Auth0 Docs

Login

Docs > Authentication and Authorization Flows

Authentication and Authorization Flows

In this article

Auth0 uses the OpenID Connect (OIDC) Protocol and OAuth 2.0 Authorization Framework to authenticate users and get their authorization to access protected resources. With Auth0, you can easily support different flows in your own applications and APIs without worrying about OIDC/OAuth 2.0 specifications or other technical aspects of authentication and authorization.

While often used interchangeably, authentication and authorization represent fundamentally different functions. In this article, we compare and contrast the two to show how they protect applications in complementary ways.

In simple terms, authentication is the process of verifying who a user is, while authorization is the process of verifying what they have access to.

Comparing these processes to a real-world example, when you go through security in an airport, you show your ID to authenticate your identity. Then, when you arrive at the gate, you present your boarding pass to the flight attendant, so they can authorize you to board your flight and allow access to the plane.

Here's a quick overview of the differences between authentication and authorization:

Authentication	Authorization

Authentication	Authorization
Determines whether users are who they claim to be	Determines what users can and cannot access
Challenges the user to validate credentials (for example, through passwords, answers to security questions, or facial recognition)	Verifies whether access is allowed through policies and rules
Usually done before authorization	Usually done after successful authentication
Generally, transmits info through an ID Token	Generally, transmits info through an Access Token
Generally governed by the OpenID Connect (OIDC) protocol	Generally governed by the OAuth 2.0 framework
Example: Employees in a company are required to authenticate through the network before accessing their company email	Example: After an employee successfully authenticates, the system determines what information the employees are allowed to access

In short, access to a resource is protected by both authentication and authorization. If you can't prove your identity, you won't be allowed into a resource. And even if you can prove your identity, if you are not authorized for that resource, you will still be denied access.

We support scenarios for server-side, mobile, desktop, client-side, machine-to-machine, and device applications.

If you're not sure which flow to use, we can help you decide. For a quick guide, see Which OAuth 2.0 Flow Should I Use?.

Authorization Code Flow

Because regular web apps are server-side apps where the source code is not publicly exposed, they can use the Authorization Code Flow, which exchanges an Authorization Code for a token.

- Authorization Code Flow

- Add Login Using the Authorization Code Flow
- Call API Using the Authorization Code Flow

Authorization Code Flow with Proof Key for Code Exchange (PKCE)

During authentication, mobile and native applications can use the Authorization Code Flow, but they require additional security. Additionally, single-page apps have special challenges. To mitigate these, OAuth 2.0 provides a version of the Authorization Code Flow which makes use of a Proof Key for Code Exchange (PKCE).

- Authorization Code Flow with Proof Key for Code Exchange (PKCE)
- Add Login Using the Authorization Code Flow with PKCE
- Call API Using the Authorization Code Flow with PKCE

Implicit Flow with Form Post

As an alternative to the Authorization Code Flow, OAuth 2.0 provides the Implicit Flow, which is intended for Public Clients, or applications which are unable to securely store Client Secrets. While this is no longer considered a best practice for requesting Access Tokens, when used with Form Post response mode, it does offer a streamlined workflow if the application needs only an ID token to perform user authentication.

- Implicit Flow with Form Post
- Add Login Using the Implicit Flow with Form Post
- Authenticate SPAs with Cookies

Hybrid Flow

Applications that are able to securely store Client Secrets may benefit from the use of the Hybrid Flow, which combines features of the Authorization Code Flow and Implicit Flow with Form Post to allow your application to have immediate access to an ID token while still providing for secure and safe retrieval of access and refresh tokens. This can be useful in situations where your application needs to immediately access information about the user, but must perform some processing before gaining access to protected resources for an extended period of time.

- Hybrid Flow
- Call API Using the Hybrid Flow

Client Credentials Flow

With machine-to-machine (M2M) applications, such as CLIs, daemons, or services running on your back-end, the system authenticates and authorizes the app rather than a user. For this scenario, typical authentication schemes like username + password or social logins don't make sense. Instead, M2M apps use the Client Credentials Flow (defined in OAuth 2.0 RFC 6749, section 4.4).

- Client Credentials Flow
- Call API Using the Client Credentials Flow

Device Authorization Flow

With input-constrained devices that connect to the internet, rather than authenticate the user directly, the device asks the user to go to a link on their computer or smartphone and authorize the device. This avoids a poor user experience for devices that do not have an easy way to enter text. To do this, device apps use the Device Authorization Flow (drafted in OAuth 2.0). For use with mobile/native applications.

- Device Authorization Flow
- Call API Using the Device Authorization Flow

Resource Owner Password Flow

Though we do not recommend it, highly-trusted applications can use the Resource Owner Password Flow, which requests that users provide credentials (username and password), typically using an interactive form. The Resource Owner Password Flow should only be used when redirect-based flows (like the Authorization Code Flow) cannot be used.

- Resource Owner Password Flow
- Call API Using the Resource Owner Password Flow

Lock

COMPANY

About Us

Blog

Jobs

Press

LEARN

Availability & Trust

Security

White Hat

API Explorer

MORE

Help & Support

Professional Services

Documentation

Open Source

WordPress

CONTACT

10800 NE 8th Street
Suite 600

+1 (888) 235-2699

+1 (425) 312-6521

Bellevue, WA 98004 +44 (0) 33-3234-1966