Gladwin Burboz [Speaker]

OAUTH/OIDC
UNDER THE HOOD OF SOCIAL LOGIN

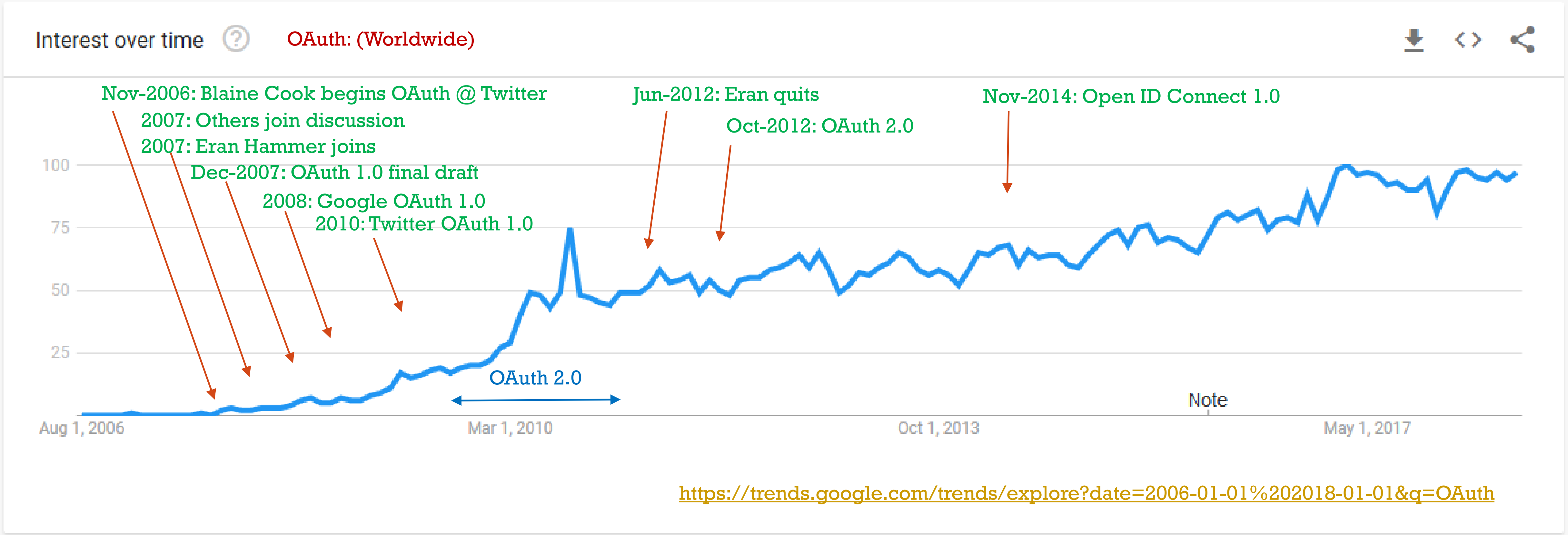
Monday, August 20, 2018 - 6:30 PM to 8:30 PM

DISCLAIMER: NON AFFILIATION & LIABILITY WAIVER

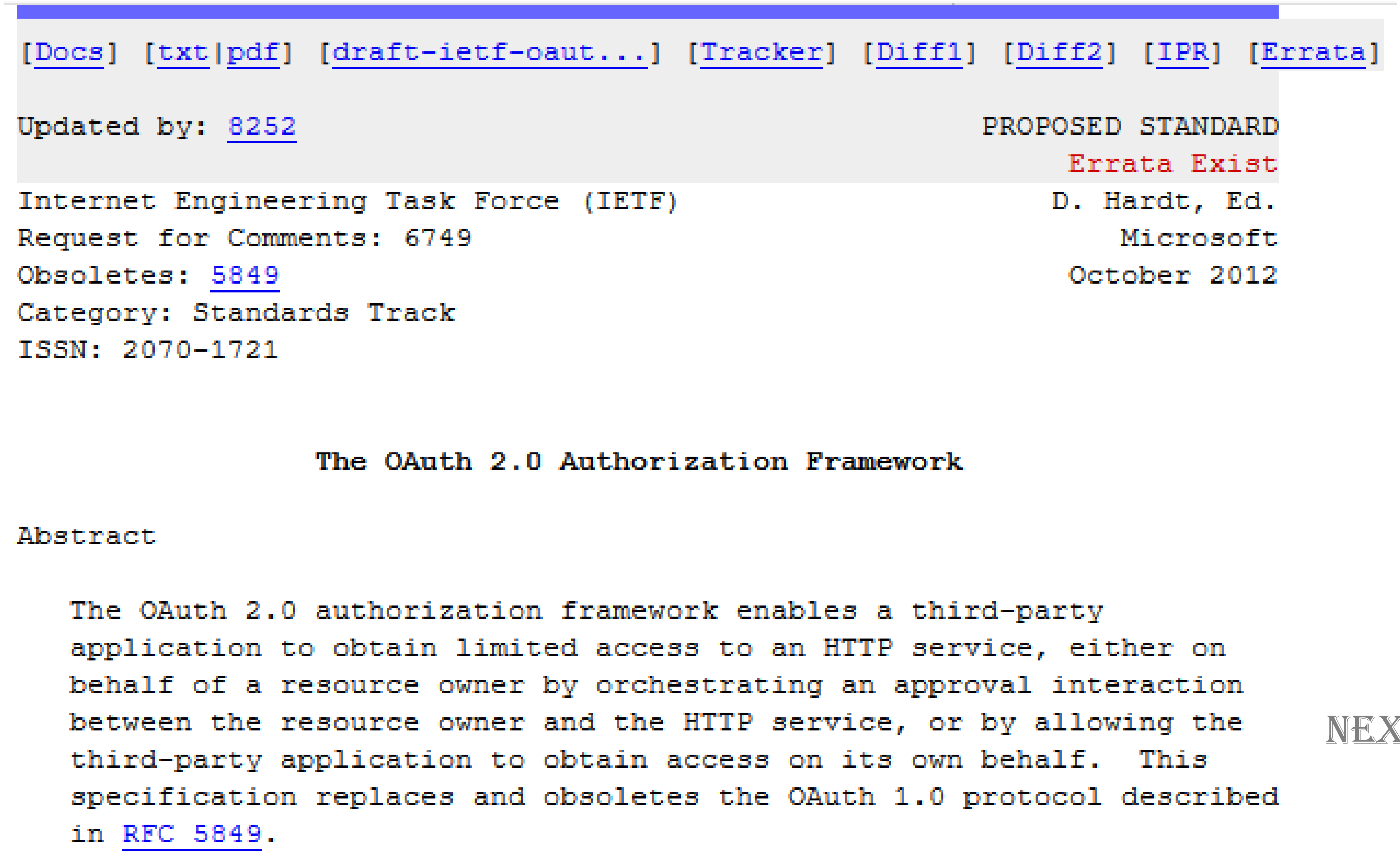
- My current or past employers has no connection or liability towards this presentation
- I am not affiliated with any of the organizations or products discussed in this presentation
- This presentation is for educational purpose only with no guarantee of accuracies with specifications
- Screenshots have been modified to present concepts

OAUTH 2.0 & OPENID CONNECT 1.0 SPEC

- The OAuth 2.0 [rfc6749]
  - https://tools.ietf.org/html/rfc6749
- Open ID Connect 1.0
  - http://openid.net/connect/



https://trends.google.com/trends/explore?date=2006-01-01%202018-01-01&q=OAuth



[Docs] [txt|pdf] [draft-ietf-oaut...] [Tracker] [Diff1] [Diff2] [IPR] [Errata]

```
Updated by: 8252                                       PROPOSED STANDARD
                                                            Errata Exist
Internet Engineering Task Force (IETF)                     D. Hardt, Ed.
Request for Comments: 6749                                     Microsoft
Obsoletes: 5849                                             October 2012
Category: Standards Track
ISSN: 2070-1721


                 The OAuth 2.0 Authorization Framework

Abstract

   The OAuth 2.0 authorization framework enables a third-party
   application to obtain limited access to an HTTP service, either on
   behalf of a resource owner by orchestrating an approval interaction
   between the resource owner and the HTTP service, or by allowing the
   third-party application to obtain access on its own behalf.  This
   specification replaces and obsoletes the OAuth 1.0 protocol described
   in RFC 5849.
```

NEXT >>

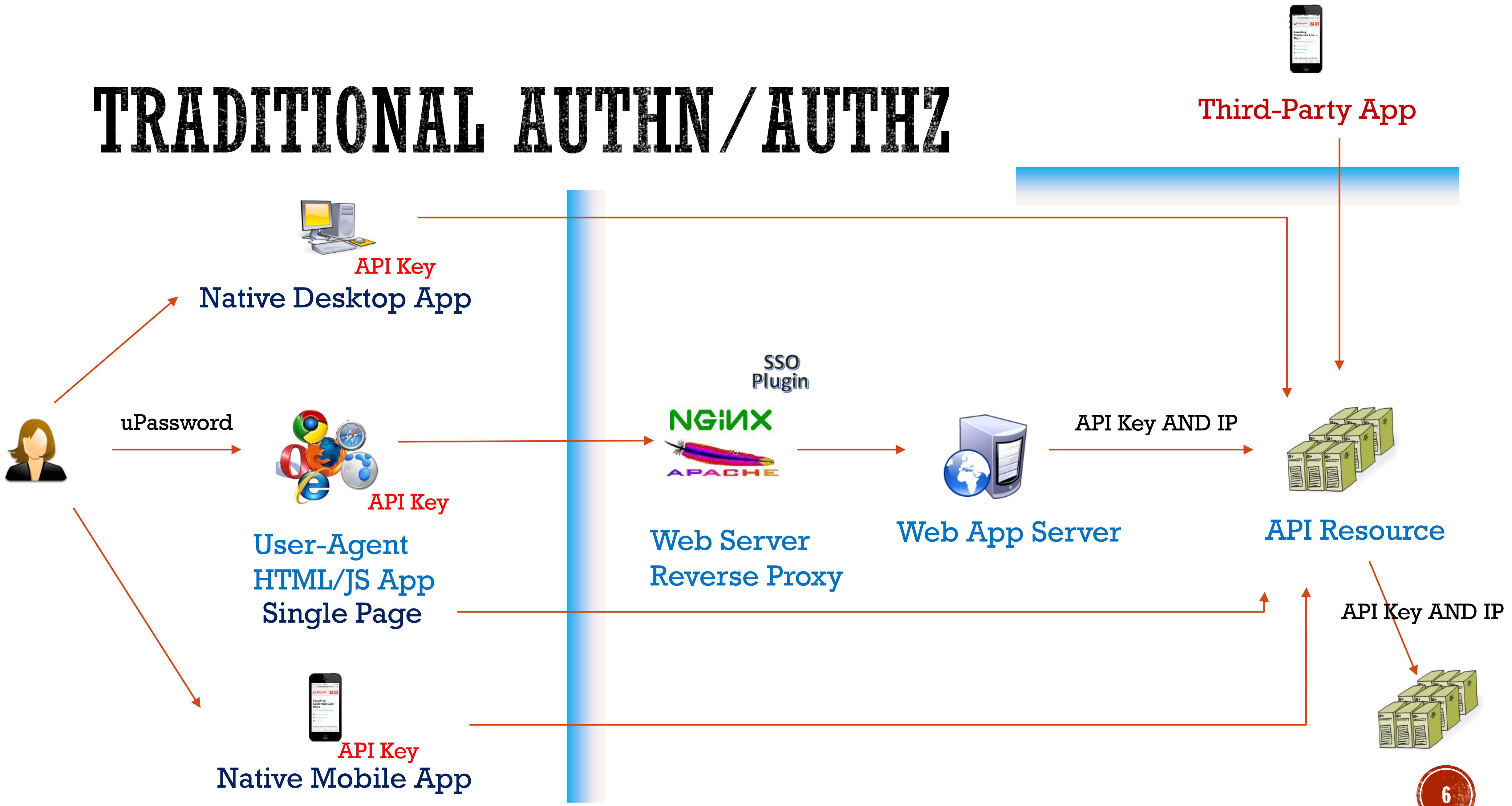
WHY HAS ROOM STARTED TO LOOK LIKE …

We will NOT be going through specifications.

Please note slide number for your questions

There will be breaks in between >> >>

TRADITIONAL AUTHN/AUTHZ



SECURE API PROBLEM? HOW DO THEY DO?

Google Developers

https://developers.google.com/

Linked in Developers

https://developer.linkedin.com/

facebook for developers

https://developers.facebook.com/

GitHub Developer

https://developer.github.com/

StackExchange

https://api.stackexchange.com/

Developer

https://developer.twitter.com/

THIRD-PARTY USE CASE

Emca Fraud
Protection service

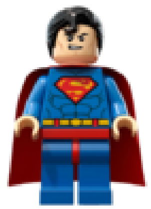
Emca a popular startup company provides fraud monitoring and protection service to banks **on behalf of** bank account owner

Emca needs from Bank

- Transaction logs API
- Stop transaction API

Acme Bank

Emca requires from A/C owner



- Account to be monitored
- Bank login credentials
- Or permission for account

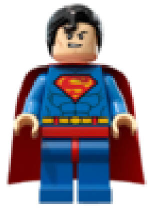
A/C Owner

NEED FOR ALTERNATE AUTHORIZATION

- Can Account Owner trust credentials with Emca?
- Can we limit Emca's access to read user's checking account and not his mortgage or trading accounts?
- Can we have a mobile app that can notify of events
  - What about session expiration
  - What if user restarts the phone

Emca

Acme Bank

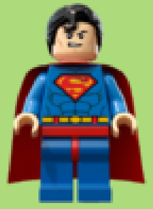

A/C Owner

NEED FOR ALTERNATE AUTHORIZATION

- Can Account Owner trust credentials with Emca?
- Can we limit Emca's access to read user's checking account and not his mortgage or trading accounts?
- Can we have a mobile app that can notify of events
  - What about session expiration
  - What if user restarts the phone

Emca

Acme Bank

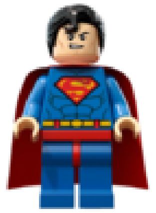

A/C Owner

Dec 2007 : OAuth 1.0
Oct 2012 : OAuth 2.0

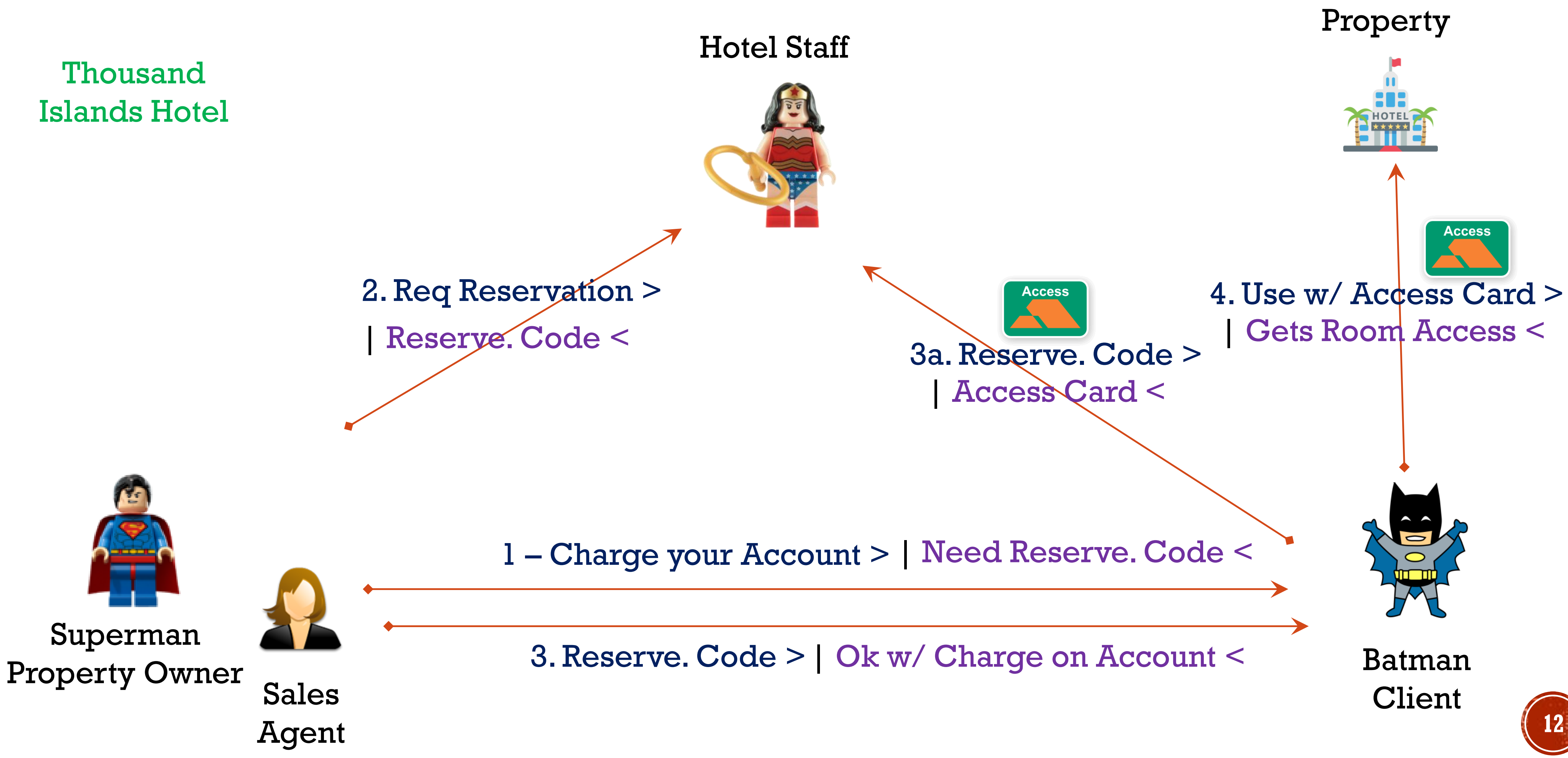
BIRTH OF OAUTH

- OAuth is open standard authorization framework for access delegation

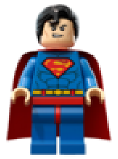
Authorizing third-party limited access to user resources



HOTEL RESERVATION FLOW ANALOGY



UNDERSTANDING OAUTH



On behalf of property owner (1)

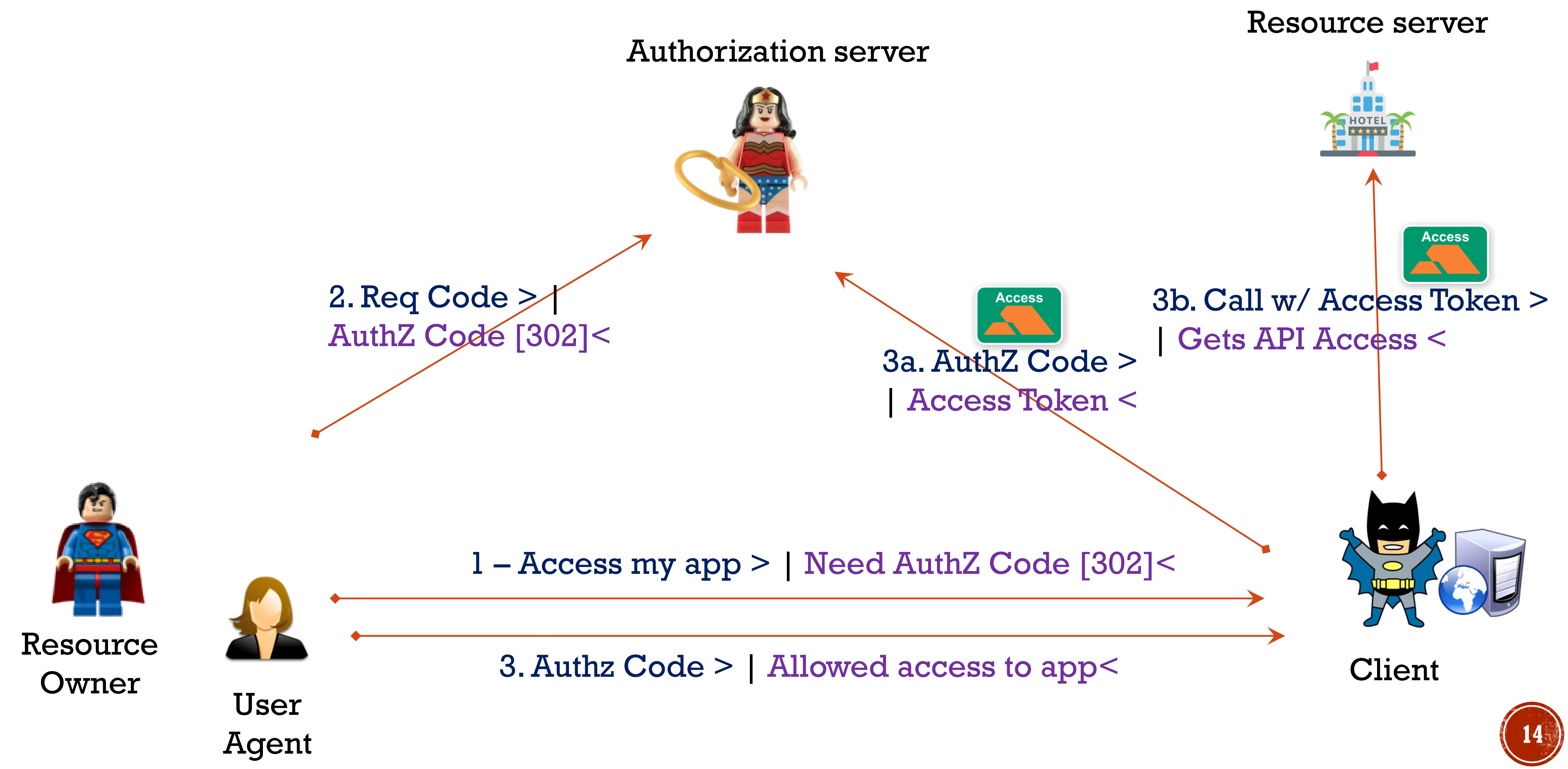
hotel staff (2) authorizes delegated access

of hotel room and facilities resources (3)

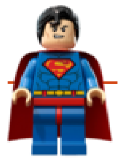
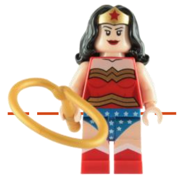
to it's guest client (4)

with limited access card (5)

OAUTH: AUTHORIZATION CODE FLOW

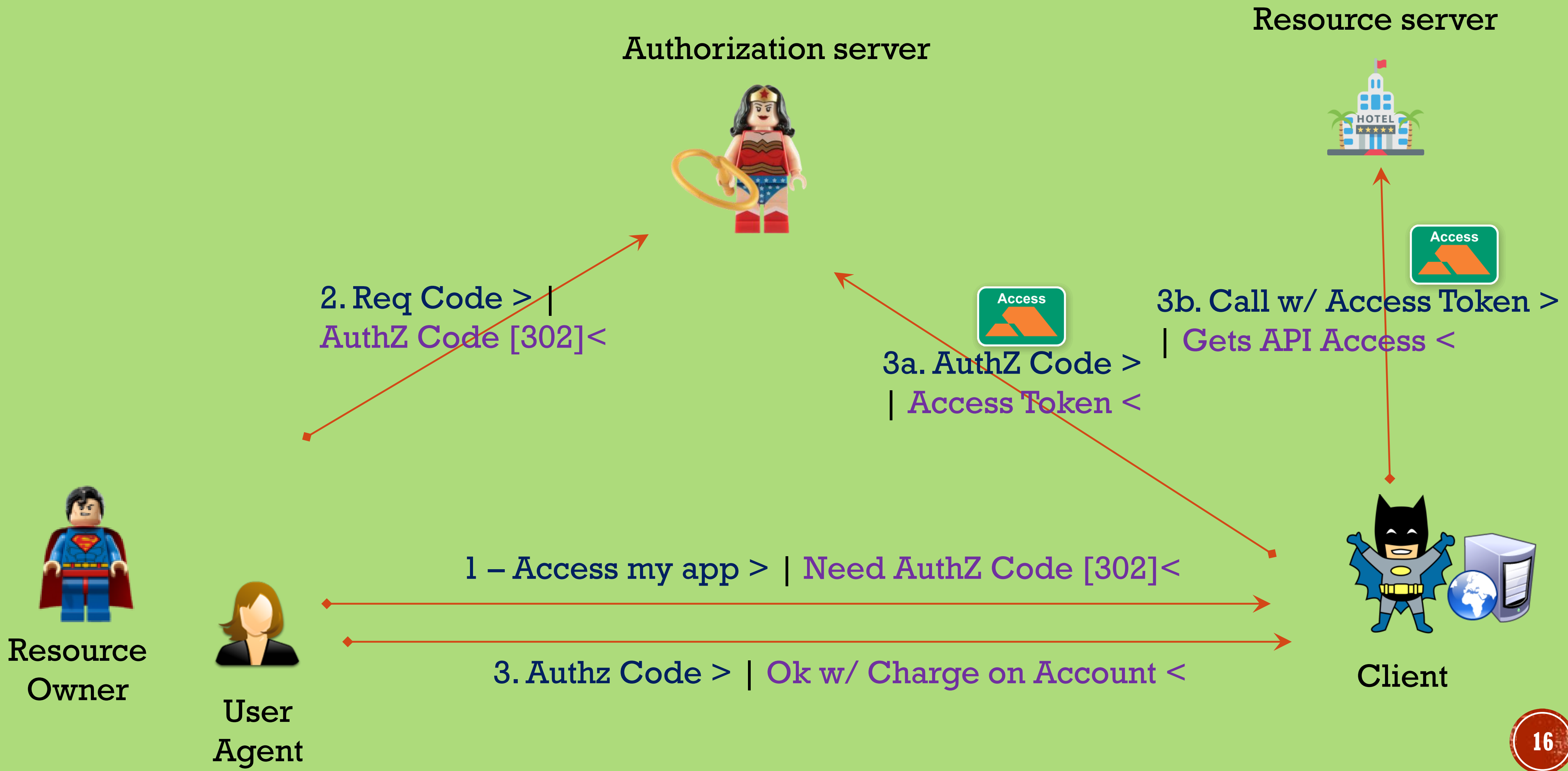

Resource server
Authorization server
Access
Access
2. Req Code > | AuthZ Code [302]<
3a. AuthZ Code > | Access Token <
3b. Call w/ Access Token > | Gets API Access <
1 – Access my app > | Need AuthZ Code [302]<
3. Authz Code > | Allowed access to app<
Resource Owner
User Agent
Client

OAUTH ROLES

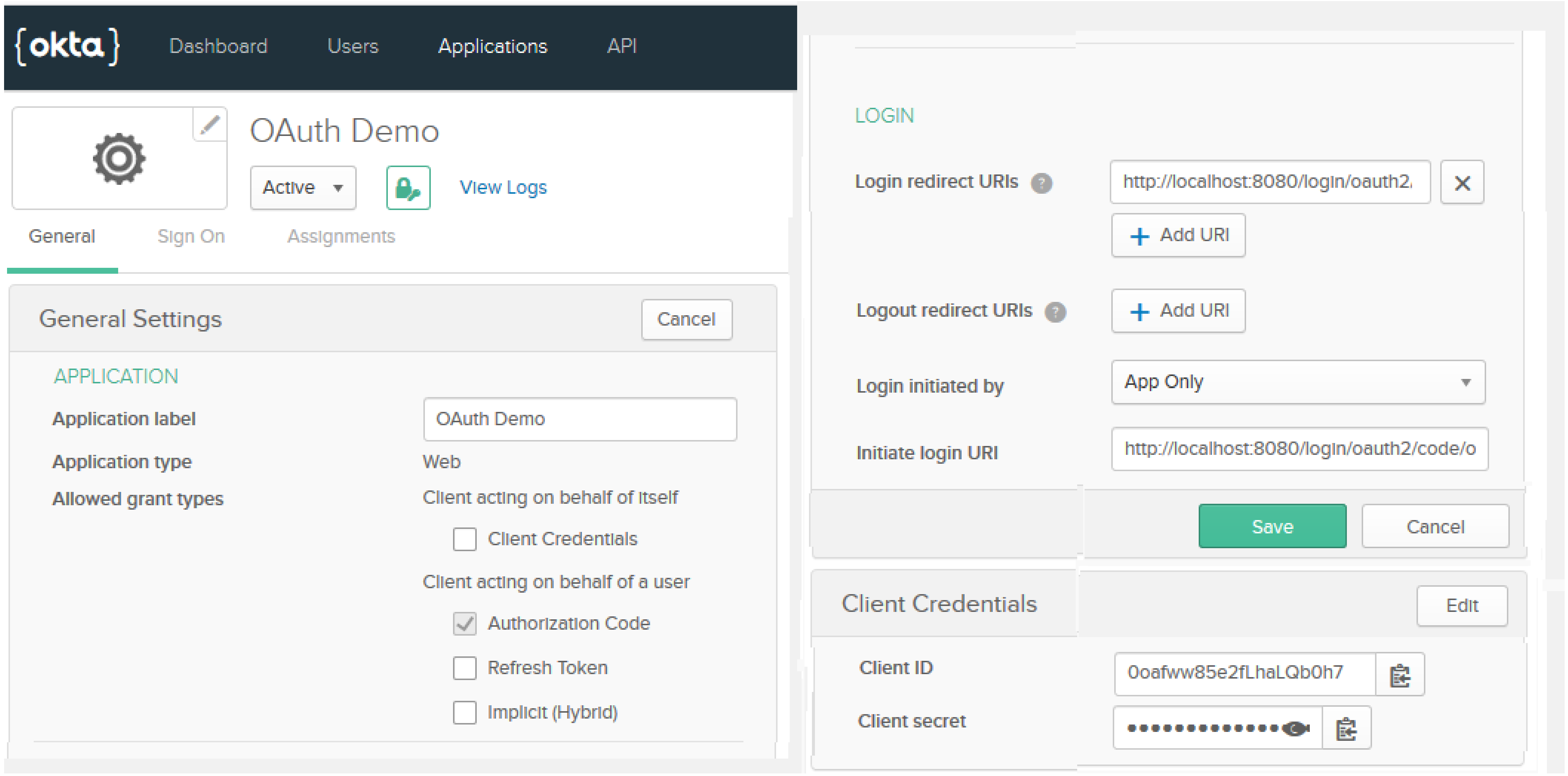
Superman (Property Owner)		Resource owner (the user)
Hotel Staff (hotel front desk reception)		Authorization server (can be the same server as the API)
Property (rooms and facilities)		Resource server (the protected API)
Batman Client (hotel guest)		Client (your app)
Access Card		*OAuth Token*
Sales Agent		*User Agent*

OAUTH: AUTHORIZATION CODE FLOW

BREAK



CLIENT REGISTRATION



OAuth Demo

Active | View Logs

General | Sign On | Assignments

General Settings — Cancel

APPLICATION

Application label	OAuth Demo
Application type	Web
Allowed grant types	Client acting on behalf of itself
	☐ Client Credentials
	Client acting on behalf of a user
	☑ Authorization Code
	☐ Refresh Token
	☐ Implicit (Hybrid)

LOGIN

Login redirect URIs	http://localhost:8080/login/oauth2,
	+ Add URI
Logout redirect URIs	+ Add URI
Login initiated by	App Only
Initiate login URI	http://localhost:8080/login/oauth2/code/o

Save | Cancel

Client Credentials — Edit

Client ID	0oafww85e2fLhaLQb0h7
Client secret	••••••••••••••

REDIRECT URL

- After a user authorizes an application, the authorization server will redirect the user back to this URL
- Redirect URL will be appended with sensitive information, so it is critical it is not arbitrary/open

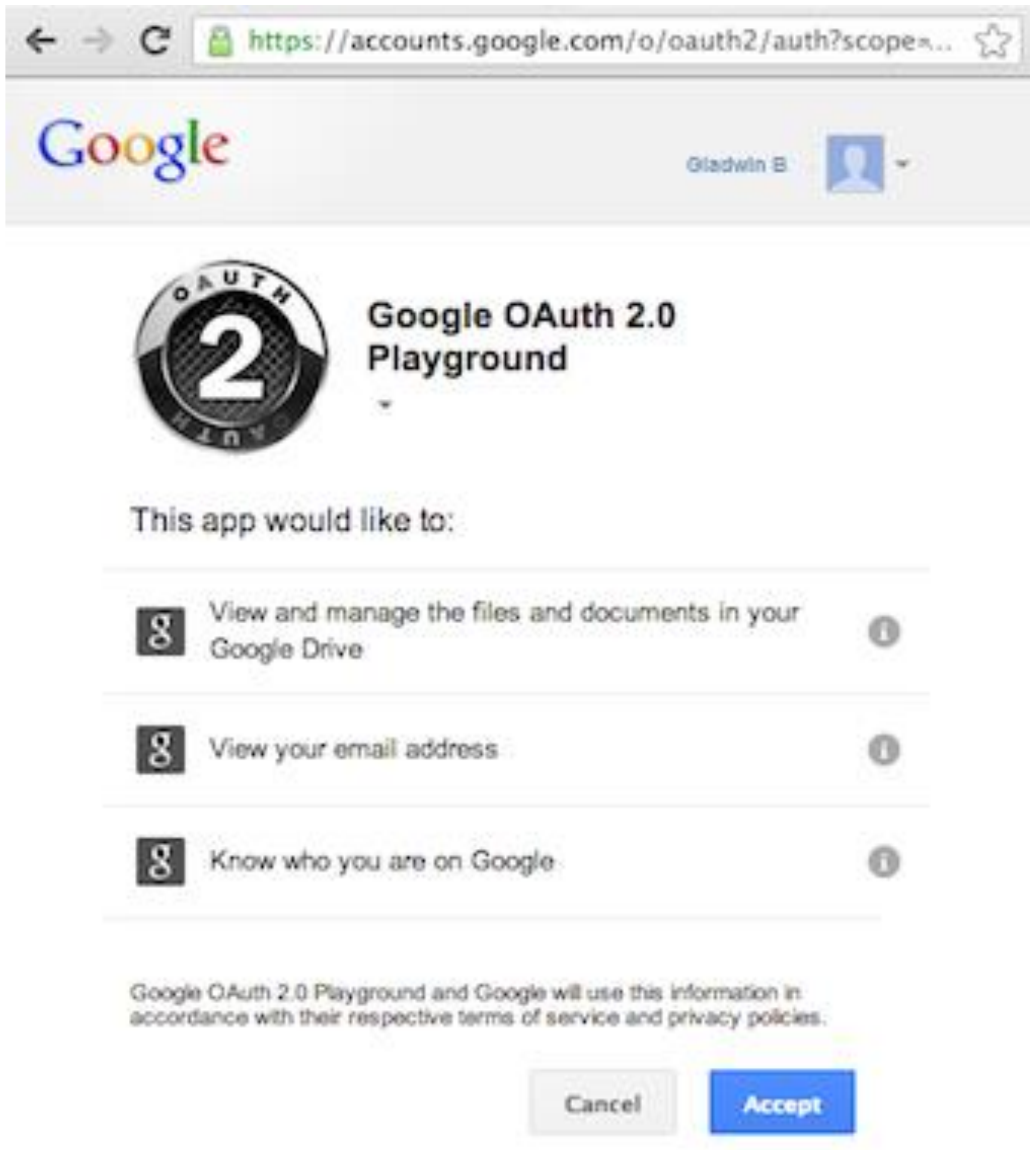
SCOPE

- provide a way to limit the amount of access that is granted
  - read , write, admin, profile, email,

OAUTH CONSENT SCREEN

- Authorization server explicitly asks user if listed permissions can be given to the Client

- Who the Client is (from registration)
- Permission List (from scope)
- For how long will this be for
- How can it be revoked if need be


https://accounts.google.com/o/oauth2/auth?scope=...

Google

Gladwin B

Google OAuth 2.0 Playground

This app would like to:

View and manage the files and documents in your Google Drive

View your email address

Know who you are on Google

Google OAuth 2.0 Playground and Google will use this information in accordance with their respective terms of service and privacy policies.

http://tools.ietf.org/html/rfc6750

BEARER TOKEN

- Bearer Tokens are the popular Access Token used with OAuth 2.0
- Any party in possession of a bearer token (a "bearer") can use it to get access
- It is opaque string with no meaning to it’s client
- Some may use string of random characters like UUID
- Others may use more structured tokens like SAML or JWT

http://tools.ietf.org/html/rfc6750

REFRESH TOKEN

- Access Tokens may have shorter expiry
- Can be used to check back token revocation by user
- Also check if user has his resource privileges revoked or changed
- Additional token in payload called Refresh Token
- Refresh token have additional future expiry

CONFUSING, TERM "CLIENT"

- Client is an application that
  - requests OAuth Token from provider and
  - uses it to access Resource on behalf of user.

  - Batman is client application that requests OAuth Token from Wonder Women

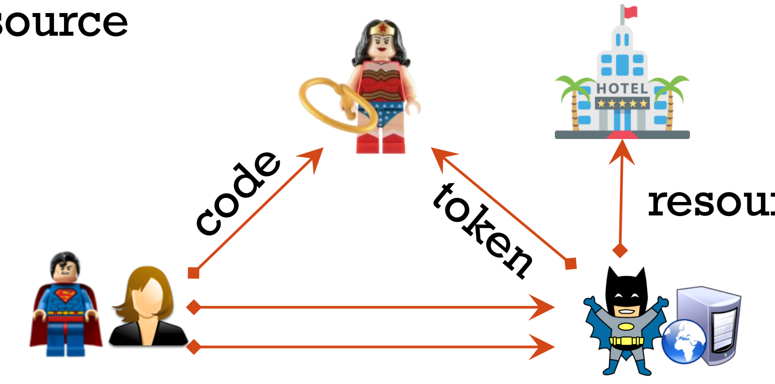
  - Batman uses token on behalf of Superman to access his protected resource

- Client is registered with OAuth Provider

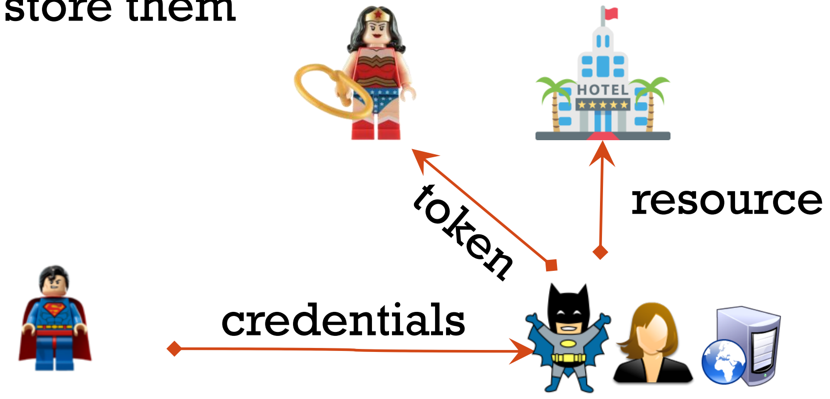
also called as
Relying Party

OAUTH FLOWS

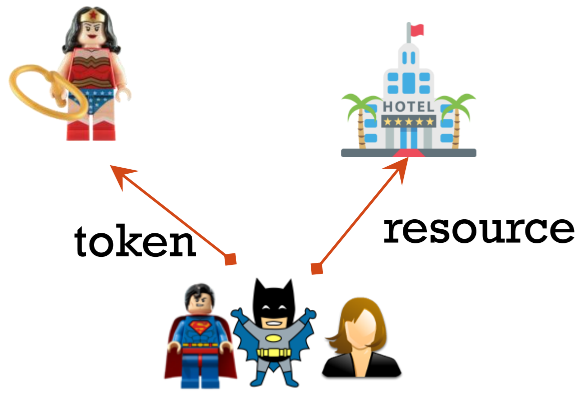
- Authorization Code Grant
  - App on Server side needs access to user's resource



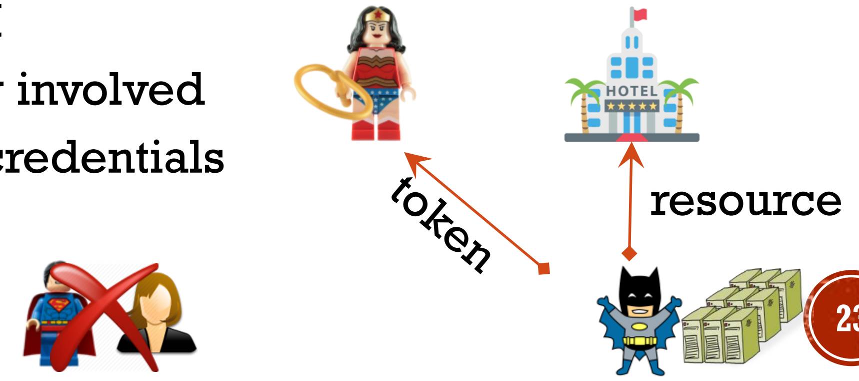
- Resource Owner Password Credential Grant
  - App can be trusted w/ user credentials, but does not store them



- Implicit Grant
  - App on User Agent needs access to user's resources



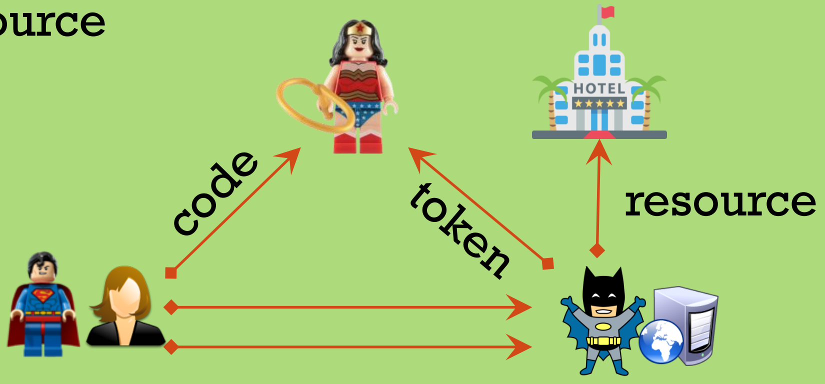
- Client Credentials Grant
  - API-API
  - No user involved
  - Client credentials



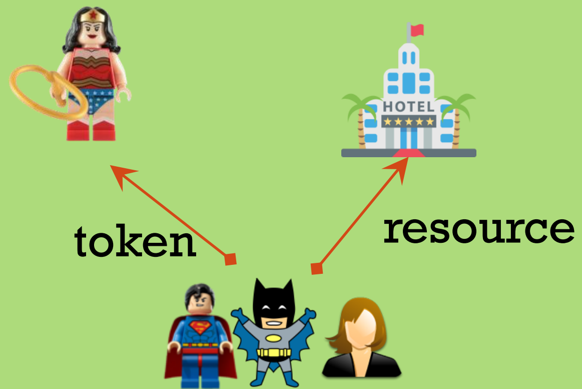
OAUTH FLOWS

BREAK

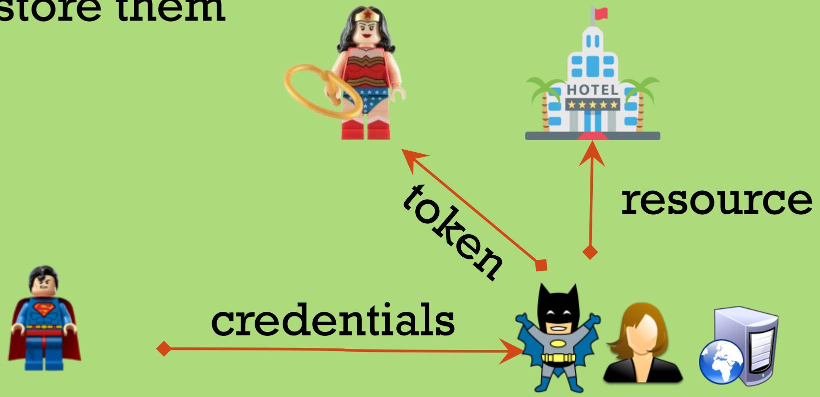
- Authorization Code
  - App on Server side needs access to user's resource



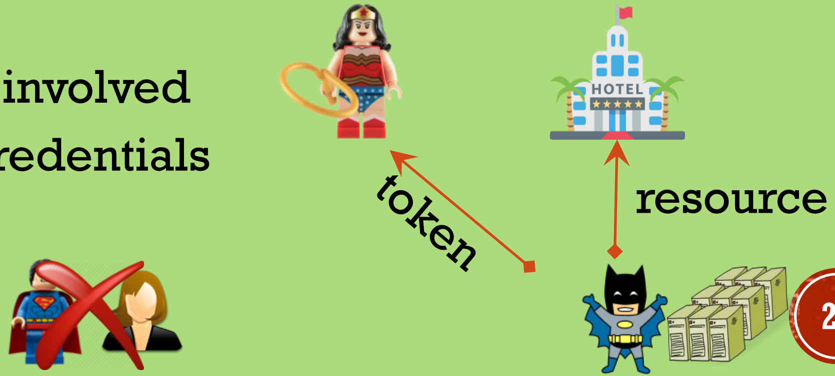
- Implicit Grant
  - App on User Agent needs access to user's resources



- Resource Owner Password Credential Grant
  - App can be trusted w/ user credentials, but does not store them



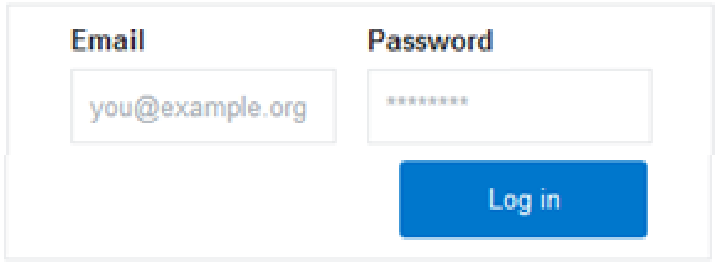
- Client Credentials
  - API-API
  - No user involved
  - Client credentials



PLEASE NO MORE NEW ACCOUNTS …

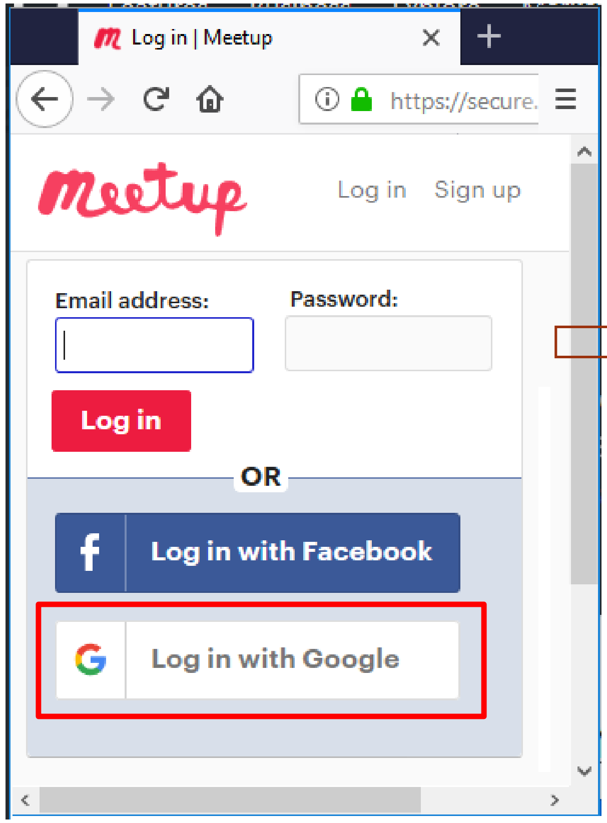
Is it secure when user has to remember so many username/password?

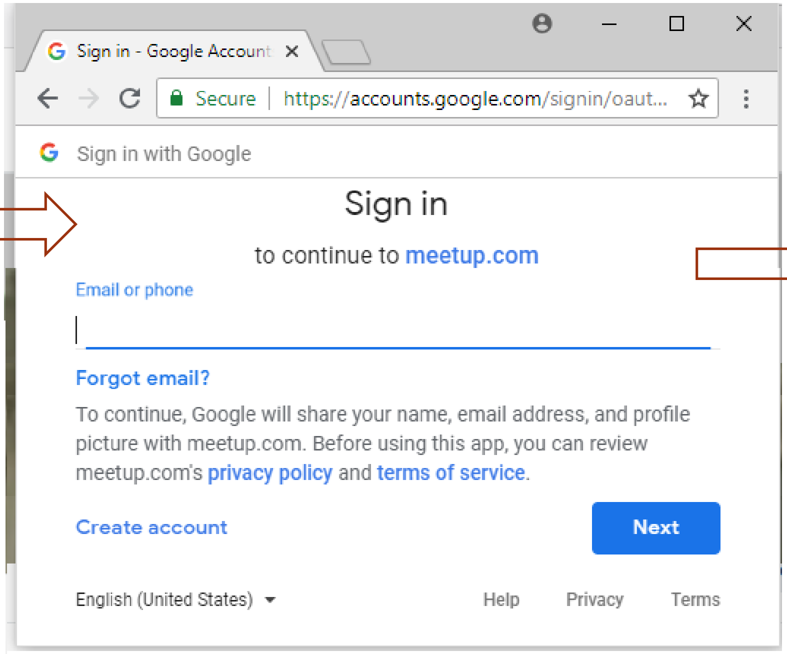
- **Example: Password Rules**
  - Minimum 8 characters long
  - At least three of following
    - upper case and lower case letter
    - Digit 0-9 and Special character @ # $ % ^
    - Change once every 30 days

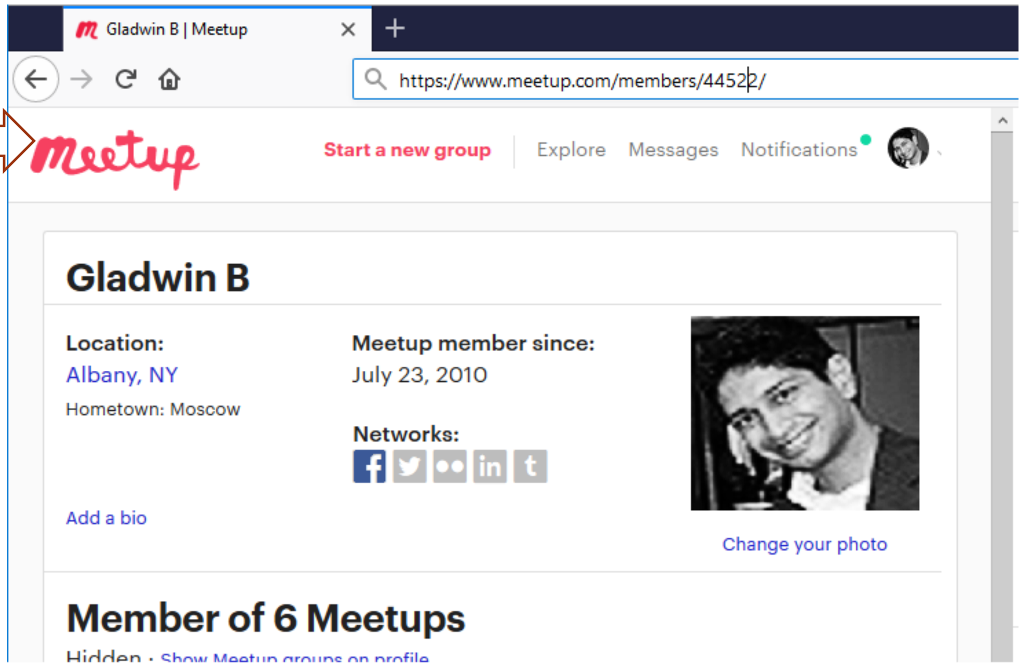


- Multiply this **pain to manage** different **passwords with all the accounts** that you have

LOG IN WITH …




Sign in - Google Account
Secure
https://accounts.google.com/signin/oaut...
Sign in with Google
Sign in
to continue to meetup.com
Email or phone
Forgot email?
To continue, Google will share your name, email address, and profile picture with meetup.com. Before using this app, you can review meetup.com's privacy policy and terms of service.
Create account
Next
English (United States)
Help
Privacy
Terms



How much effort do you think it must have taken to implement

SSO Federation between these two Enterprise organizations?

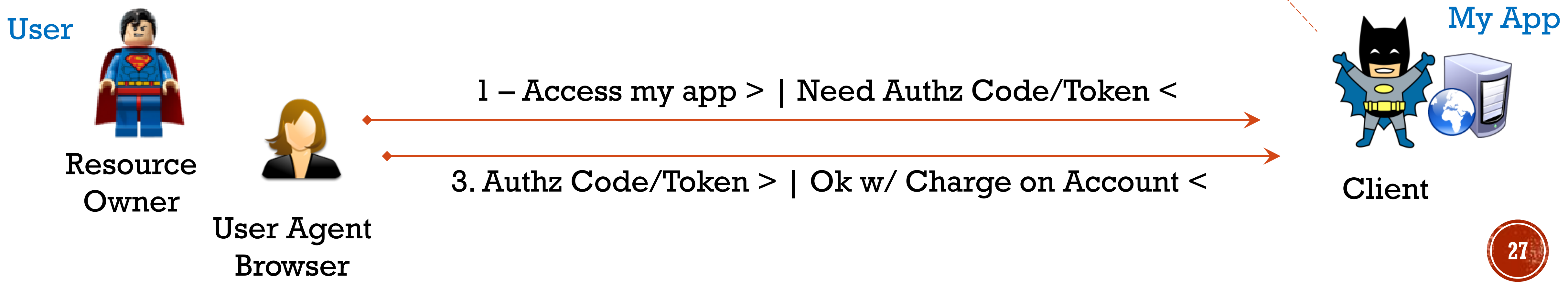
OAUTH FOR AUTHENTICATION

Eureka!

Can we somehow use this for AuthN?

Since we were got valid Access Token
to access User Superman's resource

This user must be a Superman


User
Resource Owner
User Agent Browser
1 – Access my app > | Need Authz Code/Token <
3. Authz Code/Token > | Ok w/ Charge on Account <
My App
Client

OAUTH 2.0 WITH SPRING SECURITY

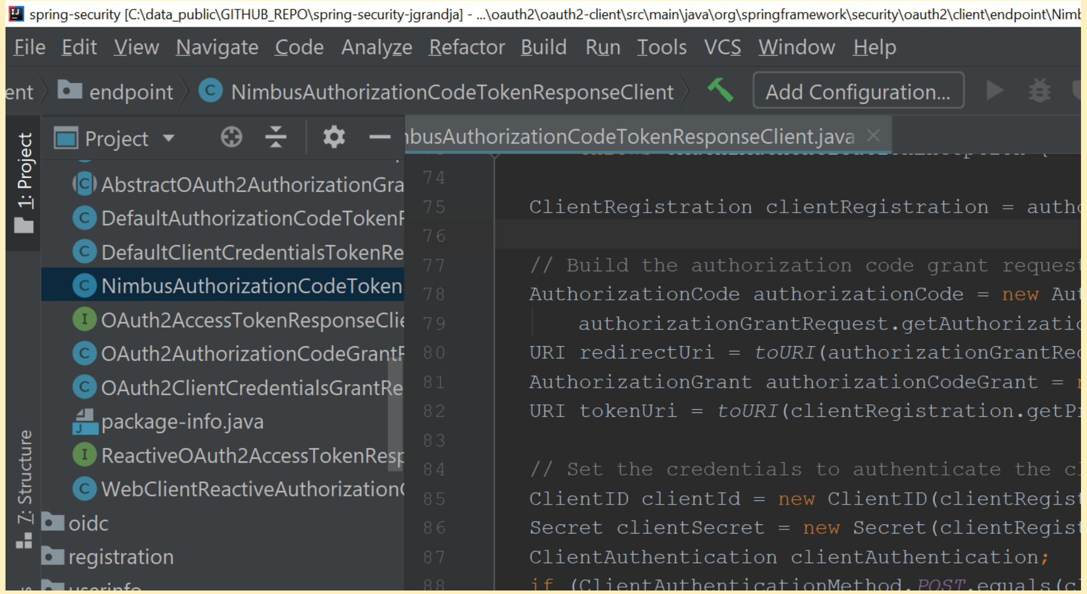
- OAuth 2.0 Support, within the Spring projects portfolio, is spread out between Spring Security OAuth, Spring Cloud Security, Spring Boot 1.5.x
- New support introduced in **Spring Security 5**
  - More extensive support for OAuth 2.0 and OpenID Connect 1.0
  - New Client support while, Resource Server and Authorization Server coming soon

https://spring.io/blog/2018/01/30/next-generation-oauth-2-0-support-with-spring-security

SPRING BOOT DEMO

https://github.com/gburboz/gb-oauth2-springboot-talk

- Step-01-InitialSpringSecurityApp
- Step-02-OAuthSimpleSocialLogin



spring-security [C:\data_public\GITHUB_REPO\spring-security-jgrandja] - ...\oauth2\oauth2-client\src\main\java\org\springframework\security\oauth2\client\endpoint\Nimb

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
ent endpoint NimbusAuthorizationCodeTokenResponseClient
Add Configuration...
Project
1: Project
7: Structure

```
AbstractOAuth2AuthorizationGra
DefaultAuthorizationCodeTokenF
DefaultClientCredentialsTokenRe
NimbusAuthorizationCodeToken
OAuth2AccessTokenResponseClie
OAuth2AuthorizationCodeGrantF
OAuth2ClientCredentialsGrantRe
package-info.java
ReactiveOAuth2AccessTokenResp
WebClientReactiveAuthorization(
oidc
registration
```

ıbusAuthorizationCodeTokenResponseClient.java

```
    ClientRegistration clientRegistration = authc

    // Build the authorization code grant request
    AuthorizationCode authorizationCode = new Aut
        authorizationGrantRequest.getAuthorizatic
    URI redirectUri = toURI(authorizationGrantRec
    AuthorizationGrant authorizationCodeGrant = r
    URI tokenUri = toURI(clientRegistration.getPr

    // Set the credentials to authenticate the cl
    ClientID clientId = new ClientID(clientRegist
    Secret clientSecret = new Secret(clientRegist
    ClientAuthentication clientAuthentication;
    if (ClientAuthenticationMethod.POST.equals(cl
```

Lets try some Social Login code!

- **Spring Boot 2.x**
- **Spring Security 5**

OAuth 2.0 Login Sample

ERAN HAMMER

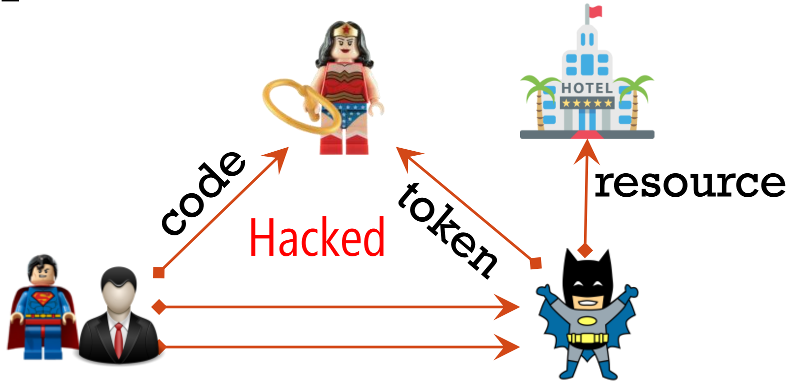
- Eran Hammer the main editor of OAuth 2.0 specification
  - After 3 years left the specification committee
  - RealtimeConf - OAuth 2.0 - Looking Back and Moving On
  - Blog - OAuth 2.0 and the Road to Hell

✓ Specs are open (may)
✓ Bearer token
✓ Standard claims

- Some points about security concerns about OAuth 2.0
  - Some have been addressed by Open ID Connect
  - Too many responsibilities on Client

OAUTH MISUSED – NOT DESIGNED FOR AUTHN

https://oauth.net/articles/authentication/



OAuth Misused

- No explicit AuthN or User ID
- Where to find User Info?
  - Not standardized
- Who is audience of token?
  - Client can impersonate as user

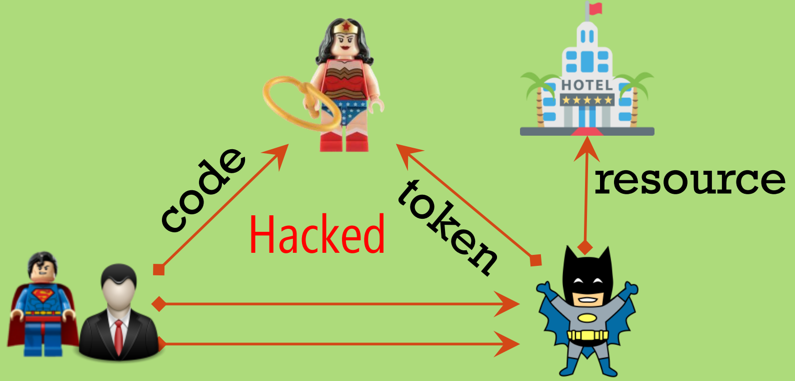
Open ID Connect

- New scope "openid"
- id_token
  - JSON Web Token (JWT)
  - Has special claim called "sub"
  - "aud" must match client_id
  - …
- User Info Endpoint w/ standard claims

BREAK

OAuth Misused – Not Designed for AuthN

https://oauth.net/articles/authentication/



OAuth Misused

- No explicit AuthN or User ID
- Where to find User Info?
  - Not standardized
- Who is audience of token?
  - Client can impersonate as user

Open ID Connect

- New scope "openid"
- id_token
  - JSON Web Token (JWT)
  - Has special claim called "sub"
  - "aud" must match client_id
  - …
- User Info Endpoint w/ standard claims

Nov 2014 : OIDC 1.0

BIRTH OF OPEN ID CONNECT

- OpenID Connect 1.0 (OIDC) is authentication protocol layer on top of OAuth 2.0
  - Based of previously known OpenID specs
  - Uses Authorization Code and Implicit OAuth flows but standardizes certain aspects for authentication purpose
- OAuth is open standard authorization framework for access delegation

SPRING BOOT DEMO

https://github.com/gburboz/gb-oauth2-springboot-talk

- Step-01-InitialSpringSecurityAppp
- Step-02-OAuthSimpleSocialLogin
- Step-03-OpenIDConnectLogin

https://en.wikipedia.org/wiki/JSON_Web_Token

https://jwt.io/

JSON WEB TOKEN [JWT]

- Base 64 encoded strings: `<header>.<payload>.<signature>`

```
{"alg": "HS256", "typ": "JWT"} .
{"sub": "1234567890", "iat": 1516239022,
  "name": "John Doe", "email": "john.doe@jwt.io",
  "athorities" : ["admin", "test"]} .
fmJX_Nk-gvrE-WU1Czs_Et-kVAeWATR1VorpjUCOg8E
```

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiZW1ha
WwiOiJqb2huLmRvZUBqd3QuaW8iLCJpYXQiOjE1MTYyMzkwMjIsImF0aG
9yaXRpZXMiOlsiYWRtaW4iXX0.fmJX_Nk-gvrE-WU1Czs_Et-
kVAeWATR1VorpjUCOg8E
```

JSON WEB TOKEN [JWT]

- Open industry standard RFC 7519 for representing claims securely
- Spring Security 5 uses connect2id Nimbus JOSE + JWT library
- Access token can be JWT but are not required to be
- Open ID Connect ID token is required to be JWT

✓ Never let the JWT header alone drive verification
✓ Know the algorithms
✓ Use an appropriate key size

OPENID CONNECT – ENDPOINTS

ID Token is Signed JWT

Google Accounts
Identity Provider (IdP)

Google
Authorization Server
(OpenID Connect Provider)

Google Profile
Resource Server

Authentication

Input user Credentials

Authenticates and establishes User Identity

AuthorizationEndpoint

Input Client ID, Client Redirect URL, scope, and AuthN User

Provides **Authorization Code** for Client to access token/claims

TokenEndpoint

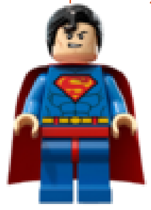
Input Client ID/Secret, Authorization Code

Provides **ID Token** w/ subject info and **Access Token** that can be used to access resources

UserInfoEndpoint

Input Access Token

Access to **Resource** / Standard Claims



Superman
User Agent

Access Client App

Sends Authorization Code

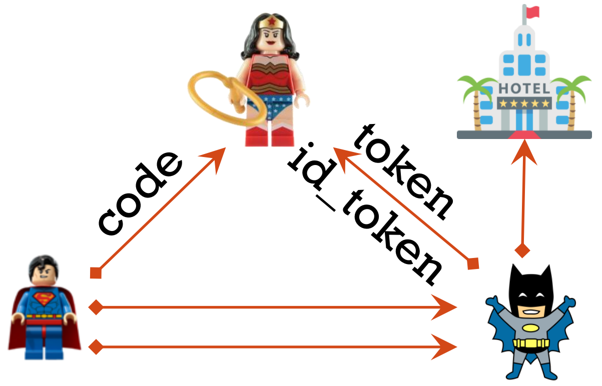
Needs to know User

Authenticated Access

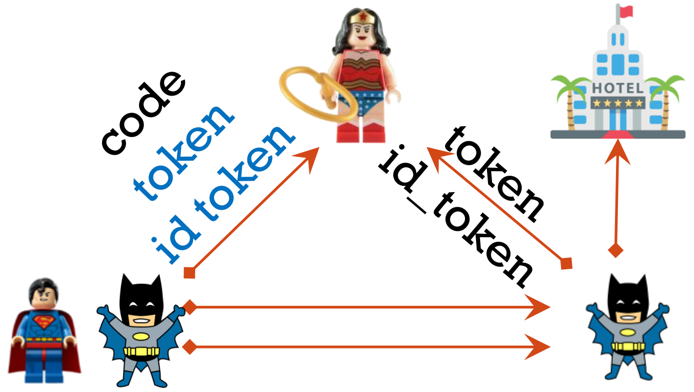
Stack overflow
Client / Relying Party
Client Secret

OPENID CONNECT FLOWS

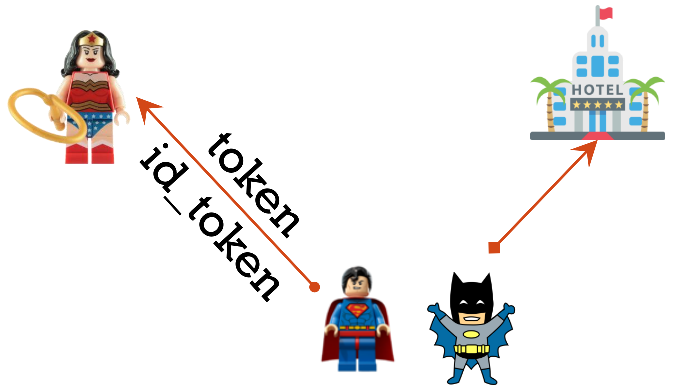
Authorization Code Flow



Hybrid Flow *(can refresh tokens)*



Implicit Flow



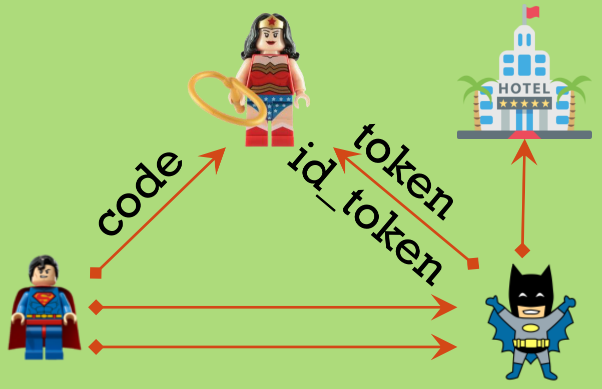
OIDC Grant

- ❑ code
  - Intermediate code returned by authz endpoint that can be used to request tokens
- ❑ id_token
  - Identity Token that contains user identity
- ❑ token
  - Token that can be used to access resources

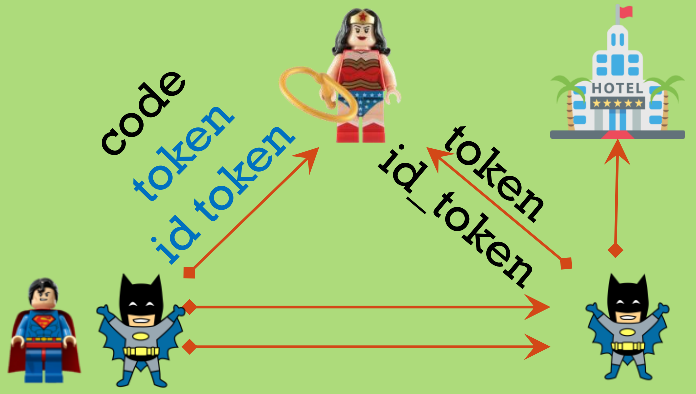
BREAK

OPENID CONNECT FLOWS

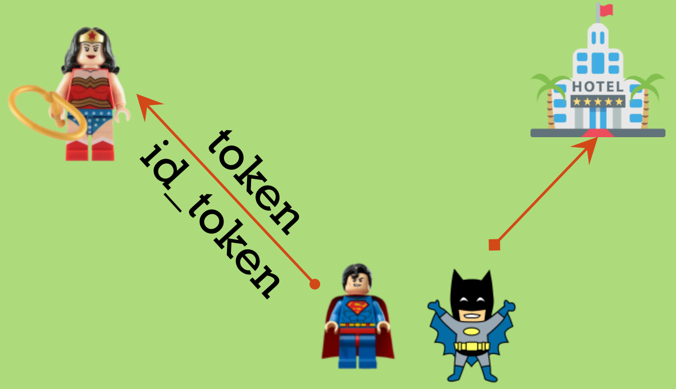
Authorization Code Flow



Hybrid Flow *(can refresh tokens)*



Implicit Flow



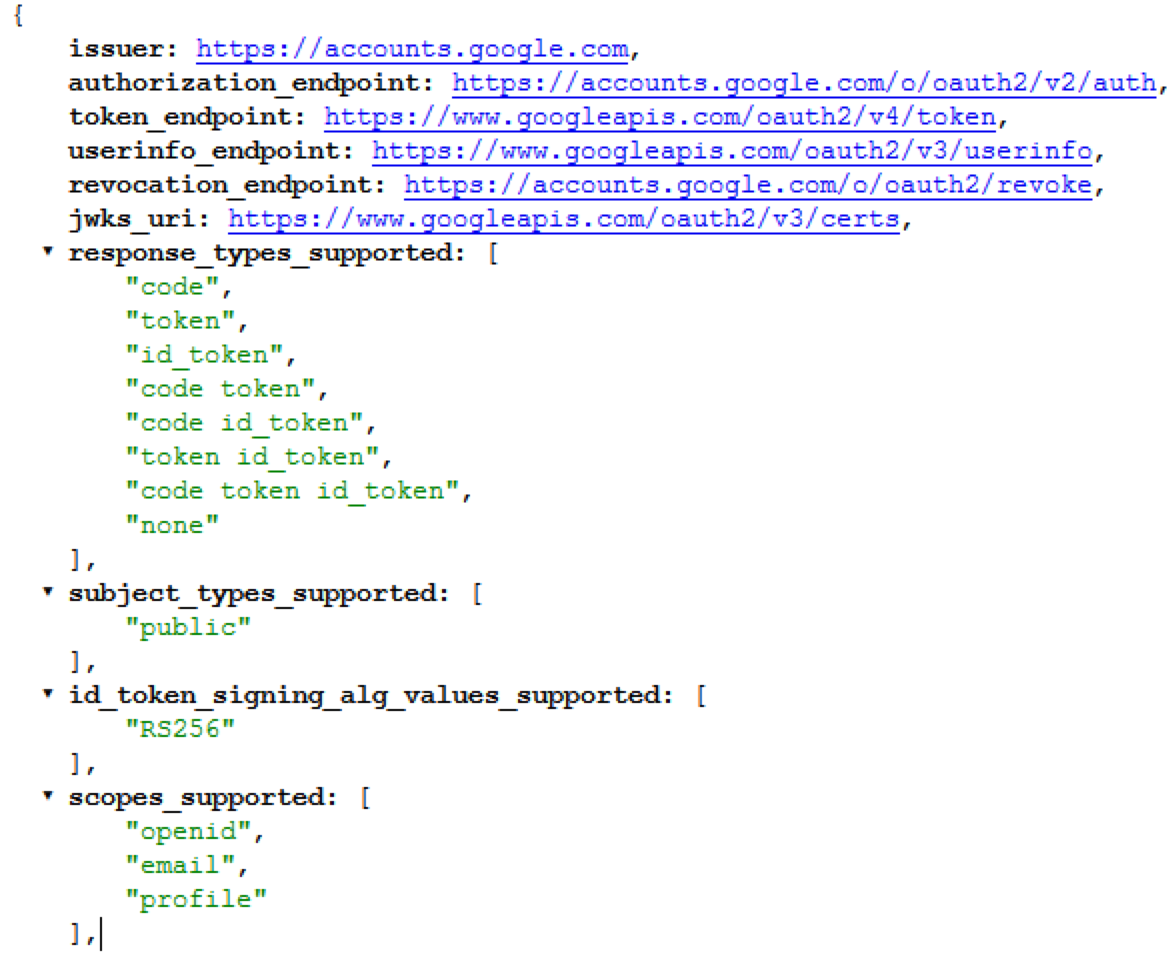
OIDC Grant

- code
  - Intermediate code returned by authz endpoint that can be used to request tokens
- id_token
  - Identity Token that contains user identity
- token
  - Token that can be used to access resources

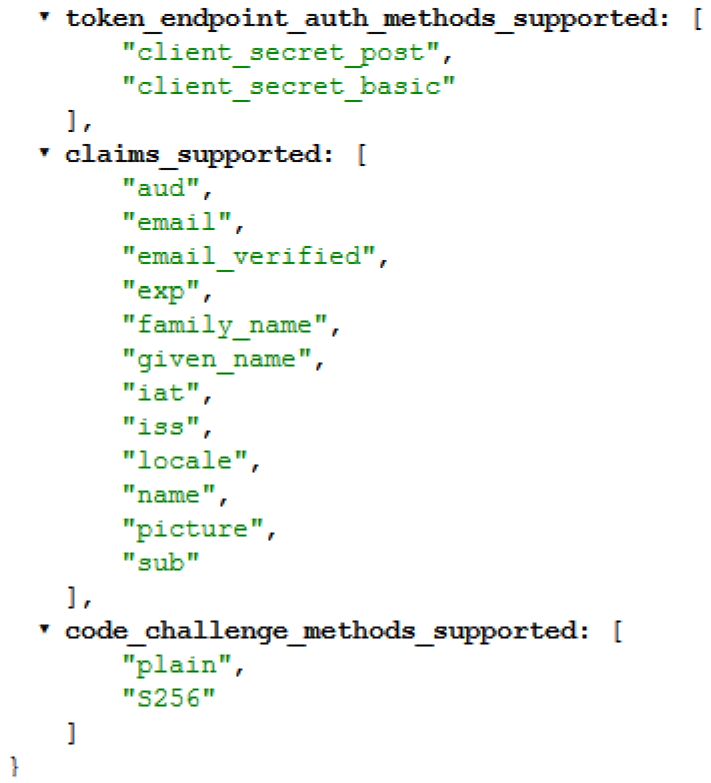
https://openid.net/specs/openid-connect-discovery-1_0.html
https://accounts.google.com/.well-known/openid-configuration

OPENID CONNECT DISCOVERY

Returns provider info in JSON Format



```
{
    issuer: https://accounts.google.com,
    authorization_endpoint: https://accounts.google.com/o/oauth2/v2/auth,
    token_endpoint: https://www.googleapis.com/oauth2/v4/token,
    userinfo_endpoint: https://www.googleapis.com/oauth2/v3/userinfo,
    revocation_endpoint: https://accounts.google.com/o/oauth2/revoke,
    jwks_uri: https://www.googleapis.com/oauth2/v3/certs,
  ▾ response_types_supported: [
        "code",
        "token",
        "id_token",
        "code token",
        "code id_token",
        "token id_token",
        "code token id_token",
        "none"
    ],
  ▾ subject_types_supported: [
        "public"
    ],
  ▾ id_token_signing_alg_values_supported: [
        "RS256"
    ],
  ▾ scopes_supported: [
        "openid",
        "email",
        "profile"
    ],
```



```
  ▾ token_endpoint_auth_methods_supported: [
        "client_secret_post",
        "client_secret_basic"
    ],
  ▾ claims_supported: [
        "aud",
        "email",
        "email_verified",
        "exp",
        "family_name",
        "given_name",
        "iat",
        "iss",
        "locale",
        "name",
        "picture",
        "sub"
    ],
  ▾ code_challenge_methods_supported: [
        "plain",
        "S256"
    ]
}
```

SPRING BOOT DEMO

https://github.com/gburboz/gb-oauth2-springboot-talk

- Step-01-InitialSpringSecurityAppp
- Step-02-OAuthSimpleSocialLogin
- Step-03-OpenIDConnectLogin
- Step-04-CustomProviderLogin

CLIENT SECURITY CONSIDERATIONS

- Man in the middle attack
  - Use TLS (`https`) to secure communication channel
- Verify standard claims
  - `iss, aud, exp, iat,`
- Verify JWT signature and algorithm
- Sensitive information exposure (`query parameters`)
  - Do not pass client secret, id token or access token as query parameter

CLIENT SECURITY CONSIDERATIONS

- CSRF Attack
  - Use `state` parameter with high entropy
- Open Redirectors
  - `https://my-legit-site.com/login?targetUrl=`~~`/home`~~*`http://evil-site.com`*
- Injection attacks
  - Validate incoming data values and context encoding before rendering
- OAuth is only for authorization,
  - For authentication use OpenID Connect

and many more …

PLAYERS ON OAUTH/OIDC

LinkedIn® Developers

Google Developers

Login with amazon

IBM DataPower Gateway

GitHub Developer

Microsoft Active Directory

facebook for developers

Microsoft Azure

apigee

ca technologies

StackExchange

Ping Identity

Developer

ZMARTZONE

KEYCLOAK

https://openid.net/developers/certified/

Gladwin Burboz [Speaker]

THANKS EVERYONE.

OAuth/OIDC
Under the hood of social login