OAuth and Open ID connect:

* Single sign on uses SAML protocol. (Sign across sites). This was used 10 years back and now also use this.
* Terminologies of OAuth:

1 Resource owner

2 Client

3 Authorization Server

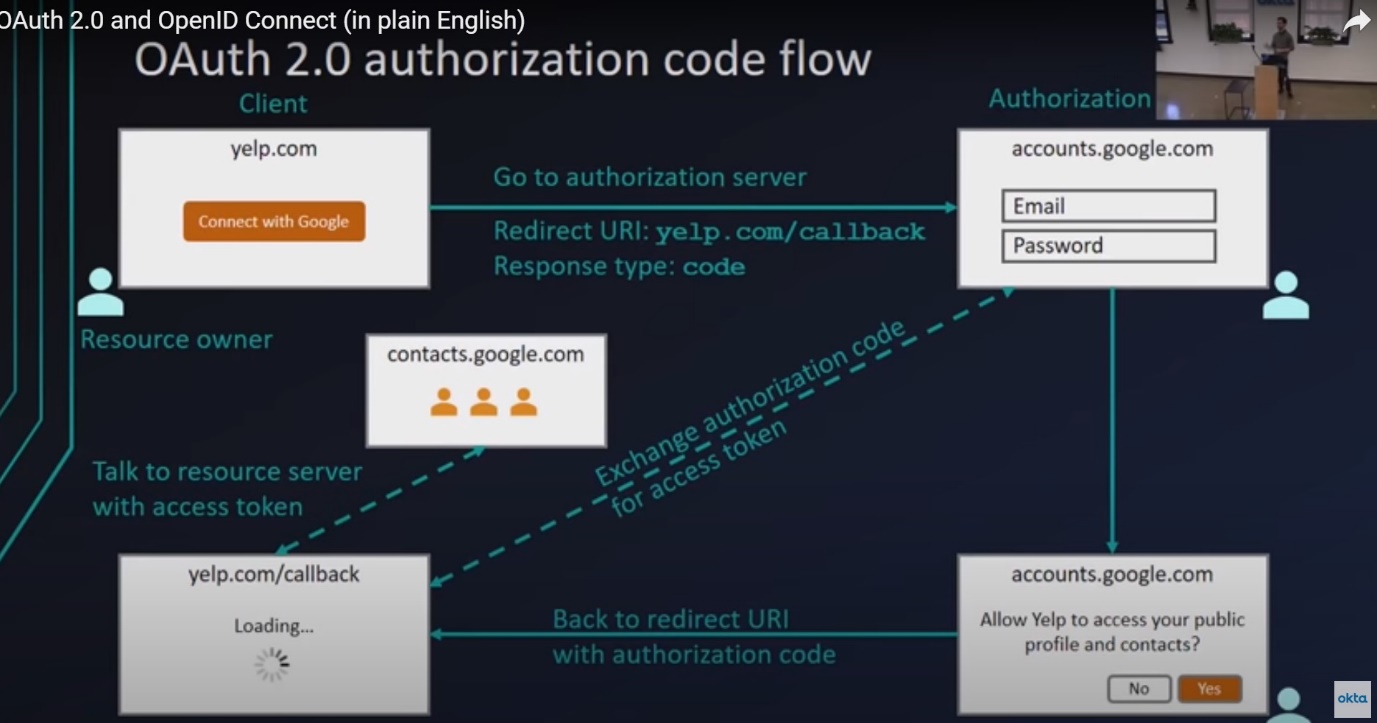
4 Resource Server

5 Authorization grant

6 Redirect URI

7 Aceess Token

Flow will look as below:



1. Why exchange authorization code for Access token? (For security purpose).

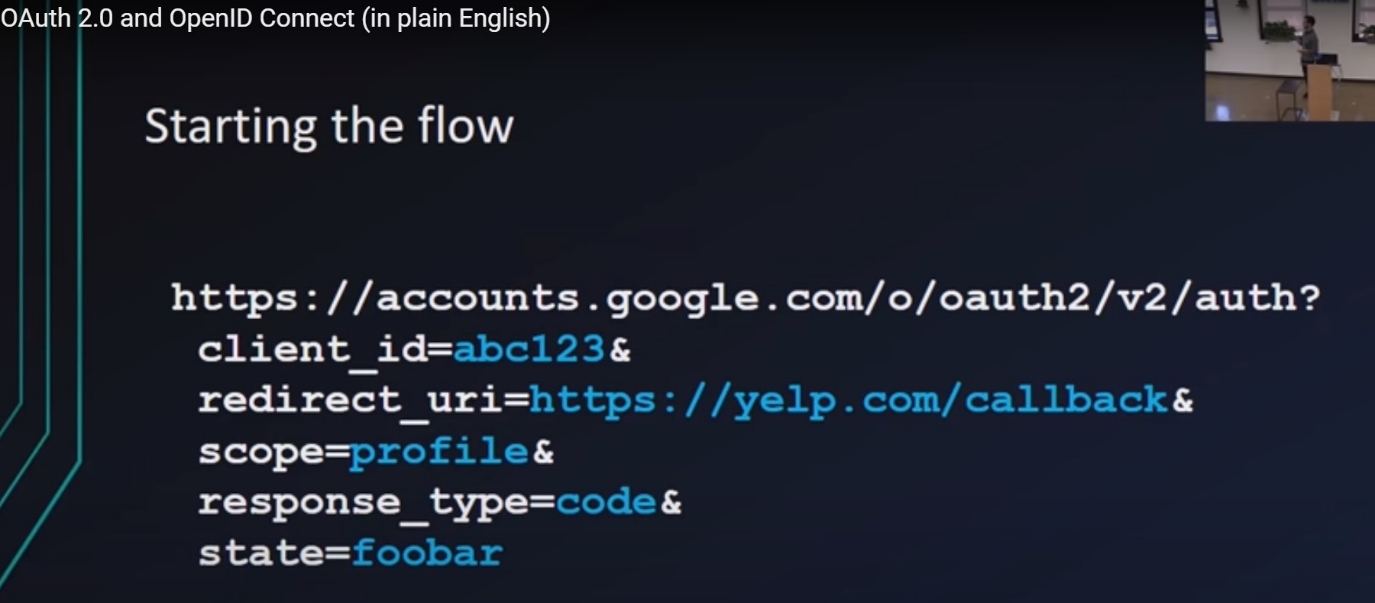
Best things of front channel and back channel are usd.

We need to know back channel and front channel for this.

Hard lines are Front Channel.

Dotted lines are in Back Channel.

How URL looks like as below:



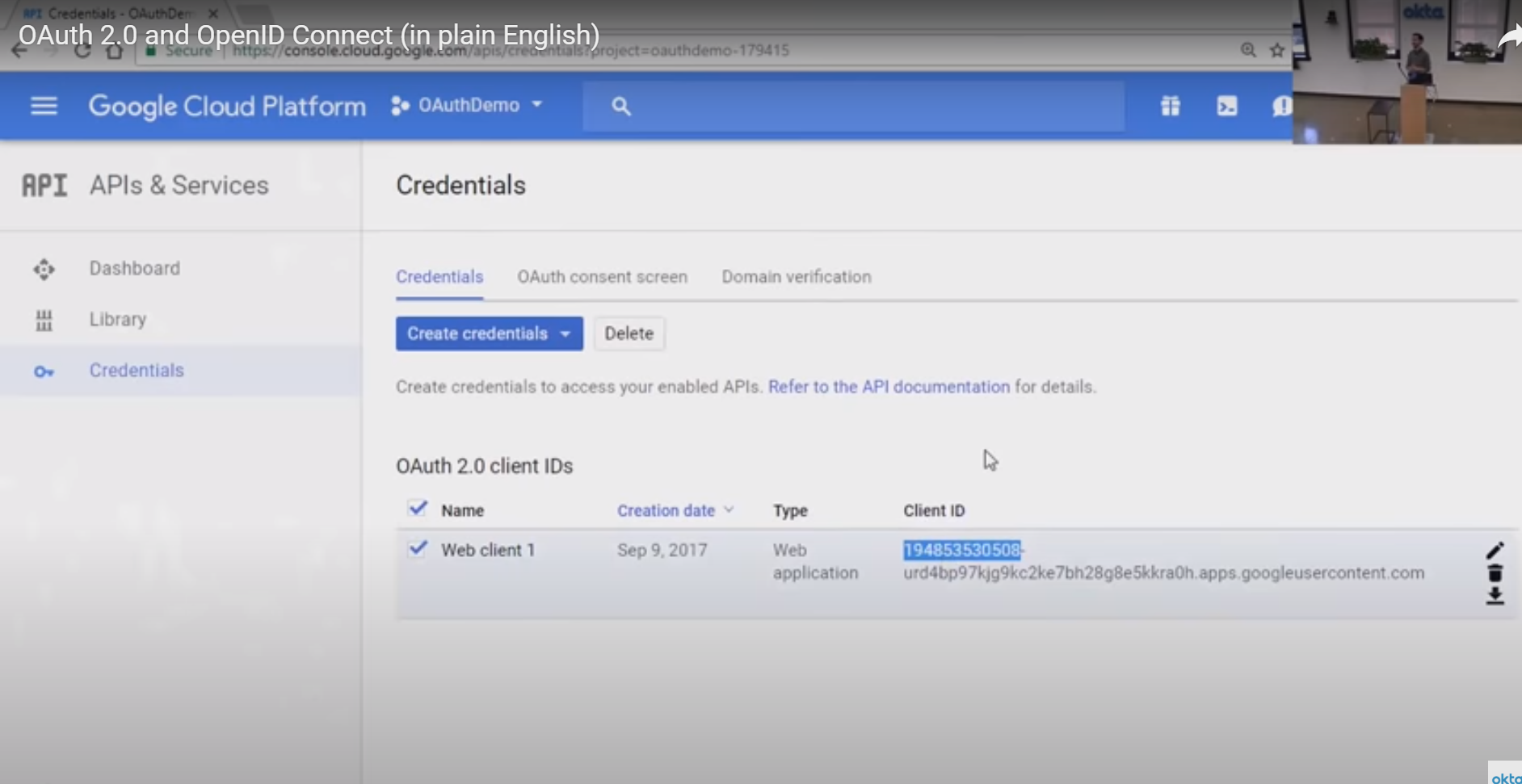
1. Does that mean, Client have already contacted Google at first to get the key?

Yes, One time set to be made with Google (create client) to get the keys.

2 Keys:

Client ID = passed through the UI with initial request.

Client Secret = back channel through token exchange request.



<https://oauthdebugger.com/> url can be used for debugging.

Request against google.

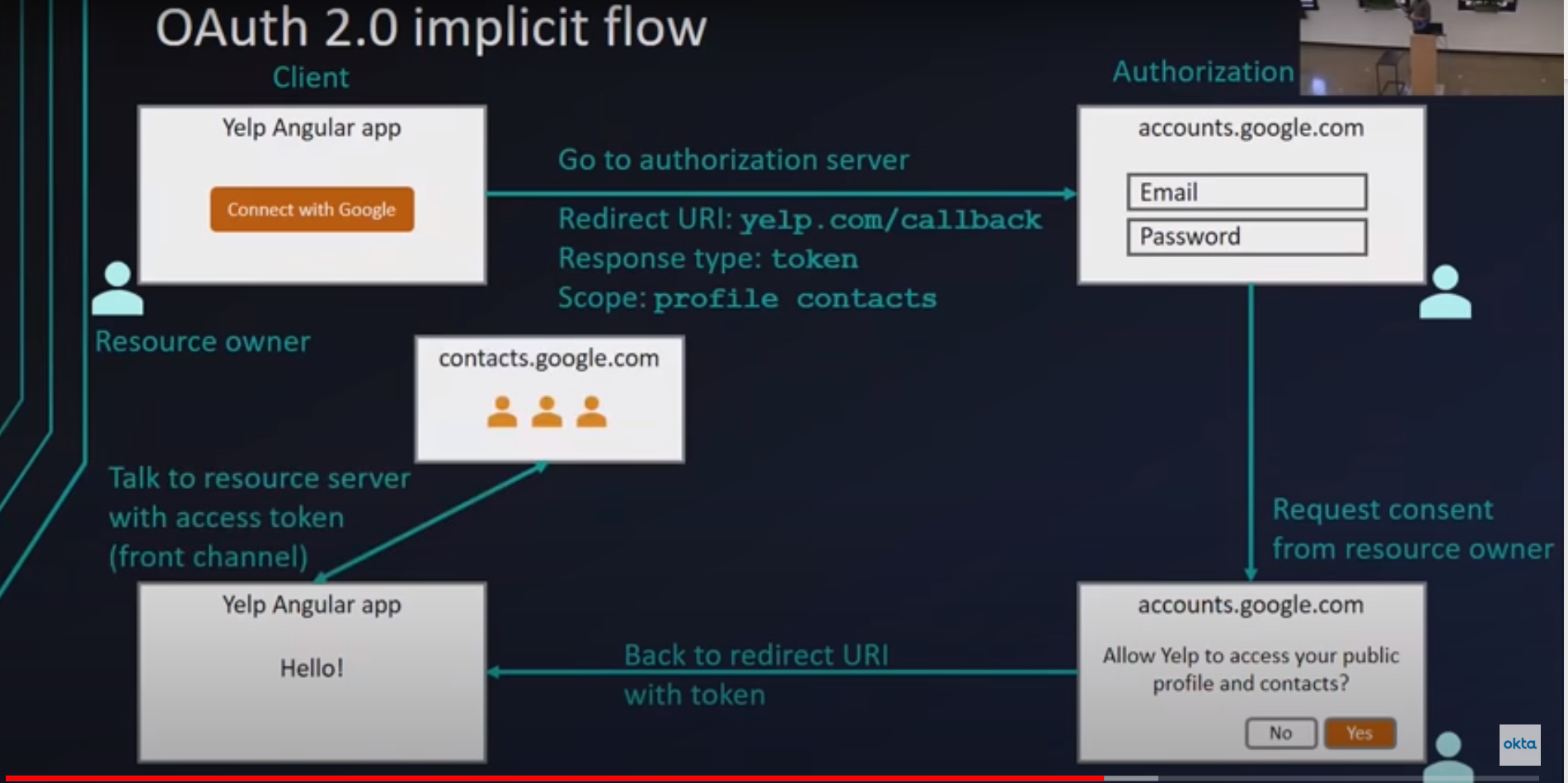
OAuth 2.0 flows can be done 4 different ways:

1. Authorization code (Front+Back) -one written above
2. Implicit (Front channel only)
3. Resource owner passed credentials (back channel only)

Used for older applications.

1. Client credentials(back channel only)

Implicit flow: Security problem

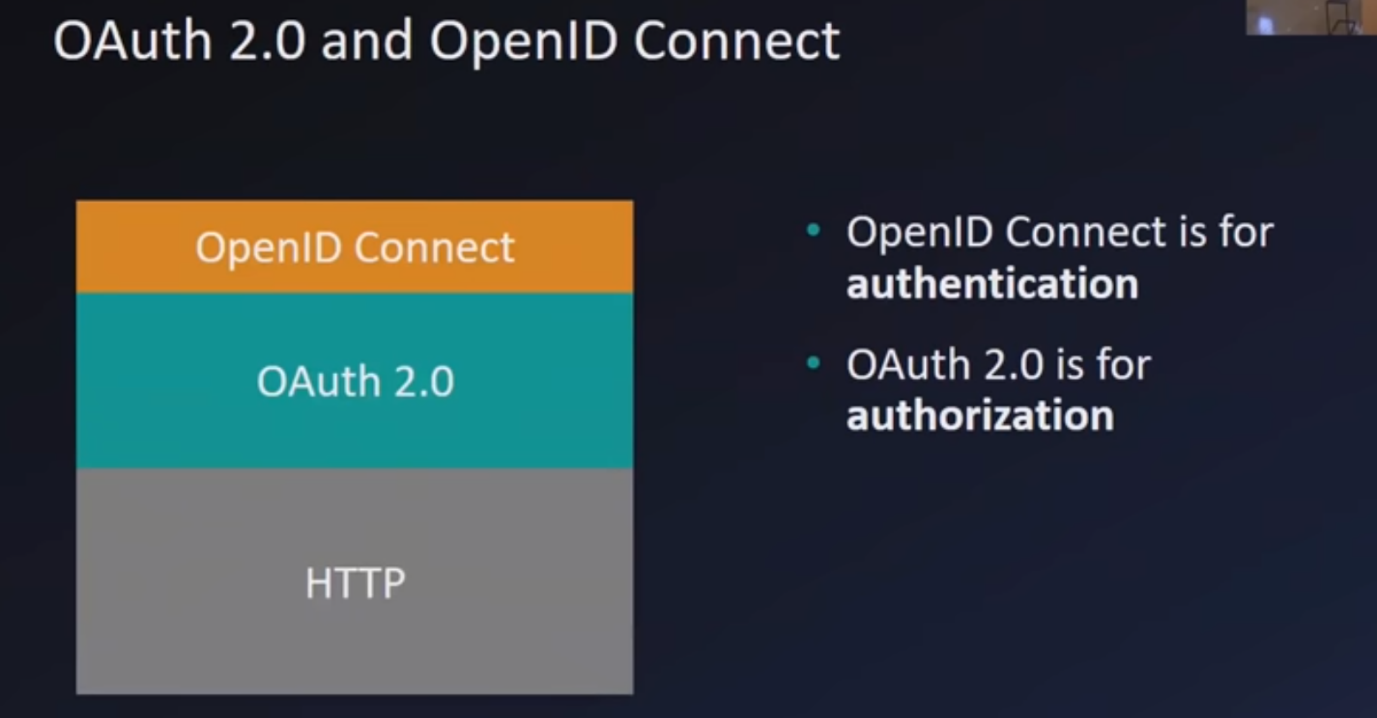


Oauth was build initially for Authorization but started use for Authentication.

Problem with Authentication :

1. No standard way to get the users info
2. Every implementation is a little different
3. No common set of scopes .

To solve authentication problem, a layer was build on top of OpenID connect.



OpenID is an extension of OAuth 2.0,

<https://www.jsonwebtoken.io/>

The ID token (JWT)

It has Header portion, a Signature portion.

Use OAuth for: (Authorization)

* Granting access to your API
* Getting access to user data in other systems

Use OpendID connect:(Authentication)

- Logging the user in

- Making your accounts available in other systems.