**Why Oauth**

OAuth is an open-standard authorization protocol or framework that provides applications the ability for “secure designated access.” For example, you can tell Facebook that it’s OK for ESPN.com to access your profile or post updates to your timeline without having to give ESPN your Facebook password. This minimizes risk in a major way: In the event ESPN suffers a breach, your Facebook password remains safe.

OAuth doesn’t share password data but instead uses authorization tokens to prove an identity between consumers and service providers. OAuth is an authentication protocol that allows you to approve one application interacting with another on your behalf without giving away your password.

OAuth (Open Authorization) is an open standard for token-based authentication and authorization which is used to provide single sign-on (SSO). OAuth allows an end user's account information to be used by third-party services, such as Facebook, without exposing the user's password.

**Terminologies**

1. Authorization Server

Responsible for authenticating user’s identity and gives an authorization token. This token is accepted by resource server and validate your identity.

It’s used to authenticate your identity to provide access\_token, which you can use to request data from resource server.

1. Resource Server
2. 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.

Large scale deployments may have more than one resource server. Google’s services, for example, have dozens of resource servers, such as the Google Cloud platform, Google Maps, Google Drive, Youtube, Google+, and many others. Each of these resource servers are distinctly separate, but they all share the same authorization server.

Smaller deployments typically have only one resource server, and is often built as part of the same code base or same deployment as the authorization server.

1. The resource server will be getting requests from applications with an HTTP Authorization header containing an access token. The resource server needs to be able to verify the access token to determine whether to process the request, and find the associated user account, etc.
2. Tokens can be of different type based on where its stored or technique to decrypt it like self-encoded tokens, token store on db, Token Introspection.

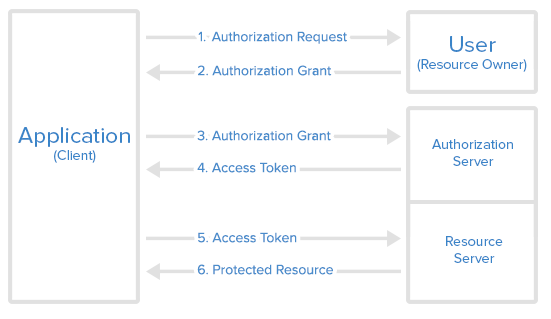
If you’re using self-encoded access tokens, then verifying the tokens can be done entirely in the resource server without interacting with a database or external servers.

1. Self-encoded tokens provide a way to avoid storing tokens in a database by encoding all of the necessary information in the token string itself. The main benefit of this is that API servers are able to verify access tokens without doing a database lookup on every API request, making the API much more easily scalable.
2. JWT

There are many ways to self-encode tokens. One way to create self-encoded tokens is to create a JSON-serialized representation of all the data you want to include in the token, and sign the resulting string with a key known only to your server.

A common technique for this is using the JSON Web Signature (JWS) standard to handle encoding, decoding and verification of tokens. The JSON Web Token (JWT) specification defines some terms you can use in the JWS, as well as defines some timestamp terms to determine whether a token is valid.

**Diagramatically**



**Example** -

Application (client) – Times of India -> To read full article TOI wants you to login using Google or FB account.

User (Resource Owner) – Owner of FB or Google account who has come on TOI site to read articles.

Authorization Server – Giving access token after authorization to use Google Account (Resource server).

Resource Server – Google account allowing application to get required info.

**Scenario** – Dheeraj who have Google account wants to read news on TOI. TOI allows users who have google account to read news on its site. Dheeraj doesn’t want to share it’s google password with TOI so here Oauth comes into picture.

API call goes to Google Authorization server and Dheeraj provides his credentials. Auth server generates token and client can use it to access resource server (google account) and get basic infos of user which TOI needs at its side.

**Sample code**

https://howtodoinjava.com/spring-boot2/oauth2-auth-server/

**Miscellaneous**

1. Where are PEM files stored on windows

Windows has a cryptographic key store, and it is simply located in a folder on your hard drive. On my Windows 10 machine, this path is C:\ProgramData\Microsoft\Crypto and inside that folder, there are various other folders for each key type

1. What are PEM files

Privacy Enhanced Mail (PEM) files are a type of Public Key Infrastructure (PKI) file used for keys and certificates. PEM, initially invented to make e-mail secure, is now an Internet security standard. HP Service Manager uses OpenSSL libraries to encrypt and decrypt SOAP messages over HTTP and requires certificates and keys in PEM format. The typical PEM files are:

key.pem contains the private encryption key

cert.pem contains certificate information

1. Base64encode in sample application

https://www.base64encode.org/

clientapp:123456

Y2xpZW50YXBwOjEyMzQ1Ng==

1. Redirect URI

A redirect URI, or reply URL, is the location where the authorization server sends the user once the app has been successfully authorized and granted an authorization code or access token.

1. Code

The Authorization Code grant type is used by confidential and public clients to exchange an authorization code for an access token.

After the user returns to the client via the redirect URL, the application will get the authorization code from the URL and use it to request an access token.

1. Bearer token

Bearer Tokens are the predominant type of access token used with OAuth 2.0. A Bearer Token is an opaque string, not intended to have any meaning to clients using it. Some servers will issue tokens that are a short string of hexadecimal characters, while others may use structured tokens such as JSON Web Tokens.