1. What is OAuth 2?

It is an open standard for token-based authentication and authorization and it allows an end user's account information to be used by third-party services, such as Facebook, without exposing the user's password

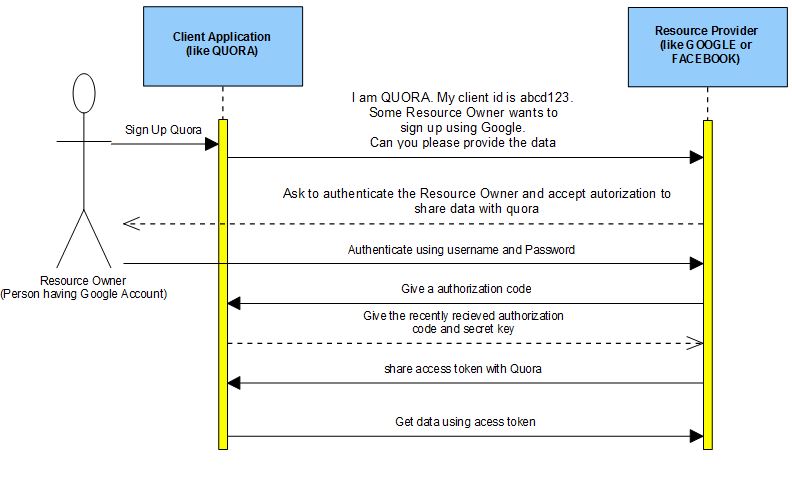
What is OAuth2 scope?

OAuth 2.0 scopes **provide a way to limit the amount of access that is granted to an access token**. For example, an access token issued to a client app may be granted READ and WRITE access to protected resources, or just READ access.

1. What is Grant Type in OAuth 2?

"grant type" refers to the way an application gets an access token.

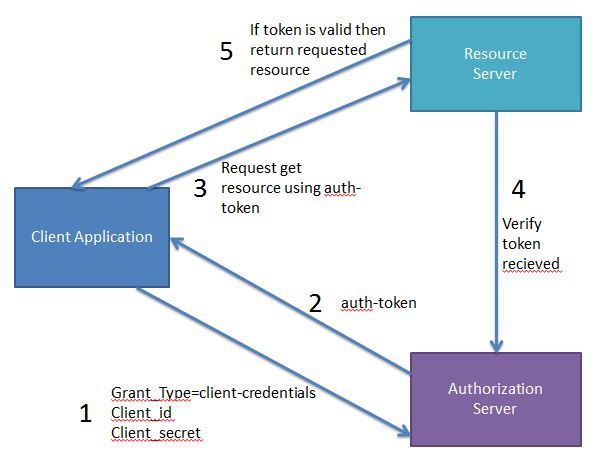
1. What are the different types of OAuth 2 Grants?
   1. **Authorization code grant**
   2. Implicit grant
   3. Resource owner credentials grant
   4. **Client credentials grant**
   5. **The Password Credentials Grant**
   6. **Refresh token grant**
2. Explain OAuth 2 - The Authorization Code Grant.



1. Explain OAuth 2 - The Client Credentials Grant.

The Client Credentials Grant involves machine to machine authentication. Oauth usually consists of following actors -

* **Resource Owner(User) -** An entity capable of granting access to a protected resource. When the resource owner is a person, it is referred to as an end-user.
* **Client Application -** The machine that needs to be authenticated.
* **Authorization Server -** The server issuing access tokens to the client after successfully authenticating the resource owner and obtaining authorization
* **Resource Server -** The resource server is the OAuth 2.0 term for your API server. The resource server handles authenticated requests after the application has obtained an access token.



1. Explain OAuth 2 - The Password Credentials Grant.

Oauth password grant usually consists of following actors -

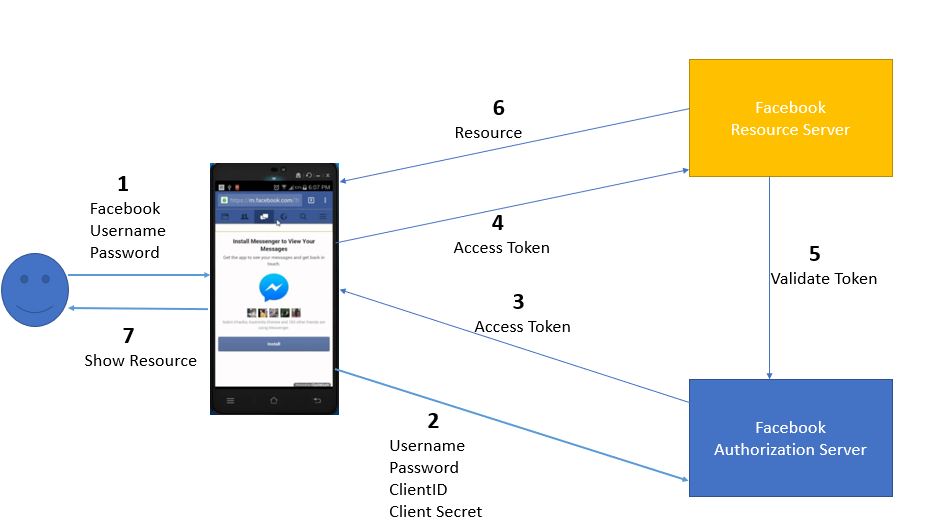
* **Resource Owner(User) -** An entity capable of granting access to a protected resource. When the resource owner is a person, it is referred to as an end-user.
* **Client Application -** The machine that needs to be authenticated.
* **Authorization Server -** The server issuing access tokens to the client after successfully authenticating the resource owner and obtaining authorization
* **Resource Server -** The resource server is the OAuth 2.0 term for your API server. The resource server handles authenticated requests after the application has obtained an access token.

**Password Credentials Grant**

In case of Password grant type the user triggers the client to get some resource. While doing so it passes the username and password to the client. The client then communicates with the authorization server using the provided username, password and also its own client ID and client Secret to get the access token. Using this access token it then gets the required resource from the resource server.

The real life example of Password grant will be you doing a login to you facebook account using its mobile application. Here the user will have to specify the facebook credentials to the app. Also the app will be having its own client id and client secret.

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| grant\_type (required) | client\_credentials |
| client\_id(required) | The client id |
| client\_secret(required) | The client secret key |
| username(required) | The username of the user |
| password(required) | The password of the user |



The Refresh Token grant type is **used by clients to exchange a refresh token for an access token when the access token has expired**.

JWT interview questions?

Anatomy of a JWT:

**Header**

The header typically consists of two parts: the type of the token, which is JWT, and the algorithm that is used, such as HMAC SHA256 or RSA SHA256. It is Base64Url encoded to form the first part of the JWT.

**Payload**

The payload contains the claims. There is a set of registered claims, for example: iss (issuer), exp (expiration time), sub (subject), and aud (audience). These claims are not mandatory but recommended to provide a set of useful, interoperable claims. The payload can also include extra attributes that define custom claims, such as employee role. Typically, the subject claim is used to create the OpenID Connect user subject. However, the Liberty JVM server can be configured to use an alternative claim. The payload is Base64Url encoded to form the second part of the JWT.

**Signature**

To create the signature part, the encoded header and encoded payload are signed by using the signature algorithm from the header. The signature is used to verify that the issuer of the JWT is who it says it is and to ensure that the message wasn't changed along the way.

{

"alg": "HS256",

"typ": "JWT"

}

{

"sub": "1234567890",

"name": "John Doe",

"iat": 1516239022

}

HMACSHA256(

base64UrlEncode(header) + "." +

base64UrlEncode(payload),

your-2ASas56-bit-secret

)

Basically, JWT is a token format. OAuth is an authorization protocol that can use JWT as a token.