

OIDC Overview

OIDC primer - a course on OpenID Connect







Introduction to OpenID Connect

OIDC is an open standard published by the **OpenID Foundation** in 02/2014.

As an open standard it can be implemented without license or intellectual property concerns.





















(*) Sustaining Corporate Members.

Introduction to OpenID Connect

OIDC defines an interoperable way to perform **user authentication**.

- 1. Clients can **verify the identity** of the end-user based on the authentication performed by an Authorization Server;
- 2. It allows clients to **obtain basic profile information** about the end-user in an interoperable and REST-like manner.

The actors involved

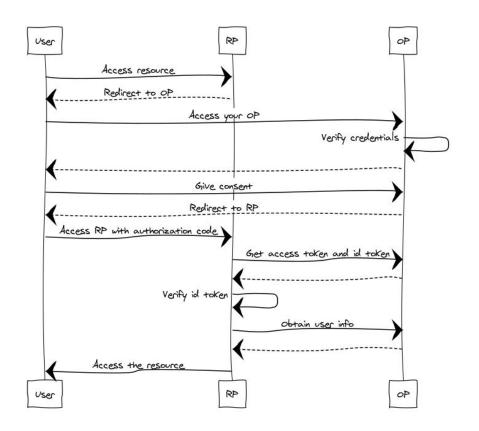
- 1. The **User** is someone trying to access a protected resource.
- 2. The **Relying Party** (or Client) is the entity that requests, receives and uses tokens. The RP can be any of a web application, a native application or mobile application.
- 3. The **OpenID Provider** is the entity that releases tokens. The OP is usually a web based server that is able to receive and process requests for tokens from RPs.

Authentication flows

OpenID supports three flows to authenticate a user and retrieve ID token:

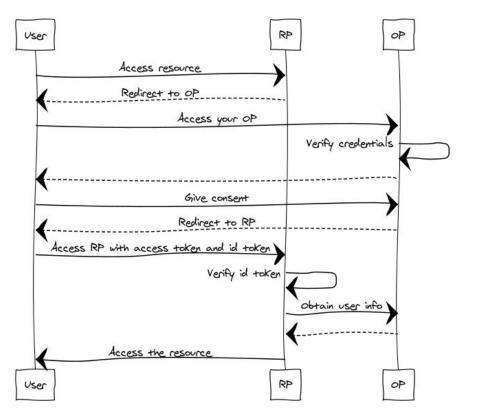
- 1. **Authorisation code flow** the most commonly used flow, intended for traditional web apps as well as native/mobile apps. This flow offers optimal security, as tokens are not revealed to the browser and the client app can also be authenticated.
- 2. **Implicit flow** for browser (JavaScript) based apps that don't have a backend. The ID token is received directly with the redirection response from the OP. No back-channel request is required here.
- 3. **Hybrid flow** rarely used, allows the app front-end and back-end to receive tokens separately from one another. Essentially a combination of the code and implicit flows (not shown in the following).

Authorization code flow



- 1. User access resource on RP.
- The RP redirect the user to the OP for authentication.
- Client sends an Authentication Request containing the desired request parameters to the OP.
- OP Server Authenticates the End-User by checking credentials.
- Authorization Server obtains End-User Consent/Authorization.
- 6. Authorization Server sends the End-User back to the Client with an Authorization Code.
- 7. Client requests a response using the Authorization Code at the Token Endpoint.
- 3. Client receives a response that contains an ID Token and Access Token in the response body.
- 9. Client validates the ID token and retrieves the End-User's Subject Identifier.

Implicit flow



- 1. Client prepares an Authentication Request containing the desired request parameters.
- Client sends the request to the Authorization Server.
- 3. Authorization Server Authenticates the End-User.
- Authorization Server obtains End-User Consent/Authorization.
- 5. Authorization Server sends the End-User back to the Client with an ID Token and, if requested, an Access Token.
- 6. Client validates the ID token and retrieves the End-User's Subject Identifier.

OpenID endpoints

The endpoints defined in the standard are:

- Authorize endpoint: this endpoint performs authentication and authorisation.
- **Token endpoint**: this endpoint allows the requester to get his tokens. If the authorize endpoint is human interaction, this endpoint is machine to machine interaction.
- UserInfo endpoint: this endpoint allows you to make a request using your access token to receive claims about the authenticated end-user

Optional endpoints are:

- Discovery: this endpoint provide metadata about the OpenID Connect provider, allowing applications to automatically configure for that provider.
- **Client Registration**: this endpoint allow a relaying party to register with the OpenID provider.

ID token

The OpenID Connect ID token is a **signed token** given to the client application. The ID token is directed to the RP and is intended to be parsed by it.

The ID token **contains a set of claims** about the authentication session, including:

- an identifier for the user (sub),
- the identifier for the identity provider that issued the token (iss), and
- the identifier of the client for which this token was created (aud).

ID token

The OpenID Connect ID token is a **signed token** given to the client application.

The ID token is directed to the RP and is intended to be parsed by it.

The ID token **contains a set of claims** about the authentication session, including:

- an identifier for the user (sub),
- the identifier for the identity provider that issued the token (iss),
- the identifier of the client for which this token was created (aud),
- the expiration time (exp),
- the time of issuing (iat),
- the authentication time (auth_time), and
- a value used to associate a Client session, to mitigate replay attacks (nonce).

Claims and Scopes

OpenID Connect specifies a set of **standard claims**, or user attributes.

They are intended to supply the client app with consented user details such as email, name and picture, upon request.

Clients can request claims in two ways:

- An entire claims category by its scope value (see the above table for the scope value to claim mappings)
- 2. **Individually**, with the optional claims request parameter.

Claims and Scopes

Scope value	Associated claims
email	email, email_verified
phone	phone_number, phone_number_verified
profile	name, family_name, given_name, middle_name, nickname, preferred_username, profile, picture, website, gender, birthdate, zoneinfo, locale, updated_at
address	address

Summary

In order to use the OpenID connect authentication:

- register a Client (this can happen dynamically or statically) and obtain a client_id & client_secret
- issue an authentication request to the OP endpoint by redirecting the user browser
- the OP will authenticate user (via username/password or any other mechanism)
- the OP will then redirect the user browser to the Client redirection endpoint and extract the access token
- [request an ID token to the Token Endpoint of the OP using the access token]
- parse the response and extract the id token
- (optionally) use the access token to retrieve user information

Q C A

Thanks for your attention!