションにそれらのトークンが渡ってしまう可能性がある
    - OPとRPが直接通信を行わない

  – 各フローの詳細は次頁以降を参照

許可コードフロー ： RPとOPが直接通信を行いIDトークンを取得する



Implicitフロー ： RPはOPと直接通信を行わずにユーザー経由でIDトークンを取得する



4. WASにおけるOpenID Connect の実装および構成方法

- WAS V8.5.5 OpenID / OAuth / OpenID Connectの対応状況
- 前提シナリオ
- OpenID Provider側の構成方法(Googleのケース)
- Relying Party側の構成方法(WAS Full Profileのケース)

WAS V8.5.5 OpenID / OAuth / OpenID Connectの対応状況(*)

- WebSphere Application Server の各プロトコルの対応状況は以下のとおり

仕様		Full Profile(従来WAS)	Liberty Profile
OpenID	Relying Party	V8.5.5.3～	V8.5.5.4～
OAuth	OAuth Service Provider	V8.5.0.1～	V8.5.0.2～
OpenID Connect	**Relying Party**	**V8.5.5.3～**	**V8.5.5.4～**
	OpenID Provider	**未対応**	**V8.5.5.4～**

(*)2015年1月現在

- **OpenID Connect Relying Party実装時の注意点**
  - クライアントがWebブラウザーの場合、許可コードフローのみサポートしており、Implicitフローはサポートされない
    - ImplicitフローはIDトークンがURLフラグメントとして提供され、許可コードフローと比較してセキュアではないため
  - Webブラウザーではないクライアントのみ Implicitフローがサポートされる

前提シナリオ

当ガイドでは以下のサンプルシナリオを想定し、OPおよびRPで必要となる構成について記載する

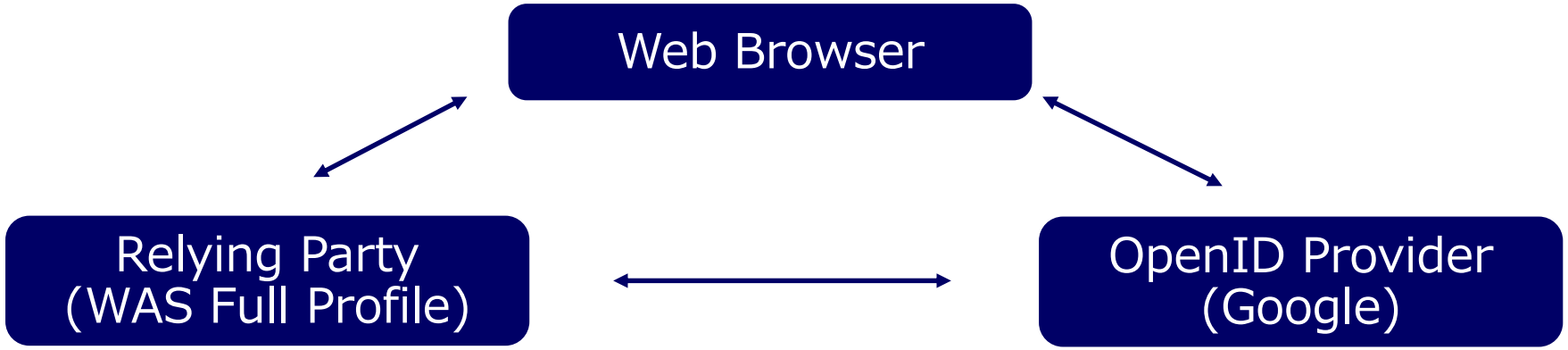
WAS上のWebアプリケーションの認証をGoogleで実施し、ユーザー属性情報をGoogleから取得する

– OpenID Provider：Google

– Relying Party：WebSphere Application Server full profile (従来WAS) 上のWebアプリケーション

- WASはV8.5.5.8以降を前提とする
  http://www-01.ibm.com/support/docview.wss?uid=swg1PI33449
- 当ガイドではDefaultApplicationのsnoopサーブレットを利用

– 使用する処理フロー：許可コードフロー



（参考情報）

- LibertyでGoogleシングルサインオンさせる方法については、下記URLを参照。

- Single sign-on with Google on Liberty
- https://developer.ibm.com/wasdev/docs/single-sign-google-liberty/

- Google OpenID Connect for applications on WebSphere Liberty
- https://developer.ibm.com/wasdev/docs/google-openid-connect-applications-websphere-liberty/

OpenID Connect 構成方法の流れ

- OpenID Connectの構成の流れは下記図のとおり
  - OpenID Provider (Google)の構成
    - OpenID ProviderにRelying Partyを登録する
    - RP側の設定に必要な情報を入手する
    - 具体的な構成方法はOpenID Providerによって異なる
  - Relying Party (WebSphere Application Sever)の構成
    - OpenID Providerに登録したRelying Partyの情報を元に構成を行う

OpenID Provider (Google)の構成

(*)Googleの場合

1. プロジェクトの作成
2. クライアントIDの作成
3. クライアントIDおよびクライアントシークレット、エンドポイントURLの情報取得

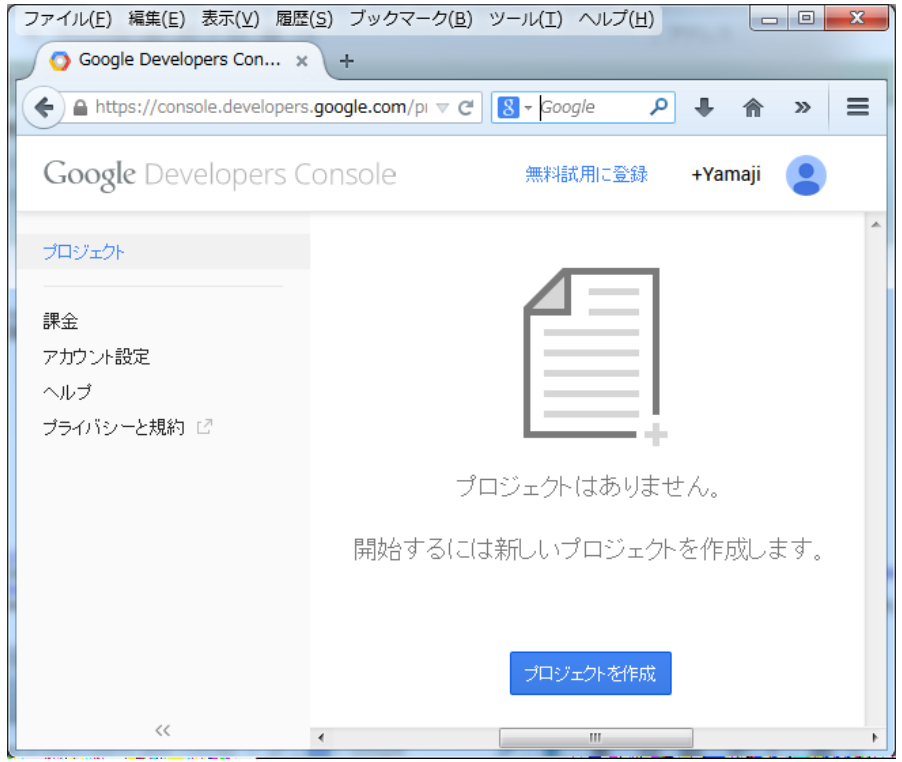
Relying Party (WebSphere Application Server)の構成

1.トラスト・アソシエーションの有効化とOpenID Connect RP用のTAIを作成
2. OpenID Connect RP用TAI カスタム・プロパティーの構成
3. グローバル・セキュリティー カスタム・プロパティーの設定
4. OpenID Providerの署名者証明書の追加
5. OIDCアプリケーションの導入
6.トラステッド認証レルムの追加

1. プロジェクトの作成

OpenID Provider (Google)の構成

- OpenID Connect Provider側でRPを登録してClientIDおよびClientSecretを入手する
  - GoogleがOpenID Connect Providerになる場合、https://console.developers.google.com/ へアクセスする
  - IDおよびパスワードを入力してログインする
  - プロジェクトを作成する
    - [プロジェクトを作成]をクリック
    - プロジェクト名を入力、同意のチェックをして[作成]をクリックする



2. クライアントIDの作成

OpenID Provider (Google)の構成

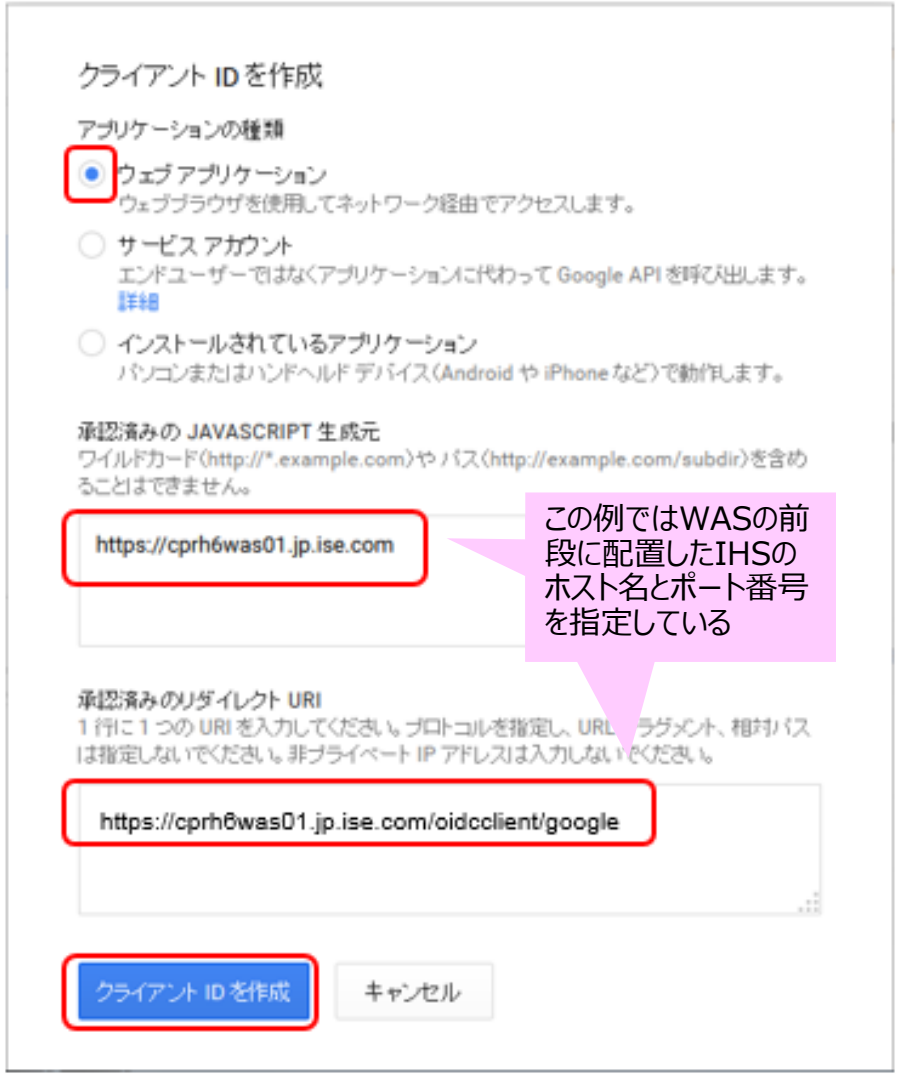
- RP側のWebアプリケーションに対応するクライアントIDを作成する

1. [APIと認証]>[認証情報]を選択し、[新しいクライアントIDを作成]をクリックする
2. [ウェブアプリケーション]にチェックが入っていることを確認して、[同意画面を設定]をクリックする
3. 同意画面の設定画面にて[メールアドレス]および[サービス名（任意）]を入力して[保存]をクリックする

2. クライアントIDの作成

OpenID Provider (Google)の構成

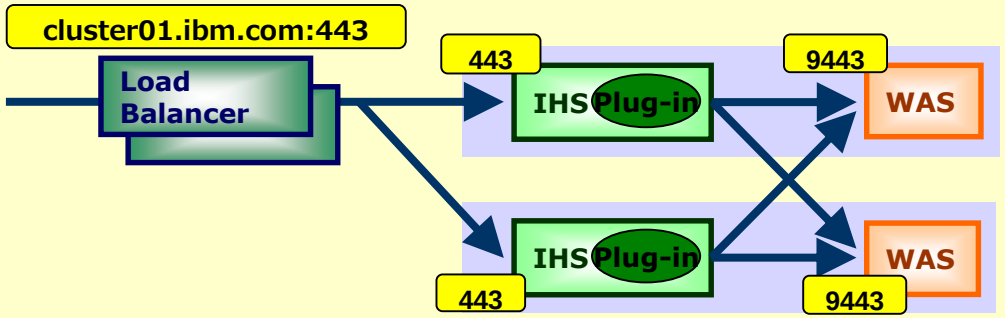
4. [ウェブアプリケーション]が選択されていることを確認する
5. [承認済みのJAVASCRIPT生成元]を指定する(*)
   - RP(WAS)が提供するWebリソースのOriginを指定する
   - 値： https://<WAS_HOSTNAME>:<PORT>
6. [承認済みのリダイレクトURI]を指定する**(*)**
   - Googleの認証および同意処理後のリダイレクト先を指定する
   - 値： https://<WAS_HOSTNAME>:<PORT>/oidcclient/google
7. [クライアントIDを作成]をクリックする



(*)<u>WASの前段にWebサーバーや負荷分散装置が配置されている場合</u>

それらのホスト名およびポート番号を指定する

下図のような構成の場合、ホスト名:ポート番号はcluster01.ibm.com:443を指定する



3. クライアントIDおよびクライアントシークレット、エンドポイントURLの取得

OpenID Provider (Google)の構成

- クライアントIDの情報としてClientIDおよびClientSecretが生成される
- [JSONをダウンロード]をクリックすると以下のような情報を入手できる
- RP側で設定する情報が含まれる
  - **client_id**
    ⇒ ClientID
  - **Client_secret**
    ⇒ ClientSecret
  - **auth_uri**
    ⇒Authorization Endpoint URI
    (許可コードを取得するURI)
  - **token_uri**
    ⇒Token Endpoint URI
    (トークンを取得するURI)

{"web":{**"auth_uri":"https://accounts.google.com/o/oauth2/auth","client_secret":"xxxxxxxxxxxxxxxxxxxx","token_uri":"https://accounts.google.com/o/oauth2/token",**"client_email":"394499933918-6166v1bfaik7gcd5g1hqu19jh9v7ha7q@developer.gserviceaccount.com","redirect_uris":["https://uchiwa.makuhari.japan.ibm.com:9472/oidcclient"],"client_x509_cert_url":"https://www.googleapis.com/robot/v1/metadata/x509/394499933918-6166v1bfaik7gcd5g1hqu19jh9v7ha7q@developer.gserviceaccount.com",**"client_id":"xxxxxxxxxxxxxxxxxxx.apps.googleusercontent.com"**,"auth_provider_x509_cert_url":"https://www.googleapis.com/oauth2/v1/certs","javascript_origins":["https://uchiwa.makuhari.japan.ibm.com:9472"]}}

0. WASの前提構成

Relying Party(WAS)の構成

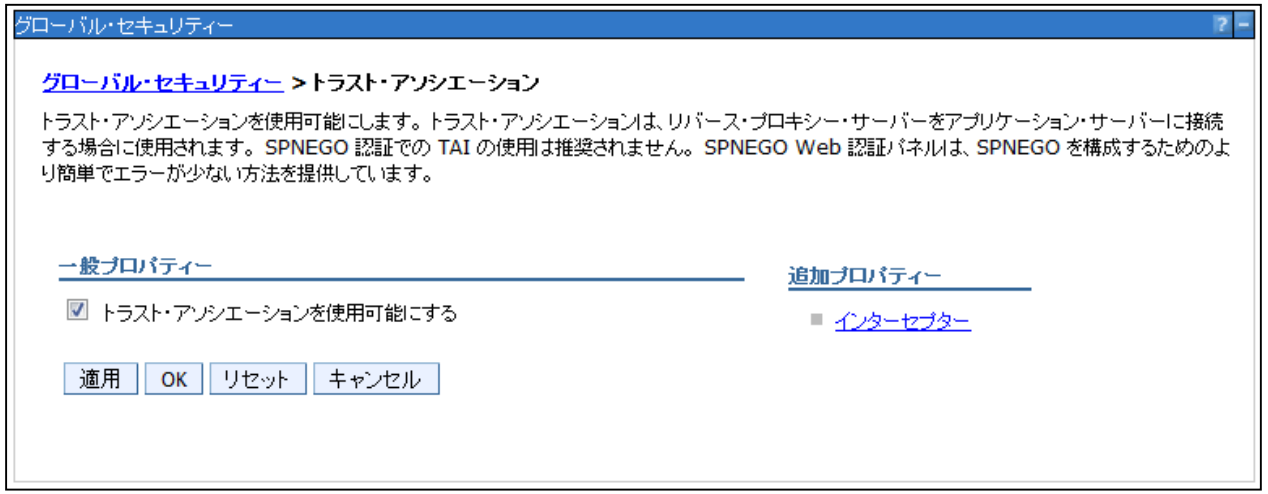
- 前提となるWASの設定は以下のとおり。構成していない場合は事前に構成を行う。
  - グローバル・セキュリティーの有効化
  - 管理セキュリティーの有効化
  - アプリケーション・セキュリティーの有効化
    - アプリケーション(EAR/WAR)とWASランタイムのセキュリティー・ロールのマッピングも必要
  - ユーザーレジストリーの構成(オプション)

1.トラスト・アソシエーションの有効化とOpenID Connect RP用のTAIを作成

Relying Party(WAS)の構成

トラスト・アソシエーションの有効化

1. 管理コンソールにログインする
2. [セキュリティー]>[グローバル・セキュリティー]>[WebおよびSIPセキュリティー]>[トラスト・アソシエーション]を選択する
3. [トラスト・アソシエーションを使用可能にする]のチェックボックスにチェックを入れて[OK]ボタンをクリックする



OpenID Connect RP用のインターセプターの作成

1. [セキュリティー]>[グローバル・セキュリティー]>[WebおよびSIPセキュリティー]>[トラスト・アソシエーション]を選択する
2. [インターセプター]を選択して[新規作成]ボタンをクリックし、インターセプター・クラス名を入力して[OK]ボタンをクリックする
   - インターセプター・クラス名
     com.ibm.ws.security.oidc.client.RelyingParty

2. OpenID Connect RP用TAI カスタム･プロパティーの構成

Relying Party(WAS)の構成

TAIのカスタム･プロパティーを追加する

1. 作成したOpenID Connect用のインターセプターを選択し、カスタム･プロパティーを追加して、[適用]ボタンをクリックしてから保存する

名前	値	説明
callbackServletContext	/oidcclient	
provider_1.clientId	xxxxxxxxxx.apps.googleusercontent.com	OpenID Providerから発行されたclientIDを指定する
provider_1.clientSecret	{xor}xxxxxxxxxxxxx	OpenID Providerから発行されたclientSecretの値をプレーンテキストあるいはXORエンコードした値を指定する
provider_1.authorizeEndpointUrl	https://accounts.google.com/o/oauth2/auth	OpenID Provider側で指定された許可コードを取得するためのエンドポイントURLを指定する
provider_1.tokenEndpointUrl	https://www.googleapis.com/oauth2/v3/token	OpenID Provider側で指定されたトークンを取得するためのエンドポイントURLを指定する
provider_1.scope	openid email profile	OPから取得したいクレーム(ユーザー属性情報)をスペース区切りで指定する。指定いない場合はデフォルトでopenidが設定される。openidの指定は必須。(詳細はHint&Tipsの章を参照)
provider_1.interceptedPathFilter	(.*)googleAuth(.*), /snoop	TAIでインターセプトする対象のリクエストURIを指定する カンマ(,)区切りで複数指定可能
provider_1.identifier	google	

次頁に続く･･･

2. OpenID Connect RP用TAI カスタム･プロパティーの構成

Relying Party(WAS)の構成

TAIのカスタム･プロパティーを追加する(続き)

前頁の続き･･･

名前	値	説明
provider_1.jwkEndpointUrl	https://www.googleapis.com/oauth2/v2/certs	OPの署名を検証するために必要となる公開鍵を取得するエンドポイントURLを指定する。JWKはJSON Web Keyの略。
provider_1.issuerIdentifier	accounts.google.com	IDトークンの発行者を指定する。デフォルトではauthorizationEndpointから得られたissureIdentifierが設定される。
provider_1.signatureAlgorithm	RS256	OPからのメッセージを保護するためのアルゴリズムを指定する。
provider_1.userIdentifier	email	RP側のユニークなユーザーIDとして使用するクレーム(属性)を指定する。
provider_1.mapIdentityToRegistryUser	false	OpenID Connectの認証済みユーザーをWAS(RP)のユーザーレジストリー内のユーザーとマッピングすることが可能。デフォルトはfalseでマッピングを行わない。

- その他のTAIカスタム･プロパティーは下記URLを参照

IBM Knowledge Center > WAS ND 8.5.5 > OpenID Connect Relying Party custom properties
http://www.ibm.com/support/knowledgecenter/en/SSAW57_8.5.5/com.ibm.websphere.nd.doc/ae/csec_oidprop.html?pos=2

2. OpenID Connect RP用TAI カスタム･プロパティーの構成

Relying Party(WAS)の構成

- Google OPの場合のTAIカスタム･プロパティーの一覧は以下の通り

一般プロパティー

＊ インターセプター・クラス名

com.ibm.ws.security.oidc.client.RelyingParty

カスタム・プロパティー

新規 編集 削除

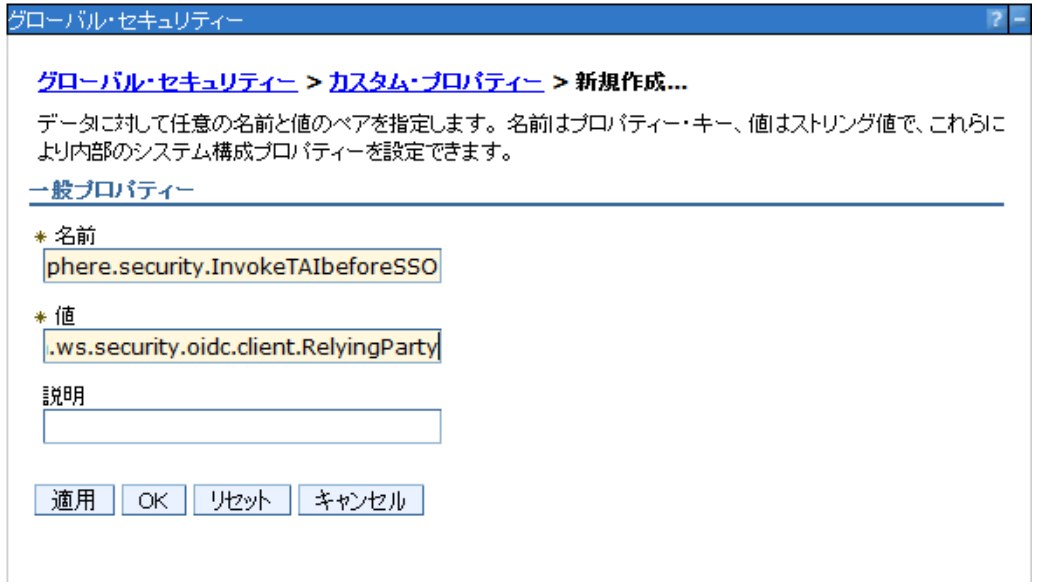
選択	名前	値
☐	callbackServletContext	/oidcclient
☐	provider_1.clientId	394[redacted]a7q.apps.googleusercontent.com
☐	provider_1.clientSecret	6D[redacted]E18
☐	provider_1.authorizeEndpointUrl	https://accounts.google.com/o/oauth2/auth
☐	provider_1.tokenEndpointUrl	https://www.googleapis.com/oauth2/v3/token
☐	provider_1.scope	openid email profile
☐	provider_1.interceptedPathFilter	(.*)googleAuth(.*), /snoop
☐	provider_1.identifier	google
☐	provider_1.jwkEndpointUrl	https://www.googleapis.com/oauth2/v2/certs
☐	provider_1.issuerIdentifier	accounts.google.com
☐	provider_1.signatureAlgorithm	RS256
☐	provider_1.userIdentifier	email
☐	provider_1.mapIdentityToRegistryUser	false

適用 OK リセット キャンセル

3. グローバル・セキュリティー カスタム・プロパティーの設定

Relying Party(WAS)の構成

1. [セキュリティー]>[グローバル・セキュリティー]>[カスタム・プロパティー]を選択する
2. [新規作成]ボタンをクリックして、以下のカスタム・プロパティーを追加する
   - 名前：com.ibm.websphere.security.InvokeTAIbeforeSSO
   - 値：com.ibm.ws.security.oidc.client.RelyingParty



3. [OK]ボタンをクリックする

4. OpenID Providerの署名者証明書の追加

Relying Party(WAS)の構成

- OpenID ProviderのSSL署名者証明書をWASのトラストストアに追加する

＜ポートから取得する場合＞

1. [セキュリティー]>[SSL証明書および鍵管理]>[鍵ストアおよび証明書]>[NodeDefaultTrustStore]>[署名者証明書]を選択する
   - DM環境の場合はCellDefaultTrustStoreを選択する
2. [ポートから取得]ボタンをクリックして、ホスト名およびポート番号、別名を指定して[署名者情報の取得]ボタンをクリックする
   - OP側の各エンドポイントにアクセスする際のホスト名およびポート番号を入力する。別名は任意の名前を入力する。
   - Googleの場合はホスト名: accounts.google.com / ポート番号: 443　を指定する。

5. OIDCアプリケーションの導入

Relying Party(WAS)の構成

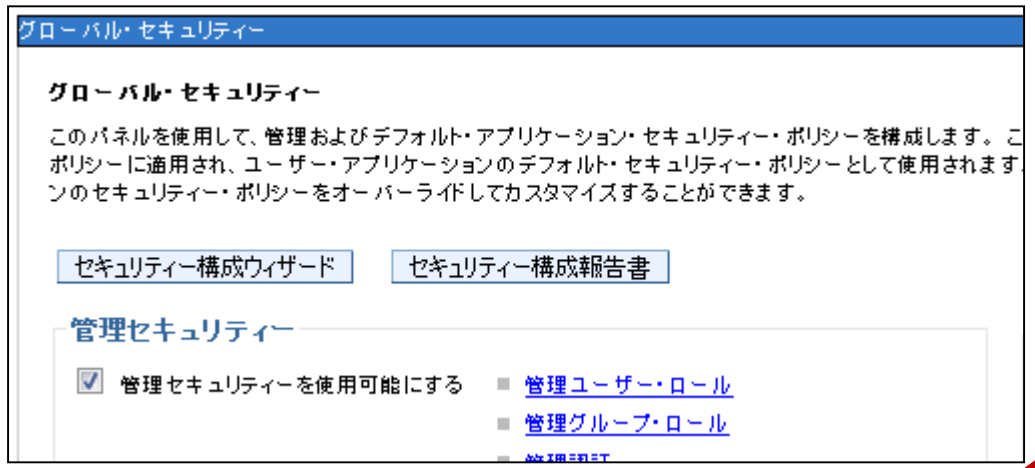
1. OIDCアプリケーションをインストールするためのスクリプトを実行する(<Install_root>/bin/installOIDCRP.py)
   - installOIDCRP.pyの実行例

```
# <Profile_root>/bin/wsadmin.sh –f <Install_root>/bin/installOIDCRP.py install  <Node_name> <Server_name> -username xxxxx –password xxxxx
WASX7209I: ノード OidcBaseNode01 のプロセス "server1" に、SOAP コネクターを使用して接続しました。プロセスのタイプは UnManagedProcess です。
WASX7303I: 次のオプションはスクリプト環境に渡され、argv 変数に格納される引数として使用可能になります: "[install, OidcBaseNode01, server1]"
Installing OpenID Connect relying party...
Deploying WebSphereOIDCRP.ear
ADMA5016I: WebSphereOIDCRP のインストールが開始されました。
ADMA5058I: アプリケーションとモジュールのバージョンが、デプロイメント・ターゲットのバージョンに対して検証されます。
ADMA5005I: アプリケーション WebSphereOIDCRP が WebSphere Application Server リポジトリーに構成されます。
ADMA5005I: アプリケーション WebSphereOIDCRP が WebSphere Application Server リポジトリーに構成されます。
ADMA5081I: クライアント・モジュールのブートストラップ・アドレスが WebSphere Application Server リポジトリーの中に構成されます。
ADMA5053I: インストール済みオプション・パッケージのライブラリー参照が作成されます。
ADMA5005I: アプリケーション WebSphereOIDCRP が WebSphere Application Server リポジトリーに構成されます。
ADMA5001I: アプリケーション・バイナリーは
/usr/IBM/WebSphere/AppServer/profiles/OidcBase01/wstemp/Script529e2ca8/workspace/cells/OidcBaseCell01/applications/WebSphereOIDCRP.ear/WebSphe
reOIDCRP.ear に保存されます。
ADMA5005I: アプリケーション WebSphereOIDCRP が WebSphere Application Server リポジトリーに構成されます。
SECJ0400I: アプリケーション WebSphereOIDCRP が appContextIDForSecurity 情報で正常に更新されました。
ADMA5005I: アプリケーション WebSphereOIDCRP が WebSphere Application Server リポジトリーに構成されます。
ADMA5005I: アプリケーション WebSphereOIDCRP が WebSphere Application Server リポジトリーに構成されます。
ADMA5113I: アクティベーション・プランが正常に作成されました。
ADMA5011I: アプリケーション WebSphereOIDCRP の一時ディレクトリーのクリーンアップが完了しました。
ADMA5013I: アプリケーション WebSphereOIDCRP は正常にインストールされました。
in loop see com.ibm.wsspi.security.web.webAuthReq(cells/OidcBaseCell01|security.xml#DescriptiveProperty_8)
persisting modified
#
```

6. トラステッド認証レルムの追加

Relying Party(WAS)の構成

1. 管理コンソールにて、[セキュリティー]>[グローバル・セキュリティーー]>[ユーザー・アカウント・リポジトリー]>[構成]>[トラステッド認証レルム - インバウンド]を選択して「外部レルムの追加」をクリックする
2. 外部レルムとして"accounts.google.com"を入力して「OK」ボタンをクリックする。
3. 設定を保存して、アプリケーションサーバーを再起動する。



～中略～

5. Hint&Tips

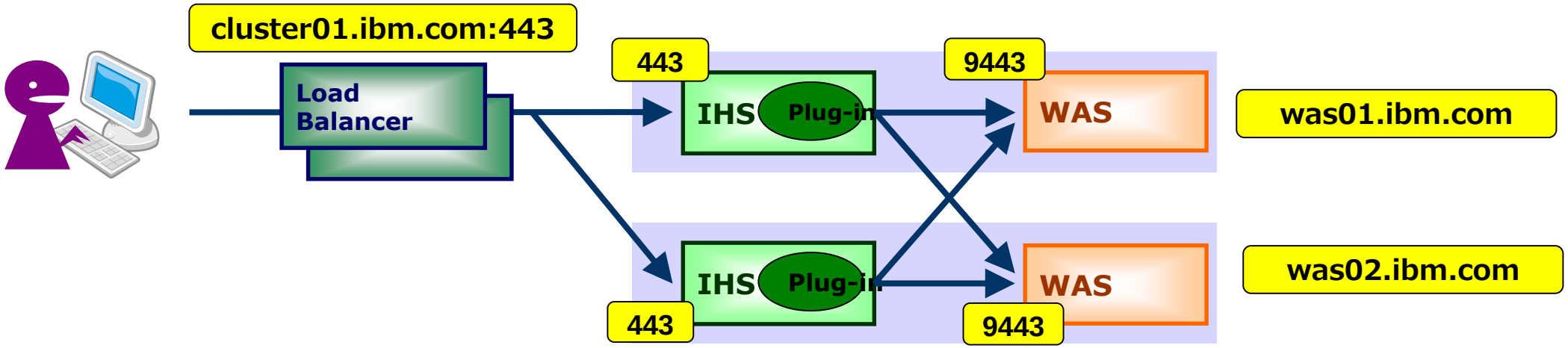
ユーザー属性情報の受け渡しとscopeの設定について

- RPがOPに要求するユーザー属性情報の指定についてはscopeパラメーターにOP が用意した「ユーザ属性のセット」を指定する
  - RP側でscopeの指定が必要となる(WASは認可リクエストを生成する際にカスタム・プロパティー”scope”の値を使用する)
    - [セキュリティー]>[グローバル・セキュリティー]>[WebおよびSIPセキュリティー]>[トラスト・アソシエーション]>[インターセプター]>[com.ibm.ws.security.oidc.client.RelyingParty]で下記のカスタムプロパティーを追加する
      - 名前: provider_<id>.scope
      - 値(例): openid email profile
- scopeについて
  - OAuth 2.0で定義されているパラメーターであり、scopeにopenidが指定されていればOpenID Connectのリクエストであると判断される。そのため、openidの指定は必須となる。
  - 仕様で定義されているscope値は以下のとおり

名前	説明
profile	デフォルトのプロフィールクレーム。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）、referred_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）

WASの前段に負荷分散装置やWebサーバーが配置されている場合

- WASの前段にIHSを配置するケースや負荷分散装置経由でクラスターIPアドレスでアクセスする場合に必要な設定について
  - RP(WAS)側の設定
    - [セキュリティー]>[グローバル・セキュリティー]>[WebおよびSIPセキュリティー]>[トラスト・アソシエーション]>[インターセプター]>[com.ibm.ws.security.oidc.client.RelyingParty]で下記のカスタムプロパティーを追加して、ホスト名とポート番号を明示的に指定する必要がある
      - 名前: provider_<id>.redirectToRPHostAndPort
      - 値: https://cluster01.ibm.com:443
    - ” provider_<id>. redirectToRPHostAndPort”を指定しない場合、WASのホスト名およびポート番号でredirectURLを自動生成するため
      - 例: https://was01.ibm.com:9443
  - OP側の設定
    - [承認済みのJAVASCRIPT生成元]および[承認済みのリダイレクトURI]のホスト名およびポート番号もRP側の設定と一致している必要がある



WAS ユーザーレジストリーとのマッピング

- OpenID Connectの認証済みユーザーをWAS(RP)のユーザーレジストリー内のユーザーとマッピングすることが可能
  - マッピングを行うケース
    - RP側の実装としてWASのユーザーレジストリーの構成が前提となっており、OPとRPのユーザーをマッピングする必要がある場合
    - RP側のアプリケーションで必要なユーザー情報をOPが提供できない場合
      - 例えばLDAPのグループメンバーシップなどの情報をWASのユーザーレジストリーに登録しておけば、必要に応じてLDAPからデーターを検索することができるようになる
      - ユーザーがWASのユーザーレジストリ側に登録されている必要がある
  - マッピングを行わないケース(デフォルト)
    - OP側で提供されるユーザー情報のみでRP側のアプリケーションが稼働できる場合
      - WASのユーザーレジストリーを構成する必要はない
      - OPのユーザーおよびグループ情報でWebSphere Subjectが作成される

- 設定方法
  - 管理コンソールより[セキュリティー]>[グローバル・セキュリティー]>[WebおよびSIPセキュリティー]>[トラスト・アソシエーション]>[com.ibm.ws.security.oidc.client.RelyingParty]を選択して下記のカスタムタム・プロパティーを追加する
    - 名前： provider_<id>.mapIdentityToRegistryUser
    - 値：true （デフォルトはfalse)

  - ユーザーIDとして扱う属性の設定方法
    - デフォルトではIDトークンに含まれる"sub"をRP側のユーザーIDとして扱う
    - TAIカスタム・プロパティー"provider_<id>.userIdentifier"にユーザー識別子としてマップするIDトークン内のクレーム(属性)を指定することが可能
    - OPがGoogleの場合はprovider_<id>.userIdentifier = emailを指定する

OpenID Connectの認証後にLTPAトークンの発行がされるかどうか

- ユーザーレジストリーとマッピングを行わない場合(mapIdentityToRegistryUserがfalseの場合)
  - WASのユーザーレジストリーにユーザーが存在する／しないに関わらず LTPAトークンが発行される

- ユーザーレジストリーとマッピングを行う場合(mapIdentityToRegistryUserがtrueの場合)
  - ユーザーレジストリーにユーザーが存在する場合はLTPAトークンが発行される
  - ユーザーが存在しない場合はエラーとなる

トラブルシューティング

- OpenID Connectに関して問題があった場合、サポート部門の指示に従い必要に応じてトレース取得を行う

- OpenID Connect関連のトレースストリングは以下の通りだが、サポート部門が指示するトレースストリングを設定すること

```
*=info:com.ibm.ws.security.oidc.*=all:com.ibm.ws.webcontainer.srt.SRTServletRequest =all
```

補足資料

OpenID Connect の用語 (1/2)

用語	説明
OpenID Provier(OP)	OAuth 2.0 認可サーバーとしてクライアントやRPにクレームを提供する。ユーザーIDを発行し，ユーザーを認証する。ユーザーに関する属性情報(氏名，メール・アドレス等)を保持し，RPからの要求に応じて提供する。
Relying Party(RP)	OPに対してクレームを要求するクライアントアプリケーション。サービスを提供するアプリケーションであり、OPに認証を委ねてユーザーを識別する。
Authorization code(許可コード)	アクセストークンを取得するために使用されるクレデンシャル。許可コードフローで利用され、Implicitフローでは利用されない。
ID token(IDトークン)	OPが発行する認証済みユーザーに関するセキュリティートークン。JSON Web Token (JWT)形式。
Access token(アクセストークン)	OPが発行する保護リソースにアクセスするためのクレデンシャル。OpenID Connectではアクセストークンを使用してOPからユーザー属性情報を取得する。
Refresh token(リフレッシュトークン)	OPが発行するトークン。アクセストークンの期限切れや追加取得のために新しいアクセストークンを取得する際に使われる。
Claim(クレーム)	ユーザーに関する情報の部分集合。JSON形式で表現される。例：profile（一般的なユーザ属性）、email（メールアドレス）、address（住所）、phone（電話番号）など
Scope(スコープ)	OAuth 2.0で定義されており、要求するアクセス権限を指定する。OpenID ConnectではScopeにClaimとして取得可能な情報を指定するために利用する。Authorization EndpointへアクセスするURLのクエリーパラメーターとしてにScopeを含める必要がある。

OpenID Connect の用語 (2/2)

用語	説明
Authorization Endpoint	ユーザーの認証認可を行うためにクライアントから送られる認可リクエストを受け付けるためのOP上のリソース。Authorization Endpointは許可コードフローではRPに許可グラント(許可コード）を返却し、インプリシットフローではIDトークンとアクセストークンを返却する。
Token Endpoint	アクセストークンおよびIDトークン、リフレッシュトークンを取得するためにRPからの許可グラント（許可コード）を受け付けるためのOP上のリソース。
UserInfo Endpoint	OPがRPにユーザー情報(クレーム)を提供するためのOP上のリソース。

IDトークンについて

- IDトークンはJSON Web Token(JWT)形式で表現される
- IDトークンの文字列はピリオド(.)で区切られた3つの文字列で構成されている
- [JWTヘッダ].[IDトークン].[署名]

eyJhbGciOiJSUzI1NiIsImtpZCI6IjdhOTUxMmY2MmFiMTljNzkwZmI1NDBiOGEwNTMzZjU5MzcyZjI1MWMifQ.eyJpc3MiOiJhY2NvdW50cy5nb29nbGUuY29tIiwic3ViIjoiMTEwNzkyNzkyMTY1MjkzMzc3NzczIiwiYXpwIjoiMTk1MzE4NjcyMjIwLTY5YWN1bjgwMmhjdXVmbTE1NWZvMmptOW5scTFvaTFiLmFwcHMuZ29vZ2xldXNlcmNvbnRlbnQuY29tIiwibm9uY2UiOiJuLTBTNl9XekEyTWoiLCJhdF9oYXNoIjoiNllpMmlYekJyYmRzYmFCaERsbXBMQSIsImF1ZCI6IjE5NTMxODY3MjIyMC02OWFjdW44MDJoY3V1Zm0xNTVmbzJqbTlubHExb2kxYi5hcHBzLmdvb2dsZXVzZXJjb250ZW50LmNvbSIsImlhdCI6MTQxMzk1Nzk5MiwiZXhwIjoxNDEzOTYxODkyfQ.UirTat9Vjd7CFU4y4WFJg9ECcfwQX1DszVTfhjUwK-Q2gm4pCophRxMOOVBMqlzjZLM6-c2A7HiVNbK24BmXSuLUxeuYTDyhRXHyP1D7FQttbd1J4wD_XOCEVSMHytrqOBpoTGzpX_LZWgyrgg9pCNHmiIy9eDCAkzxPWWImfLQ

- [JWTヘッダ]をBase64 URLデコードすると

```
{   "alg":"RS256",
    “kid”:“7a9512f62ab19c790fb540b8a0533f59372f251c“ }
```

- [IDトークン]をBase64 URLデコードすると

```
{   "iss":"accounts.google.com",
    "sub":"110792792165293377773",
    "azp":"195318672220-69acun802hcuufm155fo2jm9nlq1oi1b.apps.googleusercontent.com",
    "nonce":"n-0S6_WzA2Mj",
    "at_hash":"sTHJbOexOh-76vhM7I0PXw",
    "aud":"195318672220-69acun802hcuufm155fo2jm9nlq1oi1b.apps.googleusercontent.com",
    "iat":1413955531,
    "exp":1413959431 }
```

OpenID ConnectとSAMLの比較

- OpenID ConnectおよびSAMLの技術的な仕様の違い

	SAML	OpenID Connect
設計思想	事業者中心（事前の信頼関係に基づく）	ユーザー中心（ユーザーの同意に基づく）
プロトコル/データフォーマット	SOAP/XMLベース	REST/JSONベース
WebAPI アクセス認可	不可能 (OAuthと連携することで実現可能)	可能
認証トークン形式	SAML Assertion	JSON Web Token
署名/暗号化	XML署名/暗号化	JSON Web Signiture / Encryption
属性情報の受け渡し	任意の属性を受け渡し可能	任意の属性を受け渡し可能
実装の容易性	難しい（ミドルウェア製品を利用）	容易

参考資料

- 仕様
  - OpenID Connect
  - http://openid.net/connect/
  - http://www.openid.or.jp/
- WAS関連
  - Knowledge Center
  - http://www-01.ibm.com/support/knowledgecenter/SSAW57_8.5.5/as_ditamaps/was855_welcome_ndmp.html
  - PI14470: ADD OPENID AND OPENID CONNECT RELYING PARTY TAIS
  - http://www-01.ibm.com/support/docview.wss?uid=swg1PI14470
  - IBM Education Assistant WebSphere Application Server Version: V8.5.5.3
  - http://www-01.ibm.com/support/knowledgecenter/websphere_iea/com.ibm.iea.was_v8/was/8.5.5.3/content_list.html
  - IBM Education Assistant WebSphere Application Server Version: V8.5.5.4
  - http://www-01.ibm.com/support/knowledgecenter/websphere_iea/com.ibm.iea.was_v8/was/8.5.5.4/content_list.html
  - PI47460: Add multi-provider support to OpenID Connect Relying Party in the full profile
  - http://www-01.ibm.com/support/docview.wss?uid=swg24041056
  - PI33449: FULL PROFILE OPENID CONNECT RP DOES NOT WORK WITH GOOGLE OP
  - http://www-01.ibm.com/support/docview.wss?uid=swg1PI33449
- その他
  - Google Identity Platform – OpenID Connect
  - https://developers.google.com/identity/protocols/OpenIDConnect