Name: Minh Hoang
Student number: 152103143

Understanding OAuth2 and its Grant Types in Web Service Security

Introduction:

OAuth2, an authorization framework, is widely utilized in web service security to enable secure access to resources without disclosing user credentials. It facilitates delegated access by allowing third-party applications to obtain limited access to an HTTP service on behalf of a user. This essay delves into the concept of OAuth2, its utility, and explores its various grant types.

What is OAuth2 and its Utility?

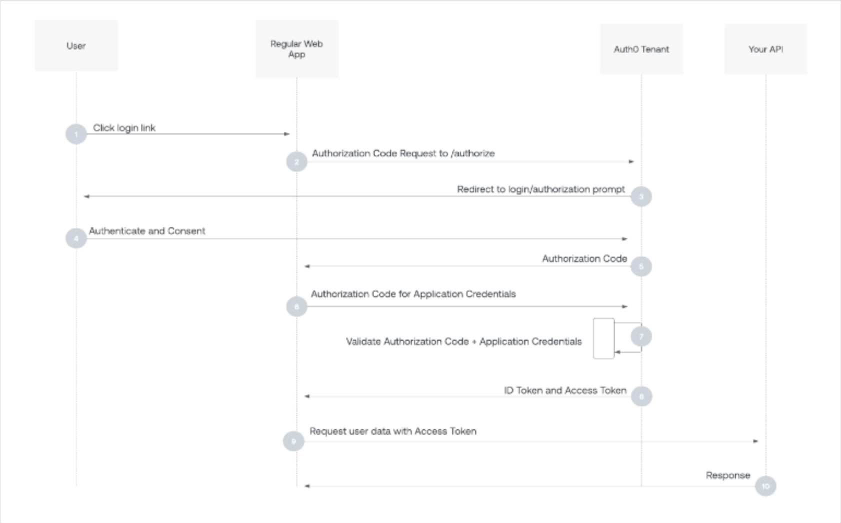
OAuth2, or Open Authorization 2.0, is a protocol designed to standardize the process of secure API authentication and authorization. It enables users to grant limited access to their resources without exposing their credentials. OAuth2 is commonly used in scenarios where applications need to access user data from other services, such as social media platforms, cloud storage, or APIs.

OAuth2 Grant Types:

OAuth2 defines several grant types to cater to different application architectures and security requirements. Let's explore some of the most common grant types:

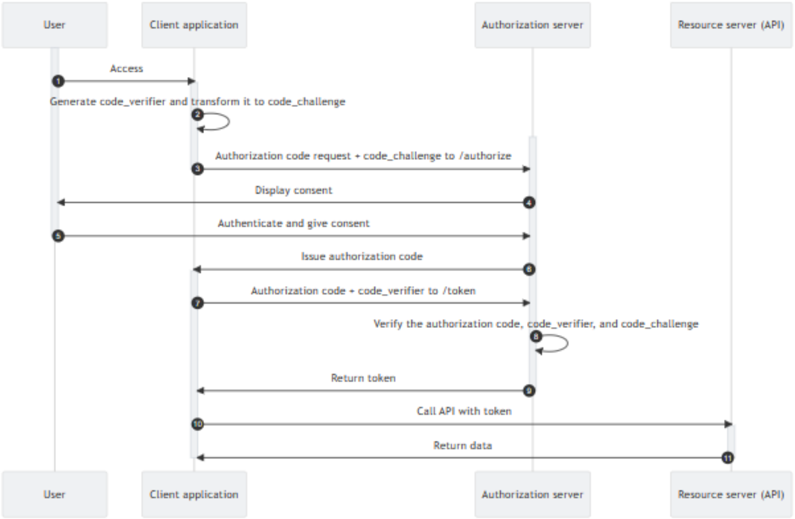
1. **Authorization Code Grant:**

- This is the most widely used OAuth2 grant type.
- It involves a two-step process: authorization request and access token request.
- After the user authenticates and authorizes the application, it receives an authorization code.
- The authorization code is exchanged for an access token, which is then used to access protected resources.


User
Regular Web App
Auth0 Tenant
Your API
1
Click login link
2
Authorization Code Request to /authorize
Redirect to login/authorization prompt
3
4
Authenticate and Consent
Authorization Code
5
6
Authorization Code for Application Credentials
Validate Authorization Code + Application Credentials
7
ID Token and Access Token
8
9
Request user data with Access Token
Response
10

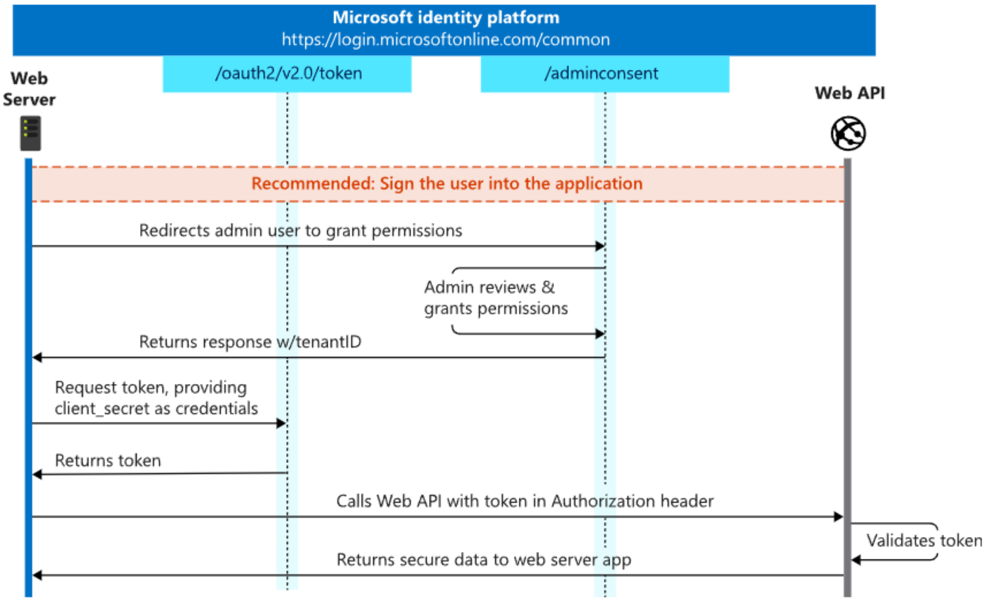
2. **PKCE (Proof Key for Code Exchange):**

- PKCE is an extension to the Authorization Code grant type designed to enhance security, particularly for mobile and native applications.
- It prevents interception of the authorization code by malicious actors by adding an additional verification step.
- PKCE mitigates the risk of authorization code interception and replay attacks.


User
Client application
Authorization server
Resource server (API)
Access
1
Generate code_verifier and transform it to code_challenge
2
Authorization code request + code_challenge to /authorize
3
Display consent
4
Authenticate and give consent
5
Issue authorization code
6
Authorization code + code_verifier to /token
7
Verify the authorization code, code_verifier, and code_challenge
8
Return token
9
Call API with token
10
Return data
11
User
Client application
Authorization server
Resource server (API)

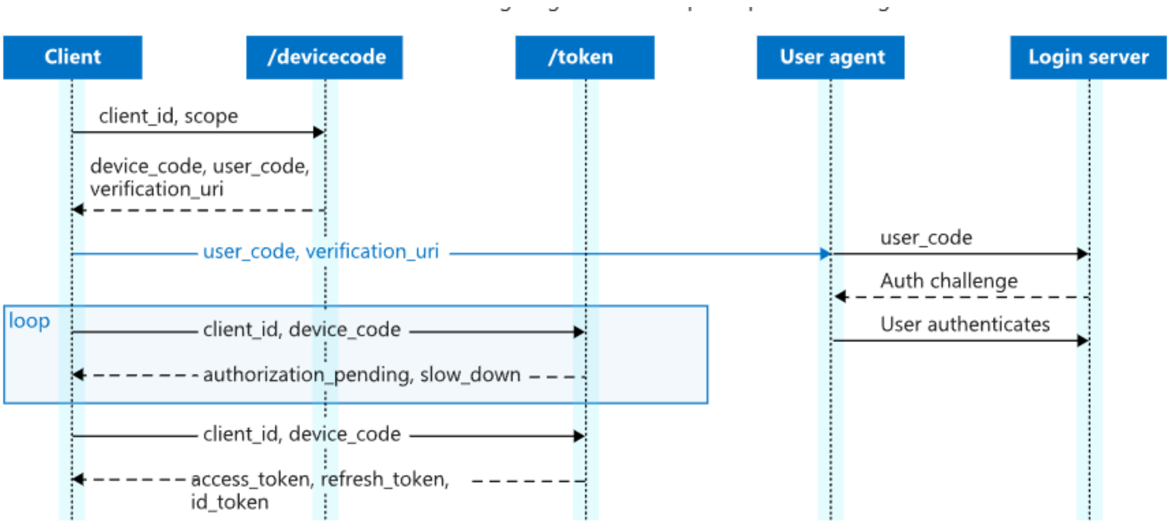
3. Client Credentials:

- In this grant type, the client directly exchanges its credentials (client ID and client secret) for an access token.
- It is suitable for machine-to-machine authentication or when the client is acting on its own behalf.
- Client Credentials grant type does not involve user authentication, making it appropriate for accessing resources that do not require user-specific permissions.


Microsoft identity platform
https://login.microsoftonline.com/common
/oauth2/v2.0/token
/adminconsent
Web Server
Web API
Recommended: Sign the user into the application
Redirects admin user to grant permissions
Admin reviews & grants permissions
Returns response w/tenantID
Request token, providing client_secret as credentials
Returns token
Calls Web API with token in Authorization header
Validates token
Returns secure data to web server app

4. Device Code:

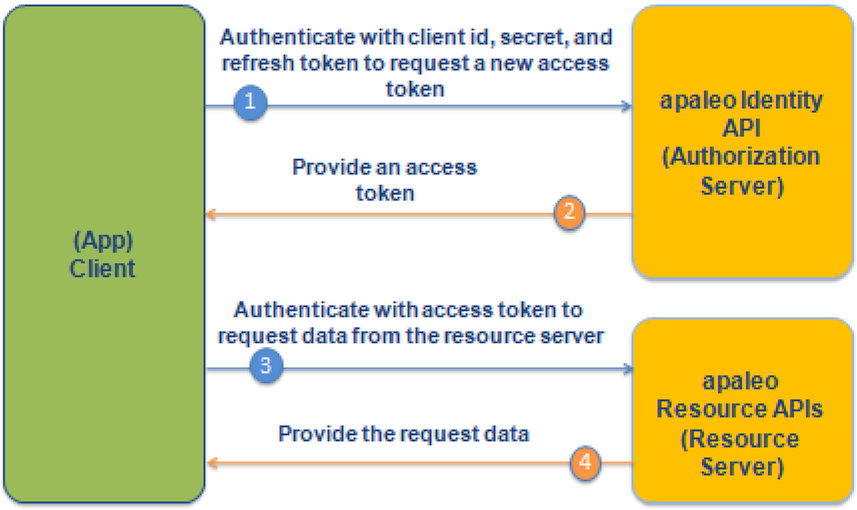
- This grant type is used for devices with limited input capabilities, such as smart TVs, gaming consoles, and IoT devices.
- It involves the device displaying a code to the user, who then enters it on a separate device with more extensive input capabilities (e.g., smartphone or computer) to complete the authorization process.



- Device Code grant type is suitable for scenarios where user interaction on the device itself is not feasible.

5. Refresh Token:

- While not strictly a grant type, refresh tokens are an essential aspect of OAuth2 for obtaining new access tokens without requiring the user to re-authenticate.
- After an access token expires, the client can use a refresh token to obtain a new access token without involving the user.
- Refresh tokens enhance user experience by allowing seamless access to resources without frequent re-authentication.


(App) Client
Authenticate with client id, secret, and refresh token to request a new access token
1
apaleo Identity API (Authorization Server)
Provide an access token
2
Authenticate with access token to request data from the resource server
3
apaleo Resource APIs (Resource Server)
Provide the request data
4