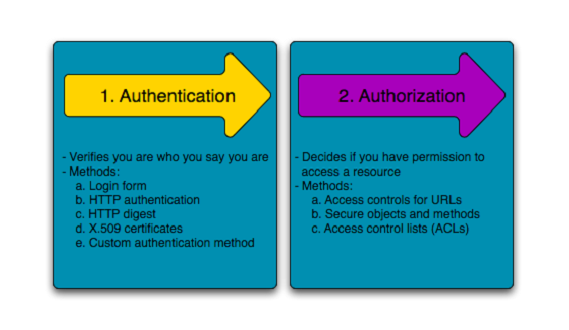
**OAUTH 2.0**

* What is the difference between Authentication and Authorization ?
  + Authentication means confirming your own identity, whereas authorization means being allowed access to the system.
  + In even more simpler terms authentication is the process of verifying oneself, while authorization is the process of verifying what you have access to.



**Authentication**

Authentication is about validating your credentials such as Username/User ID and password to verify your identity.

Authentication factors determine the many different elements the system uses to verify one’s identity before granting the individual access to anything. Based on the security level, authentication factors can vary from one of the following:

1. Single Factor Authentication
2. Two Factor Authentication
3. Multi Factor Authentication

**Single Factor Authentication**

**Two Factor Authentication**

**Multi Factor Authentication**

**Authorization**

Authorization occurs after your identity is successfully authenticated by the system, which therefore gives you full access to resources such as information, files, databases, funds, etc. However authorization verifies your rights to grant you access to resources only after determining your ability to access the system and up to what extent. In other words, authorization is the process to determine whether the authenticated user has access to the particular resources. A good example of this is, once verifying and confirming employee ID and passwords through authentication, the next step would be determining which employee has access to which floor and that is done through authorization.

Access to a system is protected by authentication and authorization, and they are frequently used in conjunction with each other. Although both have different concepts behind then, they are critical to the web service infrastructure, especially when it comes to being granted access to a system. Understanding each term is very important and a key aspect of security.

**Oauth 2.0 Authorization Framework**

[OAuth 2.0](https://oauth.net/2/) is a protocol that allows a user to grant limited access to their resources on one site, to another site, without having to expose their credentials.

Before we start with Oauth 2.0 there are some set of terminologies you must be familiar with namely

1. **Resource**

Namely mentioned as protected resource. And Oauth is an open standard to protect a resource known as protected resource.

1. **Resource Owner**

An Entity capable of authorizing access to a protected resource.

1. **Resource Server**

Server hosting the protected resource.

1. **Client**

An application making request for the protected resource on behalf of resource owner and with its authorization.

1. **Authorization Server**

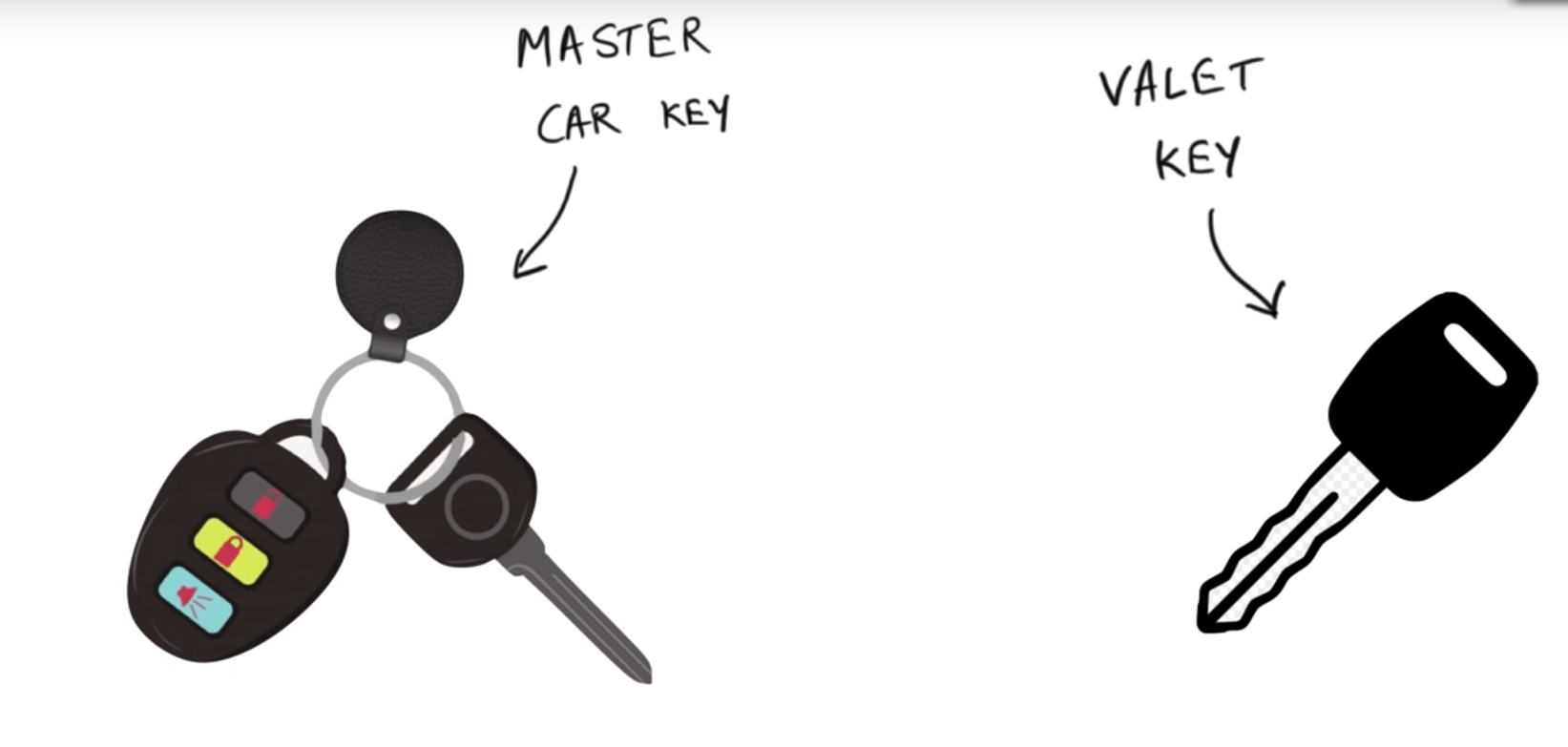
Burden of the resource holder security is fullfilled with the help of the authorization server. It issues the tokens to the client.

Models Examples:

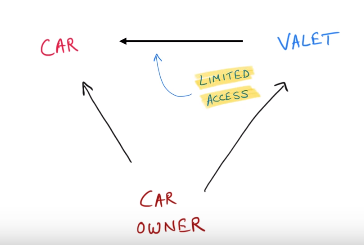
**Valet Key model**



Based on the valet key model. This rich guy asks the valet to park the car by handing him his car keys. But with the car keys he might have full access to all the lockers in the car. For this to restrict his usage many cars these days will come with the car keys.

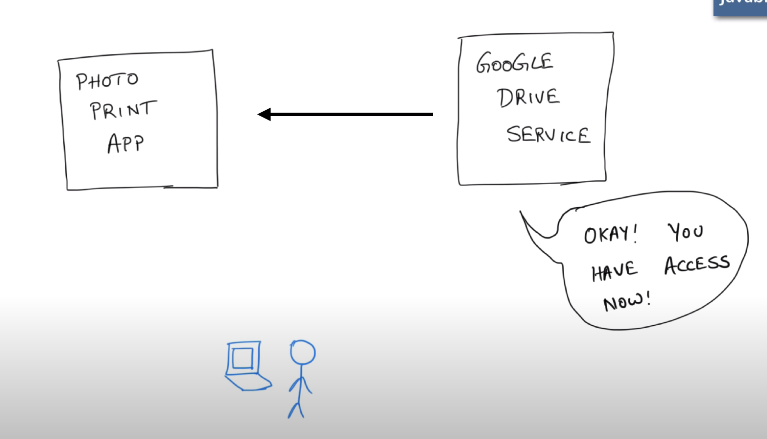


Many luxury cars today come with a valet key. It is a special key you give the parking attendant and unlike your regular key, will not allow the car to drive more than a mile or two. Some valet keys will not open the trunk, while others will block access to your onboard cell phone address book. Regardless of what restrictions the valet key imposes, the idea is very clever. You give someone limited access to your car with a special key, while using your regular key to unlock everything.



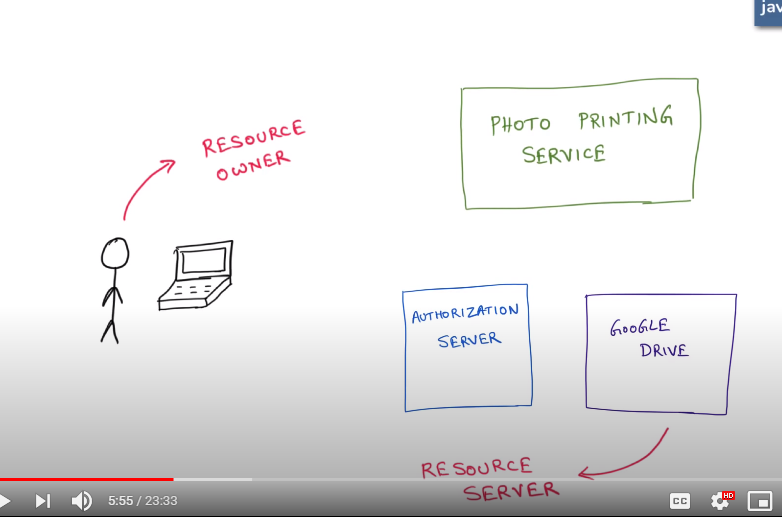
So Oauth is general is a authorization mechanisms , where services can authorize themselves on your behalf if given permission. This is basically called as the access delegation.

**Oauth Flow**



1. Suppose a user is authenticated to both the Photo Print App and the Google Drive.
2. Now Photo print app asks google drive for photos.
3. Google drive asks user should I give or not
4. If the user allows it gives back a token (valet) key.
5. Now this token can be used by the app to access photos.

**That is the reason why Oauth is authorization not authentication.**



Before we start with Oauth 2.0 there are some set of terminologies you must be familiar with namely

1. **Resource**

Namely mentioned as protected resource. And Oauth is an open standard to protect a resource known as protected resource. (Photos on google drive)

1. **Resource Owner**

An Entity capable of authorizing access to a protected resource. (like the user)

1. **Resource Server**

Server hosting the protected resource. (Google Drive)

1. **Client**

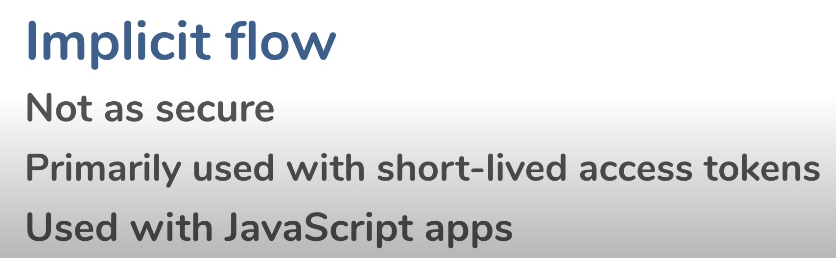
An application making request for the protected resource on behalf of resource owner and with its authorization. (Photo Printing App)

1. **Authorization Server**

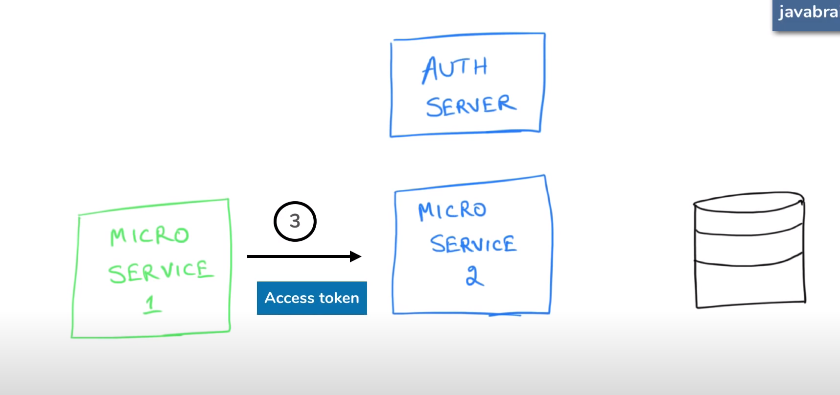
Burden of the resource holder security is fullfilled with the help of the authorization server. It issues the tokens to the client.

There are multiple variations of the Oauth flow to work.

* + - * **Authorization code flow --**
      * **Implicit Flow – instead send the access token directly. But drawback is there.**



* + - * **Oauth Authorization between microservices using Client Credential Flow.**



**OAUTH 2.0 Explanatory**

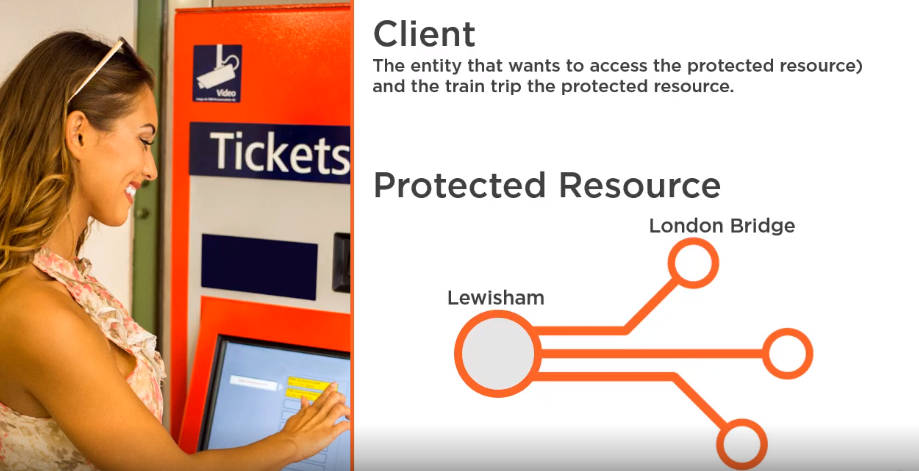
**Oauth 2**

An open standard used to protect a resource known as protected resouce.

**Train travel model**

1. Suppose a person wants to travel to from Bangalore to Hyderabad via train.
2. So this train is considered as a **protected resource**.
3. In order to access this train service the person have to go through the gates of the train station. So this train station is referred to as the **Resource server**.
4. In order to get through the gates of the train the person needs a ticket. That ticket is called as the **access token**.
5. In order to get the ticket(access token) person has to get the tickets from the ticket vending machine in this case it is called as the **Authorization server.**
6. Now in order of ticket vending machine to vend the ticket it has to asks the train service as can I vend tickets if the seats are available. In this case train service is called as the **Resource Owner.**

Actors for the Oauth



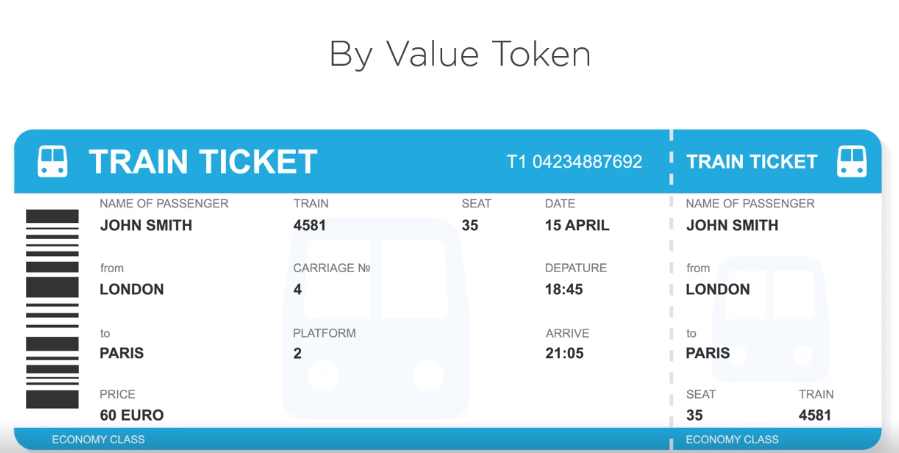
Passeger wants to travel from Lewisham station to the London Bridge Station , here passenger is known as the **client**(Entity) that wants to access the protected resource.

Here train trip is the **protected resource**.



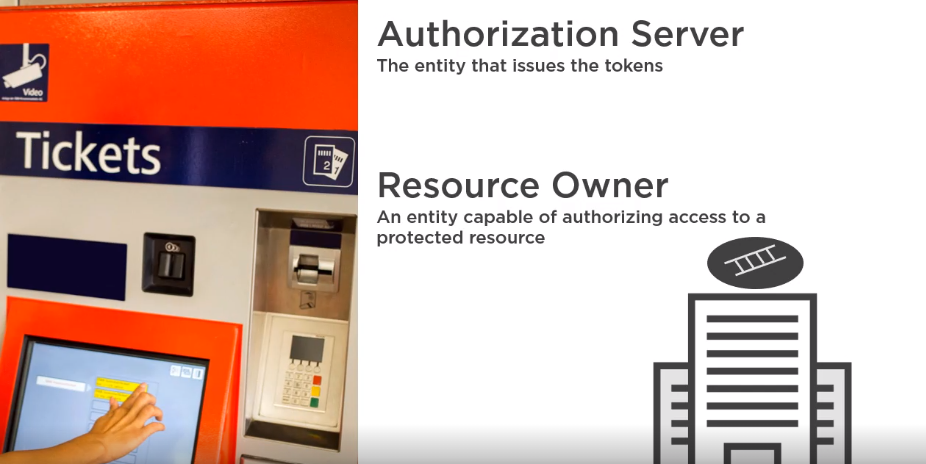
Train station is referred to as the **Resource server** , entity capable of authorizing access to a protected resource.

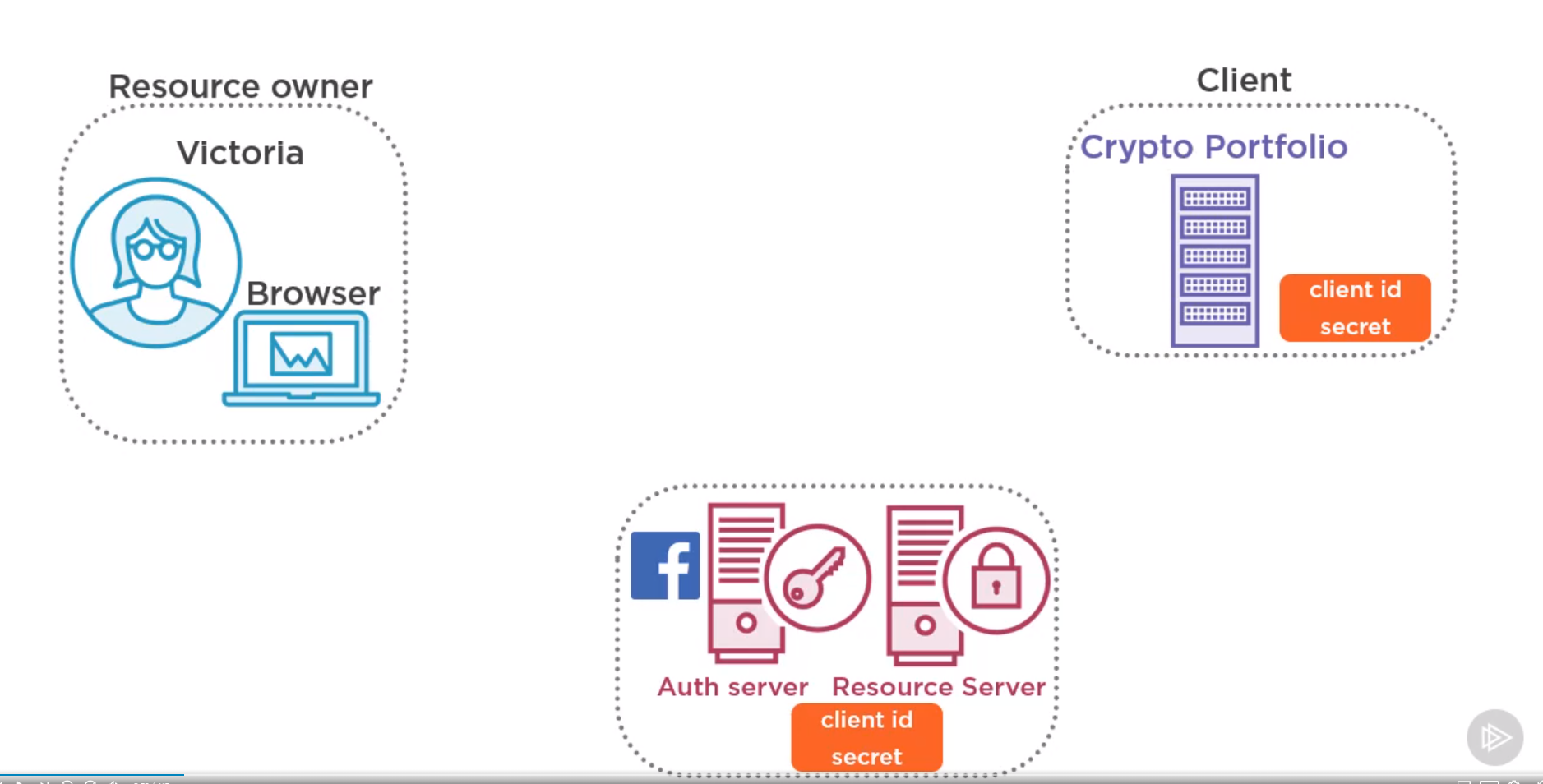
In order to get through the gates passenger needs a ticket and in oauth it is known as value token (by value or by reference)

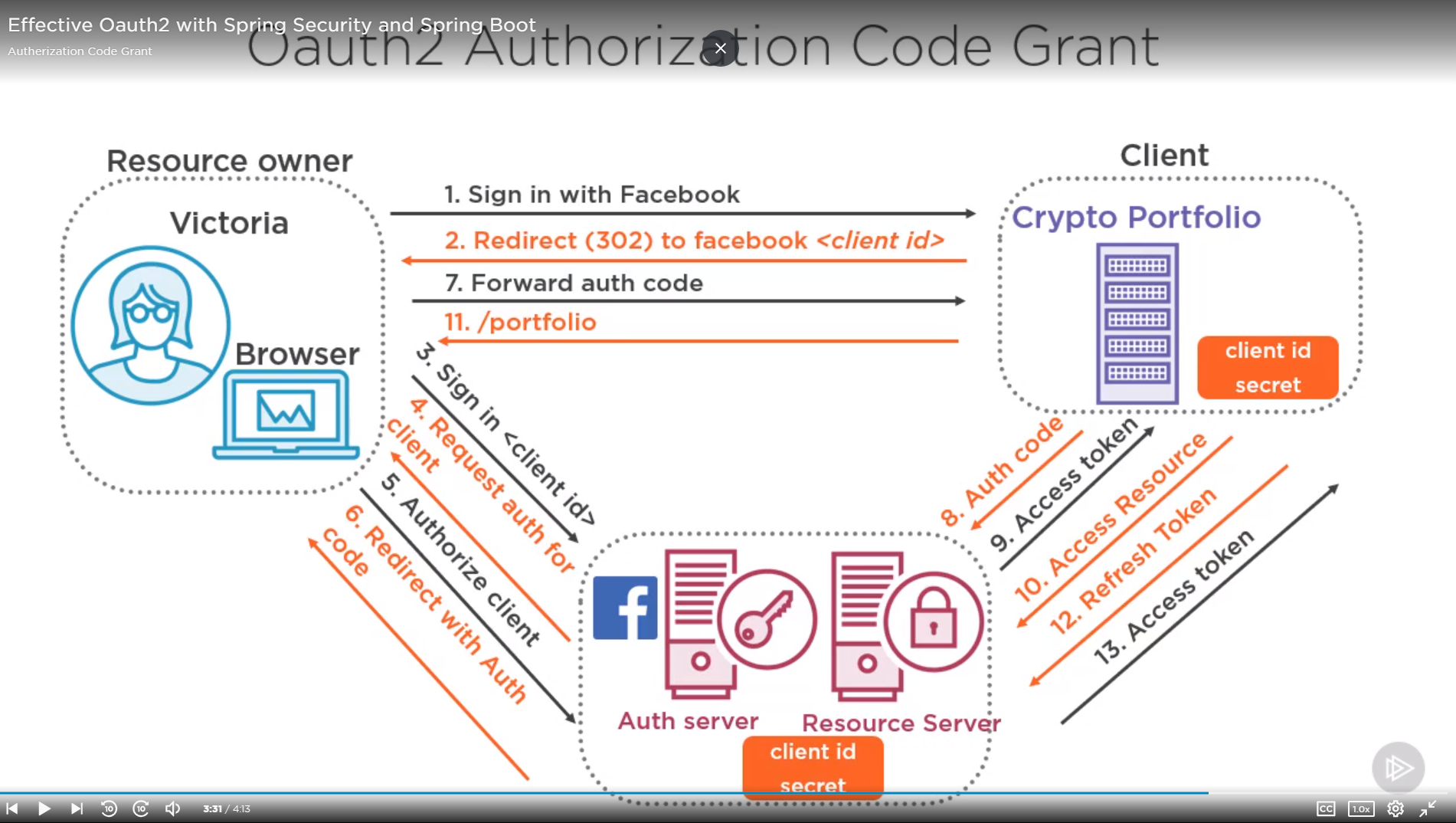


A user can get a valid ticket from the ticket vending machine , in oauth it is called as the **authorization server**.

Before giving a ticket it needs to check with the train company about the price , which is known as **resource owner**.







Victoria wanted to encrypt her holdings with Crypto Portfolio. Facebook and Crypto Portfolio have a trust relationship , i.e Crypto portfolio is the registered client with the facebook and has a unique client id and the client key. Hence victoria can authorize crypto portfolio to perform actions on her facebook account on her behalf , such as access her profile data , post to her wall or access her friends list.

Crypto portfolio will show the option like login with facebook. If victoria click on it , crypto portfolio will send a redirect request to facebook authorization server via her browser.

This begins the Oauth process with facebook, first facebook needs to authenticate victoria , if she is not already logged in with facebook she will need to do so. Once authenticated facebook authentication server , using the client id and the scope , sees that the client cyrpto portfolio wants to access her profile and email.

At this point it will ask victoria for her consent. Facebook also gives option to edit and remove some scopes. If victoria agrees , facebook will then send a redirect request reponse to the client via her browser, with the Uri specified by crypto portfolio and an authorizartion code.

This authorization code can now be used by the client to exchange the access token.

Authorization code is a random number and effectively a sender constraint token, meaning only the client can use it to get the access token.

Resource server expects the

Authorization token + Client id + Secret for the access token.

Acccess token also have a very short expiry time. The reason for using a authorization code and not passing the access token directly is that transmission via the front channel between the authorization server and the browser and client cannot be completely relied upon as secure , especially the browser. It could have some malicious plugin so sending the bearer access token would be risky.

And Authorization code on the other hand is useless without the client secret and that exchange happens via back channel where client goest directly to the authorization server . There is one other token that can be returned with the access token and that is refresh token. Because access token are Bearer token , they should have a very short expiry time. It is to limit damage and token being leaked. So these refresh token can be used by the client to get the new access token.

Refresh tokens need to be stored securely in the client. Best practice is every time a new access token is issue along with that a new refresh token should be issued.

**Most important , how people are using Oauth in authentication , like login with google or facebook like that.**

Oauth Based Authentication – This is something that people have developed as a way to authenticate. Oauth originally wasn’t meant for authentication.

But it is not always a good idea.



Someone who is building that client who is trying to access that resource server.

Setup authentication by yourself. They may have a google account or facebook account.

Once the authentication is successful on the client now it can access the resource server.

I am going to access may be google may be facebook , and I can get information about this user from the google account or the facebook account.

If that is the case why do I have to implement authentication all over again. This user is already authenticated with google and google has a reliable authentication mechanisms.

We can rely on google authentication mechanisms. So we can easily delegate authentication to that resource server in case google.

Now I can create an account on my end or rely on google or facebook account.



So when we create a spring security application , make an oauth call to the google and get’s the user’s profile information. Now your spring security application is going to put the output in to the SecurityContext to simulate the logged-in user.

Oauth single sign on functionality using the annotations.