An Introduction to OpenID Connect

By Anoop Gupta (Capital One)

To comment on this article, please visit our GitHub repository and submit an issue.

Table of Contents

Abstract

The OpenID Connect (OIDC) specification builds on the OAuth 2.0 framework as a widely adopted identity layer. It provides a standardized and interoperable method for authenticating users and retrieving basic profile information via an ID Token. By leveraging REST-like protocols, OIDC ensures seamless integration across systems and platforms.

This article delves into the process of OpenID Providers (OP) connecting with federated Identity Providers (IDPs) to authenticate users, which facilitates the retrieval of user claims that can be returned to Relying Parties (RPs), enabling them to make informed decisions.

About OpenID Connect

OpenID Connect (OIDC) is a framework that facilitates the creation of a secure Internet identity ecosystem.[i] It provides seamless integration and robust support while prioritizing security and privacy configurations. Additionally, it promotes interoperability and boasts widespread client and device support.

Authorization Servers implementing the OpenID Connect (OIDC) protocol are called OpenID Providers (OP). A Relying Party (RP) connects to the OP to obtain an ID token as an assertion that the user has successfully authenticated and to receive a unique identifier representing the user. Additional purposes are ensuring the user is authorized for requested access and obtaining user attributes. The OP may authenticate users directly or delegate authentication to federated or non-federated Identity Providers (IDPs) to facilitate sign-in flows on web or mobile applications and share user profile information upon successful authentication. OpenID Connect does not define how OPs integrate with federated IDPs. Such integrations are left to the implementation details of the OP. The OP may act as a bridge in these cases, using proprietary or other non-standard protocols to authenticate users via the federated IDPs.

RPs use the authenticated user claims found in the issued tokens to comprehend the user's authentication process and the level of assurance, allowing RPs to make well-informed decisions.

For this article, the federated IDP flow is referenced. Note that OIDC does not mandate or standardize how federated IDPs are integrated. These integrations depend on the implementation details of the OP and may involve non-standard protocols or proprietary methods.

Background

The OAuth 2.0 specification defines a framework where clients or Relying Parties (RPs) connect with Resource Servers to access API resources on behalf of resource owners.[ii] The OP is pivotal in allowing this integration between the parties involved.

Bertrand Carlier's article "An Introduction to OAuth 2.0" in the IDPro Body of Knowledge provides a comprehensive overview of OAuth 2.0 and touches on OIDC. It is a valuable starting point for understanding these widely used protocols and covers fundamental terminology and references to IETF RFC details that describe the protocols.

OIDC emphasizes user authentication and introduces the concept of the ID Token. This token is exchanged between the OP and RP. It takes the form of a JSON Web Token (JWT) and communicates details about the authenticated user's identity through predefined standard claims and information related to the authentication process.[iii, iv, v]

The OIDC specification allows RPs to access more detailed user information through the UserInfo endpoint, a protected resource in the OAuth 2.0 framework. During the authentication process, RPs can request user claims from this endpoint using the Access Token obtained from the OP. The user claims are returned as a JSON object containing name-value pairs, providing comprehensive user information to the RP.

Terminology

Terminology is primarily derived from the Terminology in the IDPro Body of Knowledge.

Term	Definition	Source
Identity Provider (IdP)	An IdP is a service that stores and manages digital identities. Companies use these services to allow their employees or users to connect with the resources they need. They provide a way to manage access, adding or removing privileges while security remains tight.	Identity Providers (IdPs): The key to secure digital access.
OAuth 2.0	The OAuth 2.0 authorization framework enables a third-party application to obtain limited access to an HTTP service, either on behalf of a resource owner by orchestrating an approval interaction between the resource owner and the HTTP service or by allowing the third-party application to obtain access on its own behalf.	The OAuth 2.0 Authorization Framework
Authorization Server (AS)	The Authorization Server is able to authorize a client, issue tokens, and potentially validate tokens. It is also responsible for authenticating users, either directly or through federation.	An Introduction to OAuth2.0
OpenID Connect	OpenID Connect is a simple identity layer on top of the OAuth 2.0 protocol. It enables Clients to verify the identity of the End-User based on the authentication performed by an Authorization Server, as well as to obtain basic profile information about the End-User in an interoperable and REST-like manner.	Federation Simplified (v2)
OpenID Provider (OP)	An OpenID Provider (OP) is an entity that has implemented the OpenID Connect and OAuth	What is OpenID Connect

	2.0 protocols, OP's can sometimes be referred to by the role it plays, such as: an identity provider (IDP), or an authorization server, or a security token service.	
Resource Server	The server hosting the protected resources, capable of accepting and responding to protected resource requests using access tokens. Typically, exposed as an API.	The OAuth 2.0 Authorization Framework
Relying Party (RP)	A component, system, or application that uses the IDP to identify its users. The RP has its own resources and logic. Note that the term 'relying service' is used in the ISO/IEC standards to encompass all types of components that use identity services, including systems, sub-systems, and applications, independent of the domain or operator. We will use the more common Relying Party (or RP). An RP roughly corresponds to the Agency Endpoint in the FICAM model or to Identity Consumers in the Internet2 model.	IAM Reference Architecture
Client	An application making protected resource requests on behalf of the resource owner and with its authorization. The term "client" does not imply any particular implementation characteristics (e.g., whether the application executes on a server, a desktop, or other devices).	The OAuth 2.0 Authorization Framework

Tokens

The Authorization Code flow is the most commonly used method for RPs to obtain tokens in OIDC. This flow ensures secure communication between the RP and the Authorization Server through the following steps:

1. User Authentication: The RP redirects the user to the Authorization Server, where the user logs in.

2. Authorization Code Issuance: After successful authentication, the Authorization Server issues an Authorization Code to the RP.
3. Token Exchange: The RP exchanges the Authorization Code for tokens via a secure back-channel server-to-server call. These tokens typically include:
    - ID Token: Provides authenticated user identity claims.
    - Access Token: Grants access to protected resources.
    - Refresh Token (optional): Allows the RP to obtain new Access Tokens without requiring the user to log in again.

To enhance security, particularly for public clients, the Proof Key for Code Exchange (PKCE) extension is mandatory.[vi] While PKCE is mandatory for public clients, its use is also encouraged for confidential clients to provide additional protection against interception attacks. PKCE mitigates interception attacks by binding the authorization request to the token exchange process.[vii]

ID Token

The ID Token is a critical element of OpenID Connect, designed to securely communicate user authentication details. It is structured as a JWT and includes claims such as:

- sub: A unique identifier for the authenticated user.
- Optional Claims: Additional claims like name, profile, or email, which must be explicitly requested using scopes (e.g., openid, profile, email).

To ensure integrity, the Authorization Server digitally signs the ID Token using JSON Web Signature (JWS).[viii] For enhanced confidentiality, the ID Token can also be encrypted with JSON Web Encryption (JWE) using symmetric or asymmetric algorithms.[ix]

Both Access Tokens and ID Tokens have expiration times to ensure security. RPs should validate the expiration (exp) claim in tokens before processing them.

UserInfo Endpoint

The UserInfo endpoint is a key feature of OIDC, enabling RPs to retrieve additional user claims beyond those provided in the ID Token. Common claims retrieved via the UserInfo endpoint include name, email, birthdate, and address, depending on the scopes requested by the RP. These claims, formatted as a JSON object, offer more detailed information about the user's profile or authentication process.

RPs should follow the principle of data minimization by requesting only the claims necessary for their application to function.

Access and Authorization

The UserInfo endpoint requires an Access Token, issued by the Authorization Server during the authorization process, to authenticate and authorize requests. The claims available from the UserInfo endpoint depend on:

- The scopes requested during the authorization process (e.g., profile, email).
- The OpenID Provider's (OP) configuration and policies.

RPs may also explicitly request specific claims using the claims parameter in the authorization request. This allows for fine-grained control over the user information retrieved from the OP.

Optional Claims and Provider Discretion

The OP determines which claims are included in the ID Token and which are provided through the UserInfo endpoint. Not all claims in the ID Token are guaranteed to be available via the UserInfo endpoint. This separation ensures flexibility while adhering to privacy and security considerations.

Data Security

To ensure confidentiality, the response from the UserInfo endpoint can be encrypted using JWE. This encryption can be implemented using symmetric or asymmetric algorithms, depending on the RP's requirements, to safeguard sensitive user information during transmission.

Enhanced Decision-Making

The detailed claims retrieved from the UserInfo endpoint empower RPs to make informed decisions about user access, authentication levels, and personalized experiences within their applications.

By offering this additional layer of user information, the UserInfo endpoint complements the ID Token, allowing RPs to access a richer set of data while maintaining robust security and privacy standards.

OIDC Flow

OIDC provides a secure framework for authenticating users and exchanging identity information between RPs and OPs In some cases, the OP may delegate authentication to an external IDP. The typical sequence of the OIDC flow includes the following steps:

1. The Relying Party (RP) initiates the OpenID Connect (OIDC) flow with the Authorization Server to obtain user claims.

2. The Authorization Server initiates a new OIDC flow with the federated IDP, providing user authentication, and claims. The login screen is not rendered on the federated IdP if the user has already been authenticated.
3. After the user is successfully authenticated, the Identity Provider (IDP) issues an authorization code to the Authorization Server.
4. The Authorization Server exchanges the authorization code to retrieve the tokens.
5. The Authorization Server can optionally call the UserInfo endpoint of the federated IdP to retrieve extra user claims.
6. The Authorization Server returns an authorization code to the Relying Party.
7. The Relying Party (RP) exchanges the authorization code to retrieve the tokens from the Authorization Server.
8. The RP can optionally call the UserInfo endpoint on the Authorization Server to retrieve additional user claims.
9. The RP presents the access token to the Authorization Server.
10. The Authorization Server authenticates the RP using an access token to retrieve the protected resource.
11. The Authorization Server provides the protected resource to the RP.

Additional Notes

- Flexibility in Claims Sharing: The OP and, if applicable, the IDP may provide claims through their respective UserInfo endpoints. This flexibility allows RPs to access detailed user profiles tailored to their needs.
- Simplified Representation: This explanation omits intermediate browser redirects and technical details for brevity.
- Error Handling: If an error occurs at any step, such as invalid credentials or an expired authorization code, the OP should return an error response detailing the reason. RPs must handle these errors gracefully to provide feedback to the user.

By following this structured flow, OIDC ensures secure and reliable authentication while providing a flexible mechanism for retrieving identity and profile information. This enables RPs to create personalized and secure user experiences.

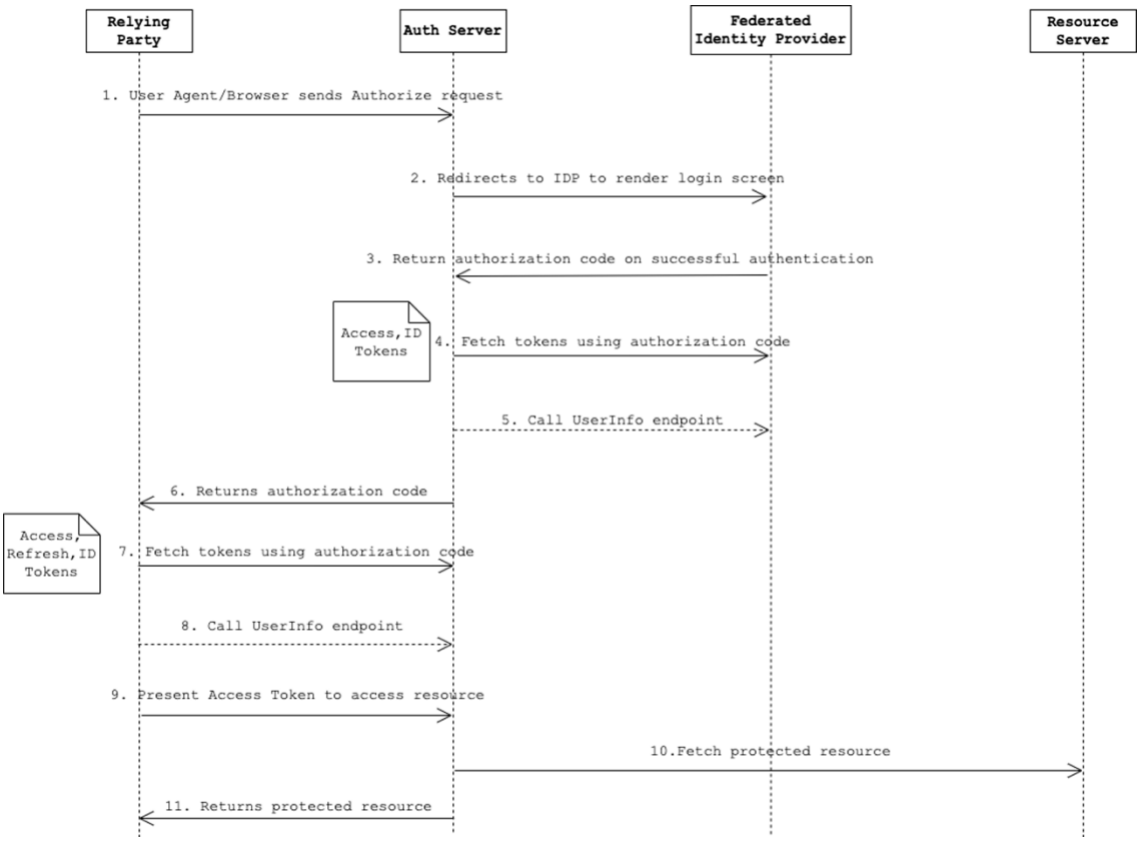

Figure 1: OAuth 2.0 and OpenID Connect flow

Logout

OIDC supports RP-initiated and back-channel logout mechanisms. However, the implementation of these protocols varies among OPs. RPs should consult the OP's metadata to determine the supported logout methods. The OP and IDP can provide logout endpoints for the RP to call during the user logout flow. Logout endpoints ensure no further calls can be made to the Resource Server using invalidated access tokens. In addition to invalidating tokens, RPs should ensure session cookies are cleared to fully terminate the user's session.

RP-initiated and back-channel logout are two mechanisms for ending a user session on the Identity Provider and invalidating the associated tokens.

- RP-Initiated Logout (https://openid.net/specs/openid-connect-rpinitiated-1_0.html)
- Back Channel Logout (https://openid.net/specs/openid-connect-backchannel-1_0.html)

Some OPs also support Single Logout (SLO), enabling simultaneous session termination across multiple RPs. RPs should review the OP's metadata to determine the supported logout methods and endpoints.

In general, implementations of logout mechanisms may vary significantly across OPs. RPs should test these flows thoroughly to ensure compatibility.

Discovery

The OpenID Connect (OIDC) discovery mechanism simplifies the integration of Relying Parties (RPs) with OpenID Providers (OPs) by providing a standardized method for retrieving configuration metadata. This metadata is accessible at a well-known URL: /.well-known/openid-configuration.

Metadata Overview

The discovery document includes essential details about the OP's capabilities, such as:

- **Endpoints:** URLs for token issuance, user authentication, and the UserInfo API.
- **Supported Grant Types:** Information on flows like authorization code and implicit.
- **Cryptographic Algorithms:** Supported methods for signing and encrypting tokens.
- **Scopes and Claims:** Available scopes and the claims that can be requested by RPs.

Dynamic Client Registration

The metadata also supports dynamic client registration, allowing RPs to programmatically register with the OP. This feature automates integration, reducing manual setup and ensuring consistency across different implementations. Dynamic client registration should be protected with appropriate measures, such as client authentication or pre-authorization by the OP, to prevent misuse.

Interoperability and Security

By centralizing these configuration details, the discovery mechanism:

- Streamlines the setup process for RPs.
- Enhances interoperability between systems.
- Improves security by ensuring that the RP operates with accurate, up-to-date information.

Best Practices

RPs should periodically validate the information retrieved from the discovery document to account for any updates or changes in the OP's configuration. This ensures seamless operation and adherence to the latest security standards.

The discovery mechanism is a cornerstone of OIDC's flexibility, enabling a wide range of applications to integrate quickly and securely with OpenID Providers.

Beyond The Introduction

In addition to what has been mentioned in this introduction, the OIDC specification covers a wide range of measures to ensure the secure transmission of user data. Several other topics are included that can enhance the authentication process.

- OAuth 2.0 Step Up Authentication Challenge Protocol (RFC 9470)
- Vector of Trust (RFC 8485)
- Client Dynamic Registration (*RFC 7591*)
- Authorization Server Metadata (*RFC 8414*)

Additional Reading

1. Lodderstedt, T., Ed., McGloin, M., and P. Hunt, "OAuth 2.0 Threat Model and Security Considerations", RFC 6819, DOI 10.17487/RFC6819, January 2013, <https://www.rfc-editor.org/info/rfc6819>.
2. M. Jones, "JSON Web Key", RFC 7517, DOI 10.17487/RFC7517, May 2015, <https://www.rfc-editor.org/info/rfc7517 >
3. Jones, M. and D. Hardt, "The OAuth 2.0 Authorization Framework: Bearer Token Usage", RFC 6750, DOI 10.17487/RFC6750, October 2012, <https://www.rfc-editor.org/info/rfc6750>.
4. Richer, J., Ed., "OAuth 2.0 Token Introspection", RFC 7662, DOI 10.17487/RFC7662, October 2015, <https://www.rfc-editor.org/info/rfc7662>.

---

[i] Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., Mortimore, C. “OpenID Connect Core 1.0 incorporating errata set 1,” OpenID Foundation, November 2014, https://openid.net/specs/openid-connect-core-1_0.html .

[ii] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <https://www.rfc-editor.org/info/rfc6749>.

[iii] Y.Sheffer, D.Hardt, M.Jones, "JSON Web Token Best Current Practices", RFC 8725, DOI 10.17487/RFC8725, February 2020, <https://www.rfc-editor.org/info/rfc8725>.

[iv] Bertocci, V., "JSON Web Token (JWT) Profile for OAuth 2.0 Access Tokens", RFC 9068, DOI 10.17487/RFC9068, October 2021, <https://www.rfc-editor.org/info/rfc9068>.
[v] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/info/rfc7519>.
[vi] Sakimura, N., Ed., Bradley, J., and N. Agarwal, "Proof Key for Code Exchange by OAuth Public Clients", RFC 7636, DOI 10.17487/RFC7636, September 2015, <https://www.rfc-editor.org/info/rfc7636>.
[vii] Jones, M., Nadalin, A., Campbell, B., Ed., Bradley, J., and C. Mortimore, "OAuth 2.0 Token Exchange", RFC 8693, DOI 10.17487/RFC8693, January 2020, <https://www.rfc-editor.org/info/rfc8693>.
[viii] M. Jones, J.Bradley, N.Sakimura M, "JSON Web Signature", RFC 7515, DOI 10.17487/RFC7515, May 2015, < https://www.rfc-editor.org/info/rfc7515 >.
[ix] M. Jones, James Hilderbrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <https://www.rfc-editor.org/info/rfc7516>.