OAuth Server functionality in AEM

Embrace Federation and unleash your REST APIs!

Antonio Sanso (@asanso)
Software Engineer
Adobe Research Switzerland

Who is this guy, BTW?

```
eyJhbGciOiJIUzI1NiIsInR5cCI6Ik
pXVCJ9.eyJhdWQiOiJjb25uZWN0MjA
xNCIsImlzcyI6ImFzYW5zbyIsInN1Y
iI6ImFzYW5zbyIsImV4cCI6MTQwMzY
wMTU1OSwiaWF0IjoxNDAzNjAxNTU5f
Q.9-
MaGUiPg07ezuP9yAOaVLETQH6HMOpf
oGwg_c0-PDw
```

Who is this guy, BTW?

{ Software Engineer Adobe Research Switzerland

{ VP (Chair) Apache Oltu (OAuth protocol implementation in Java)

{ Committer and PMC member for Apache Sling

{ Google Security hall of fame, Facebook security whitehat

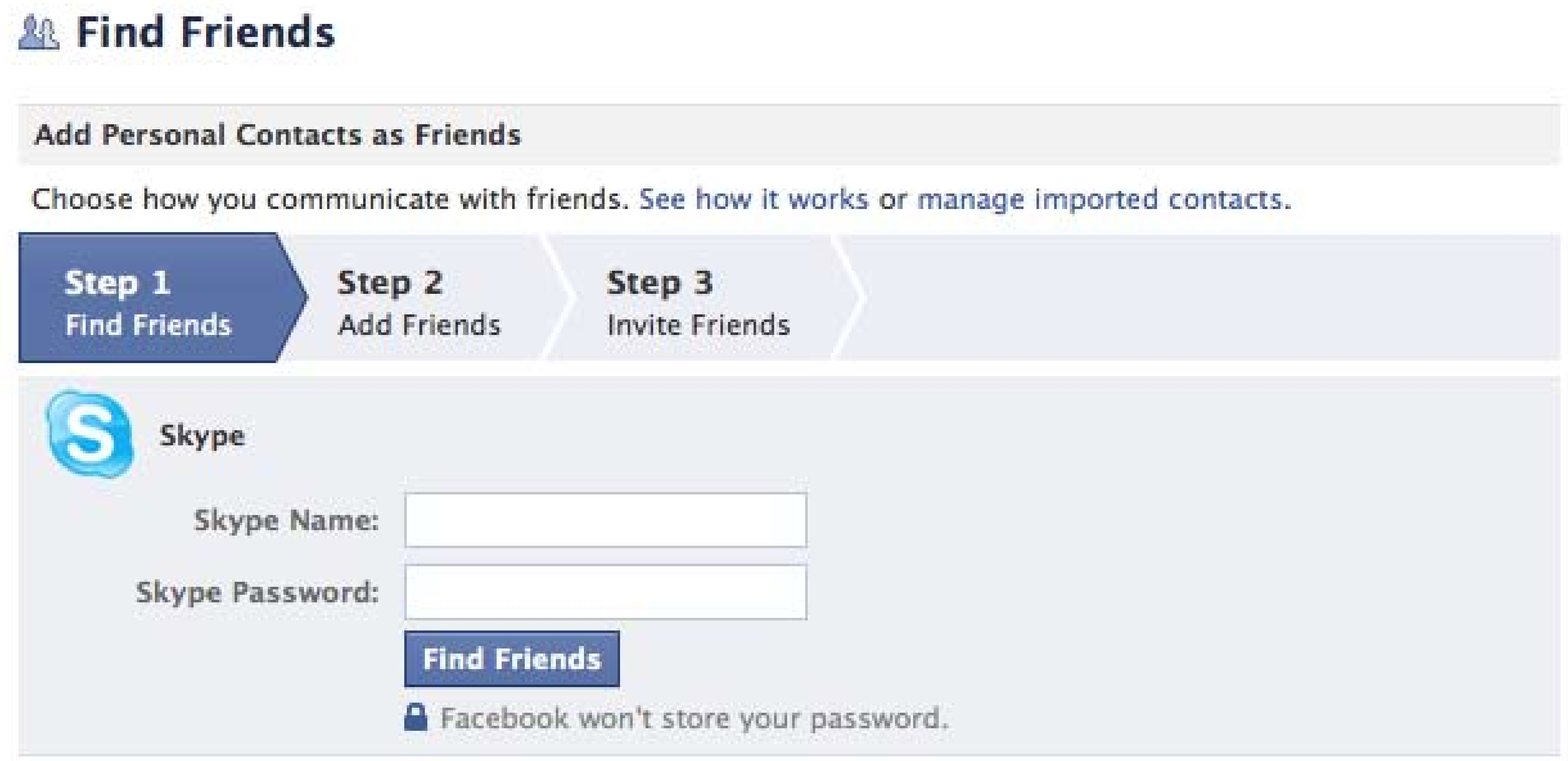
Agenda

- { OAuth introduction
- { "OAuth dance"
- { Implementing OAuth
- { AEM and OAuth
- { Extend Authentication in AEM
- { OAuth server to server

Why OAuth?

Several web sites offer you the chance to import the list of your contacts.

It ONLY requires you giving your username and password.



Find Friends

Add Personal Contacts as Friends

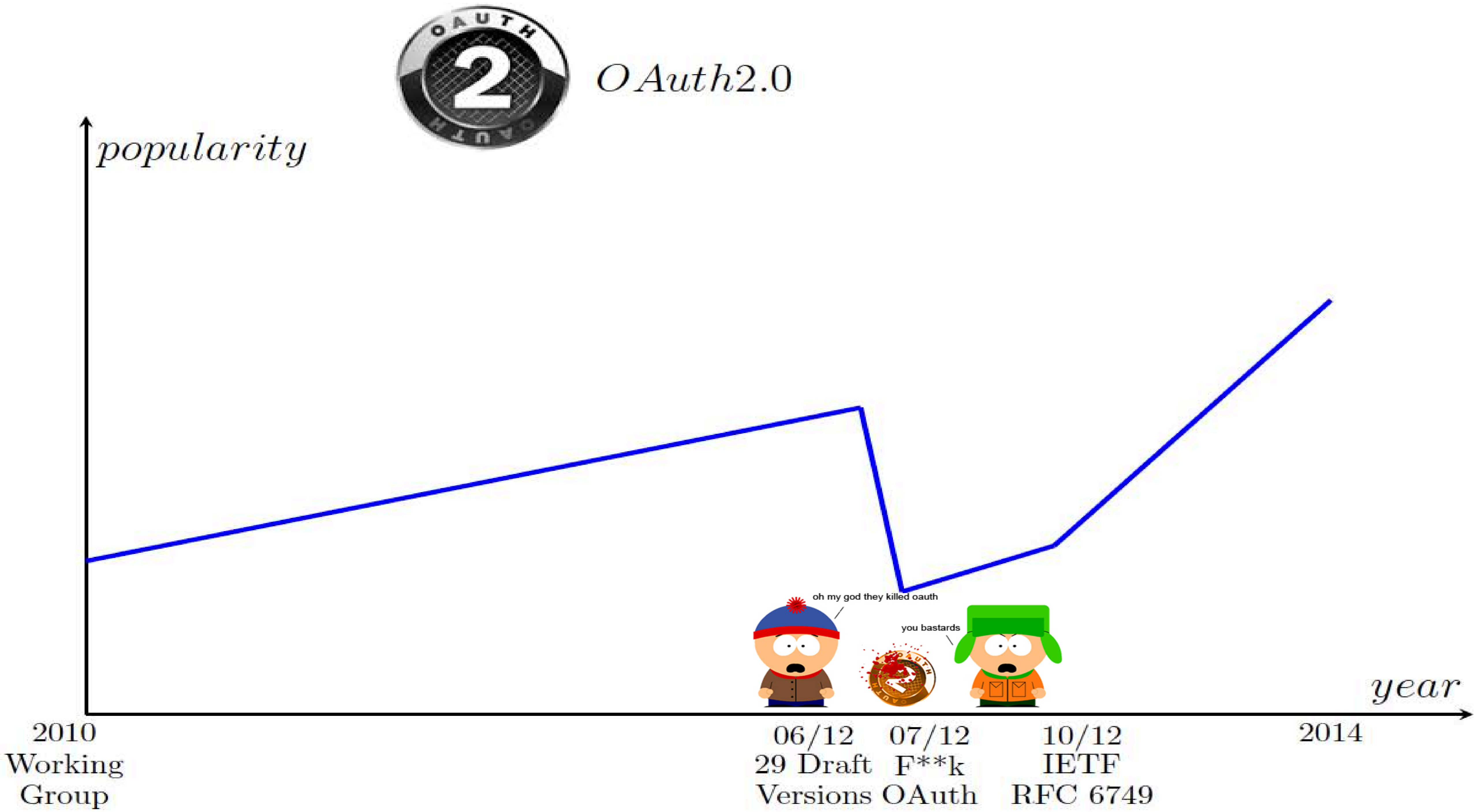
Choose how you communicate with friends. See how it works or manage imported contacts.

Skype Name:

Skype Password:

Facebook won't store your password.

A bit of history - OAuth 2.0



OAuth flows

- Authorization Code Grant (aka server side flow)

  ✓

- Implicit Grant (aka Client side flow)

- Resource Owner Password Credentials Grant

- Client Credentials Grant

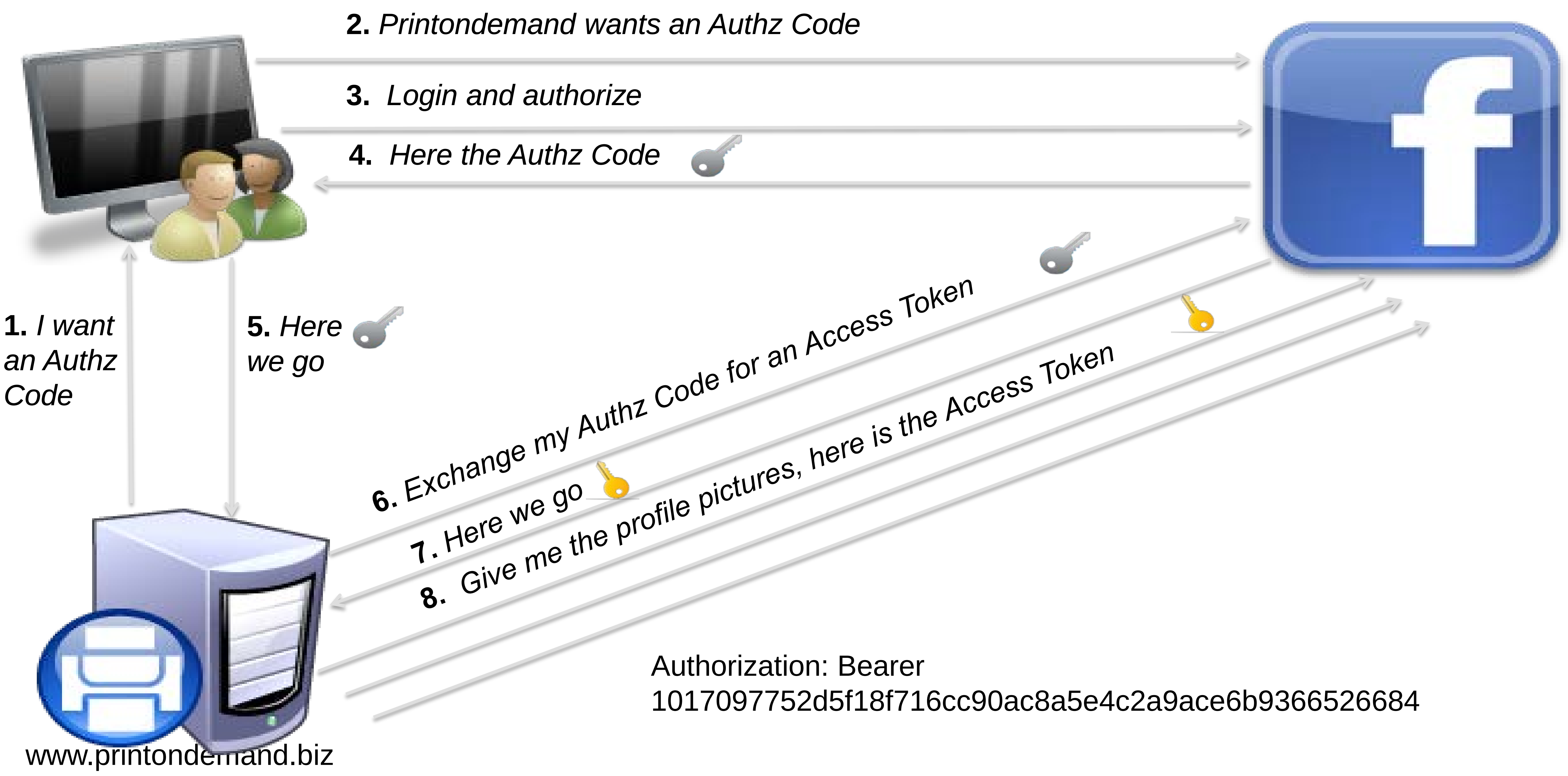
OAuth Actors

- Resource Owner (Alice)

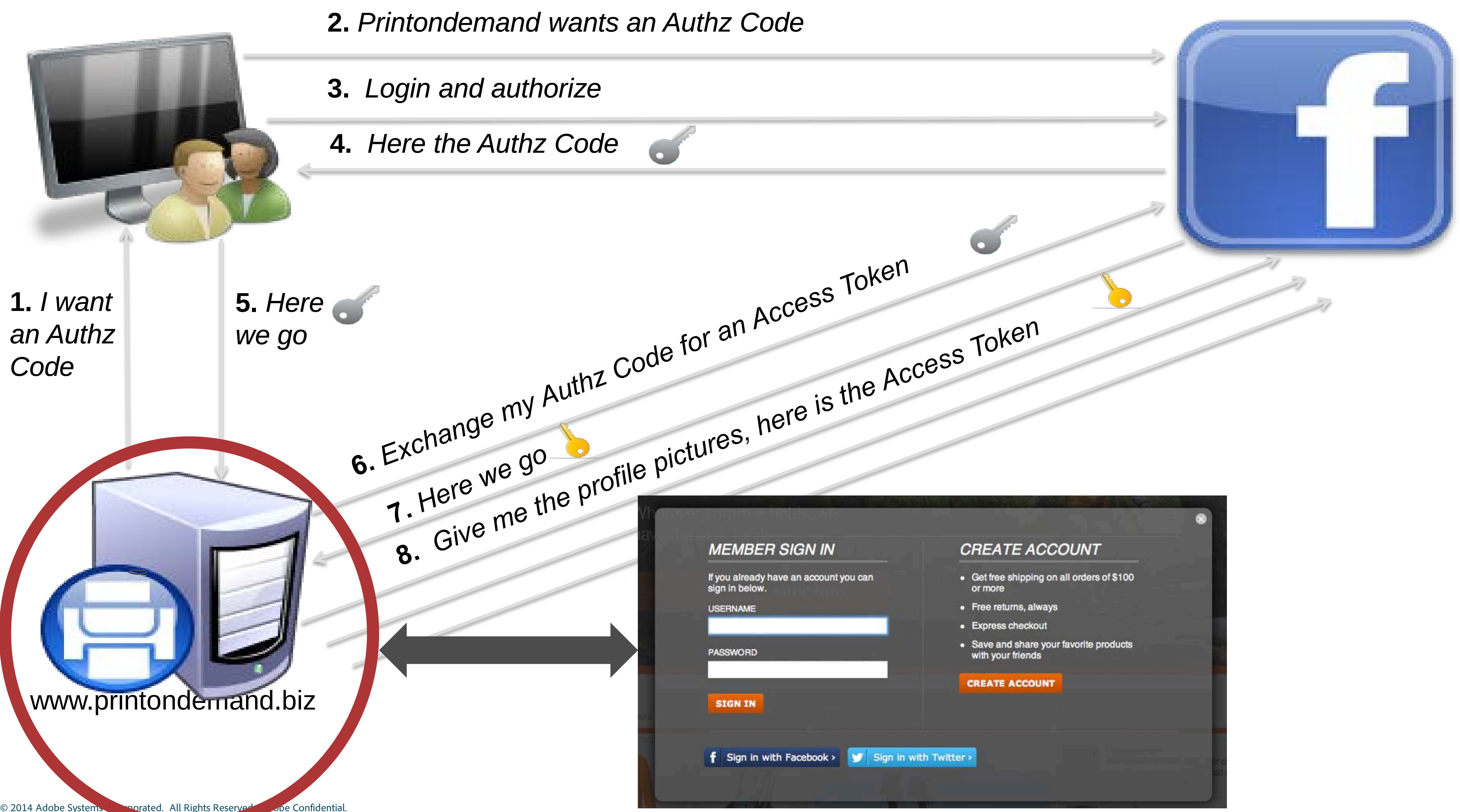
- Client (Bob, worker at www.printondemand.

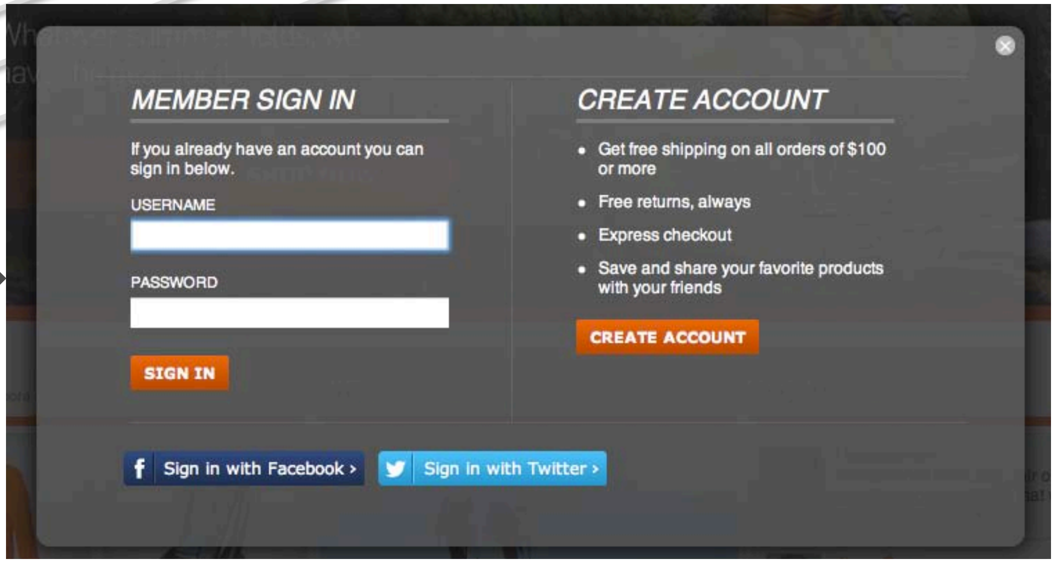
- Server (Carol from Facebook

Traditional OAuth "dance" - Authorization Code Grant aka server side flow


2. *Printondemand wants an Authz Code*
3. *Login and authorize*
4. *Here the Authz Code*
1. *I want an Authz Code*
5. *Here we go*
6. Exchange my Authz Code for an Access Token
7. Here we go
8. Give me the profile pictures, here is the Access Token
Authorization: Bearer 1017097752d5f18f716cc90ac8a5e4c2a9ace6b9366526684
www.printondemand.biz

Traditional OAuth "dance" - Authorization Code Grant aka server side flow


2. Printondemand wants an Authz Code
3. Login and authorize
4. Here the Authz Code
1. I want an Authz Code
5. Here we go
6. Exchange my Authz Code for an Access Token
7. Here we go
8. Give me the profile pictures, here is the Access Token
www.printondemand.biz


MEMBER SIGN IN

If you already have an account you can sign in below.

USERNAME

PASSWORD

SIGN IN

CREATE ACCOUNT

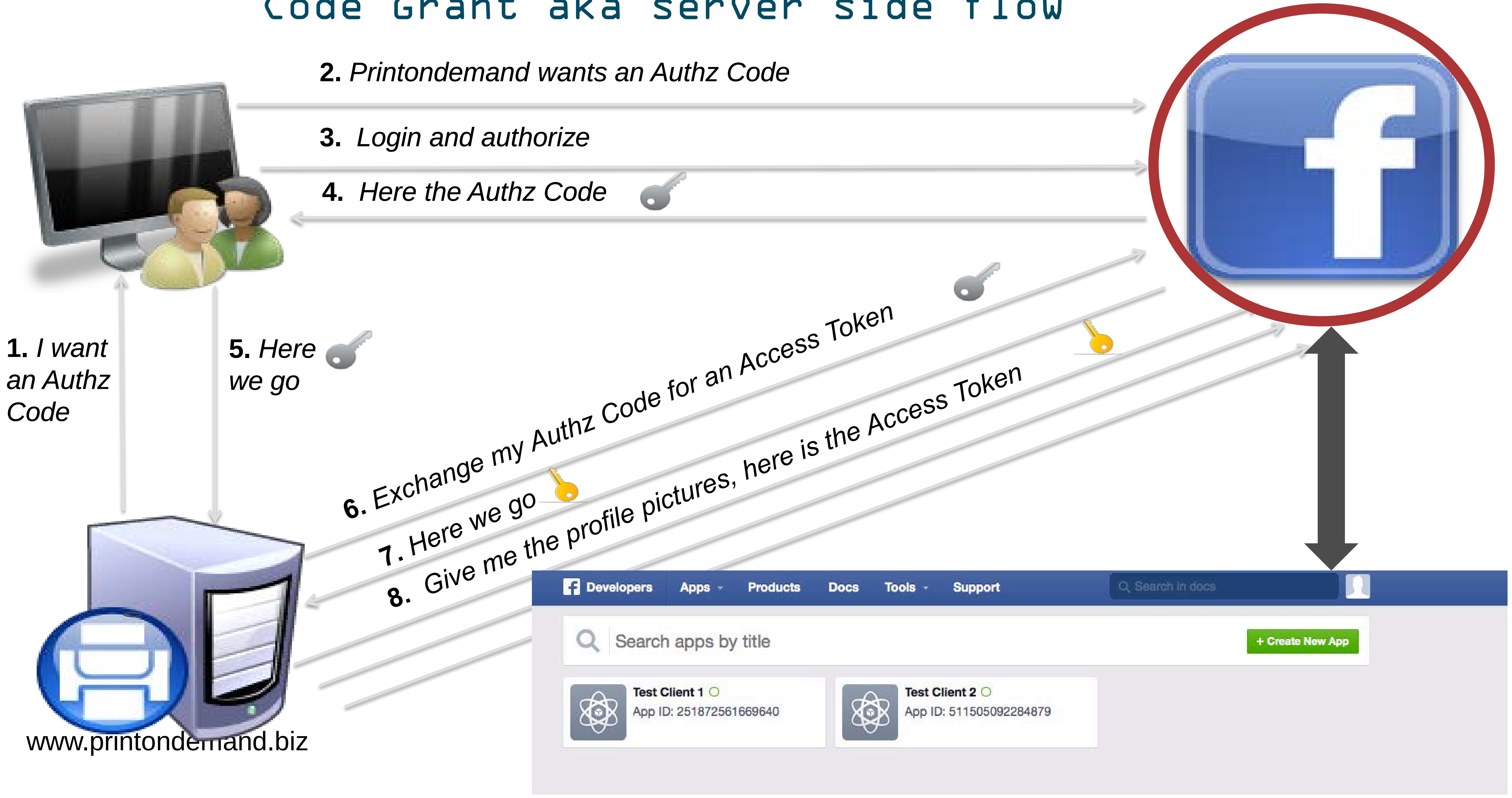
- Get free shipping on all orders of $100 or more
- Free returns, always
- Express checkout
- Save and share your favorite products with your friends

CREATE ACCOUNT

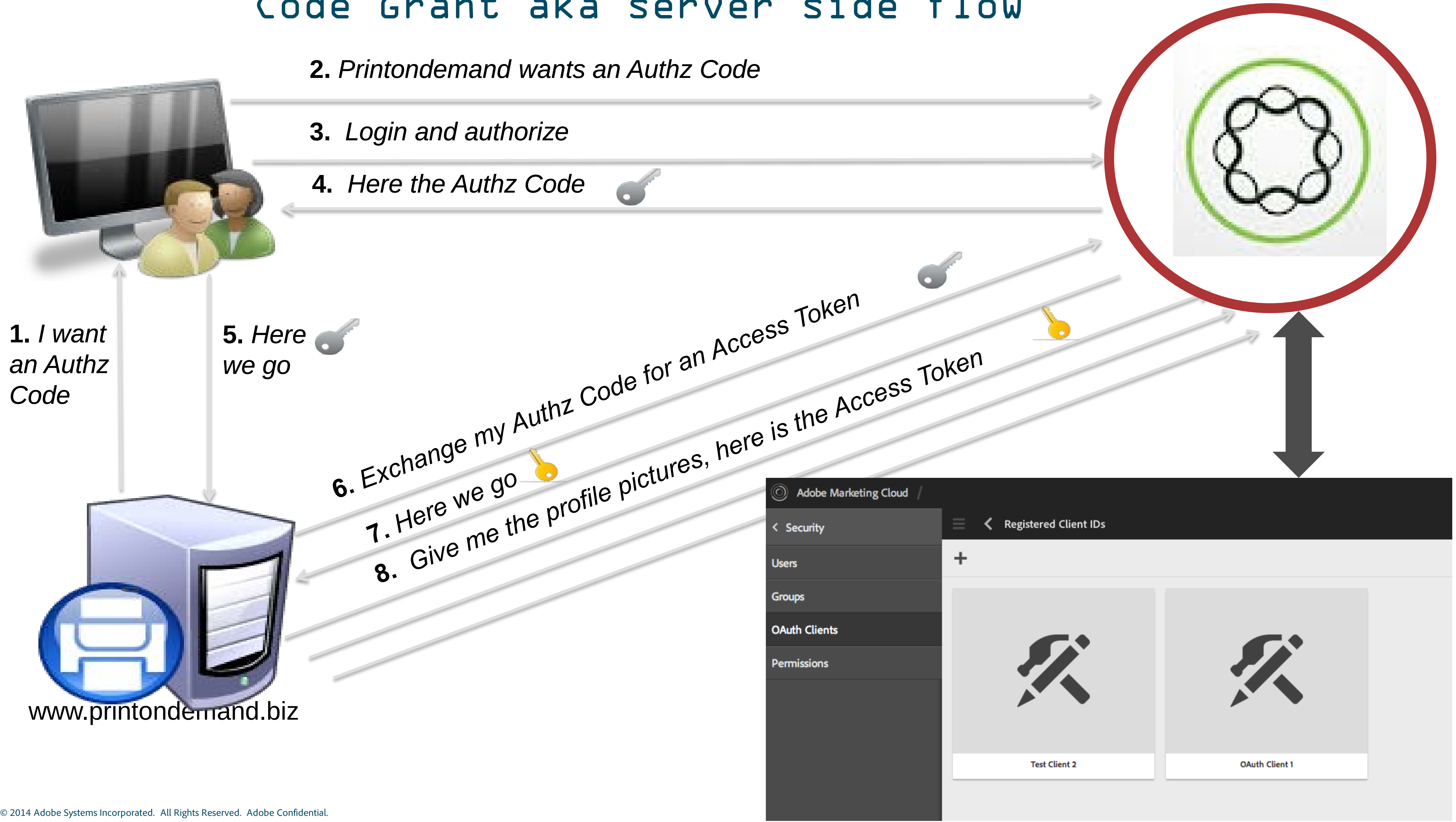
Sign in with Facebook ›

Sign in with Twitter ›

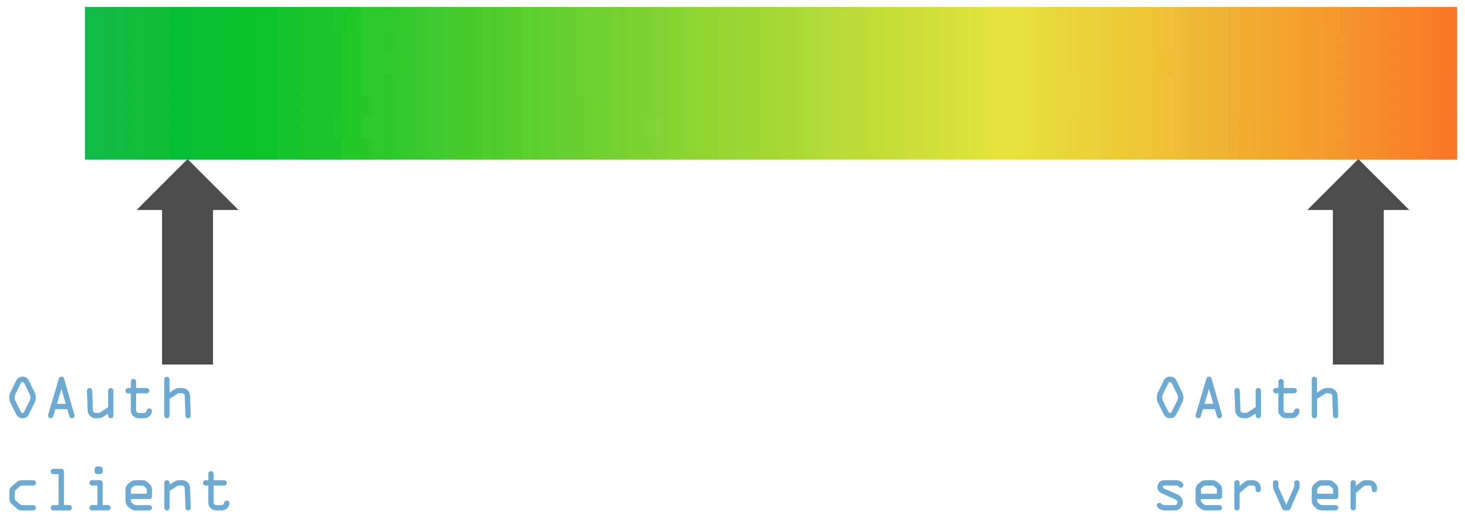
Traditional OAuth "dance" - Authorization Code Grant aka server side flow



Traditional OAuth "dance" - Authorization Code Grant aka server side flow



How difficult is to implement OAuth ?



Bearer Token

```
Authorization: Bearer
1017097752d5f18f716cc90ac
8a5e4c2a9ace6b9366526684
```

Scalable OAuth server

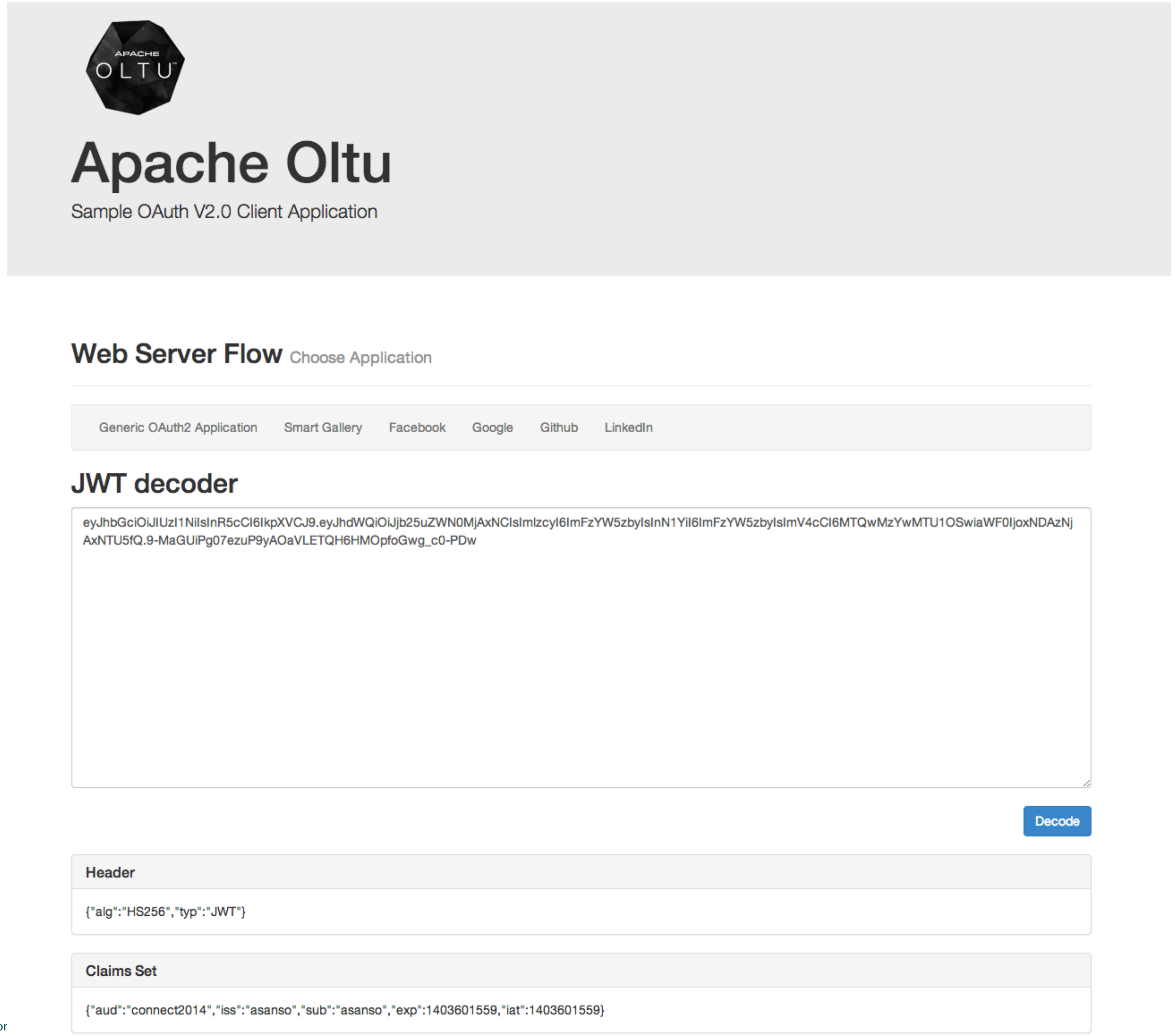
- derive encryption key using $salt_1$
- derive mac key using $salt_2$
- generate random iv
- encrypt, then mac($salt_1$ + iv + data)
- transmit $salt_1$, $salt_2$ iv and encrypted

JSON Web Token

```
eyJhbGciOiJIUzI1
NiIsInR5cCI6IkpX
VCJ9.
eyJhdWQiOiJjb25u
ZWN0MjAxNCIsImlz
cyI6ImFzYW5zbyIs
InN1YiI6ImFzYW5z
byIsImV4cCI6MTQw
MzYwMTU1OSwiaWF0
IjoxNDAzNjAxNTU5
fQ.MaGUiPg07ezuP
9yAOaVLETQHbHMOp
foGwg_c0-PDw
```

Header	{"alg":"HS256","typ":"JWT"}
Claims	{"aud":"connect2014","iss":"asanso","sub":"asanso","exp":1403601559,"iat":1403601559}
Signature	HMAC

JSON Web Token



Apache Oltu

Sample OAuth V2.0 Client Application

Web Server Flow Choose Application

JWT decoder

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdWQiOiJjb25uZWN0MjAxNCIsImlzcyI6ImFzYW5zbyIsInN1YiI6ImFzYW5zbyIsImV4cCI6MTQwMzYwMTU1OSwiaWF0IjoxNDAzNj
AxNTU5fQ.9-MaGUiPg07ezuP9yAOaVLETQH6HMOpfoGwg_c0-PDw
```

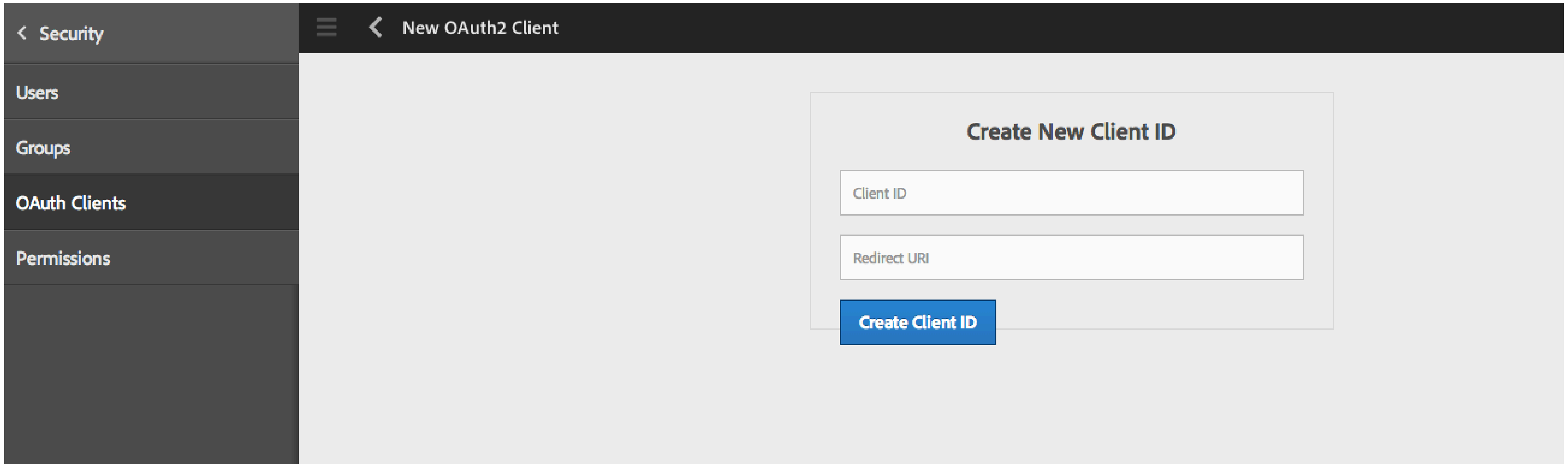
Header

```
{"alg":"HS256","typ":"JWT"}
```

Claims Set

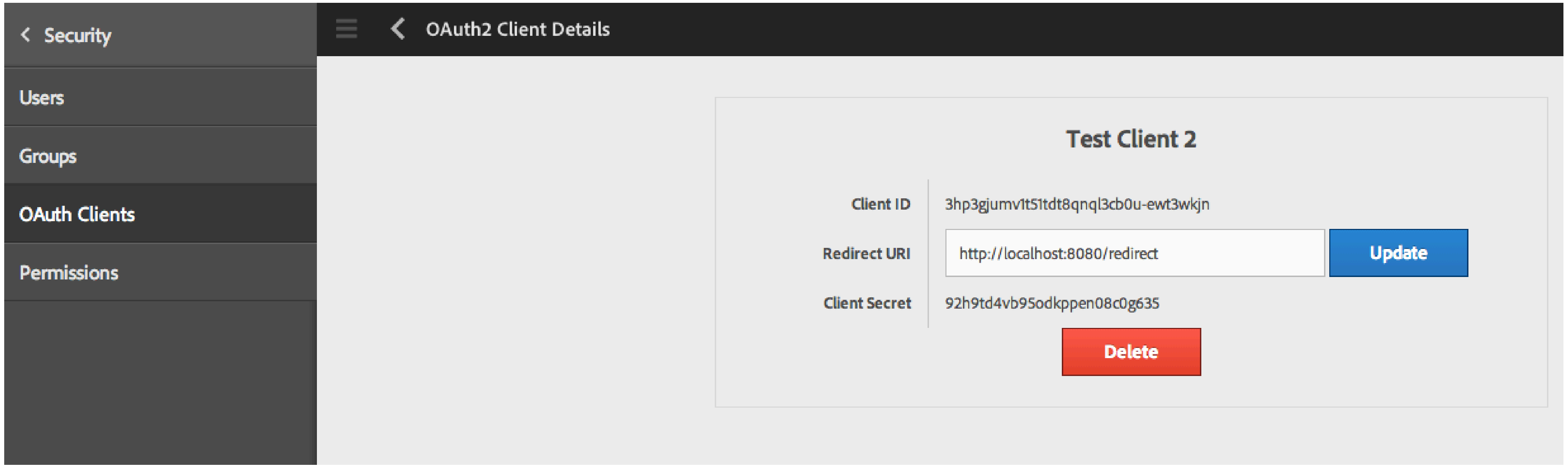
```
{"aud":"connect2014","iss":"asanso","sub":"asanso","exp":1403601559,"iat":1403601559}
```

AEM: register an OAuth client



http://<hostname>:<port>/libs/granite/oauth/content/newclient.html

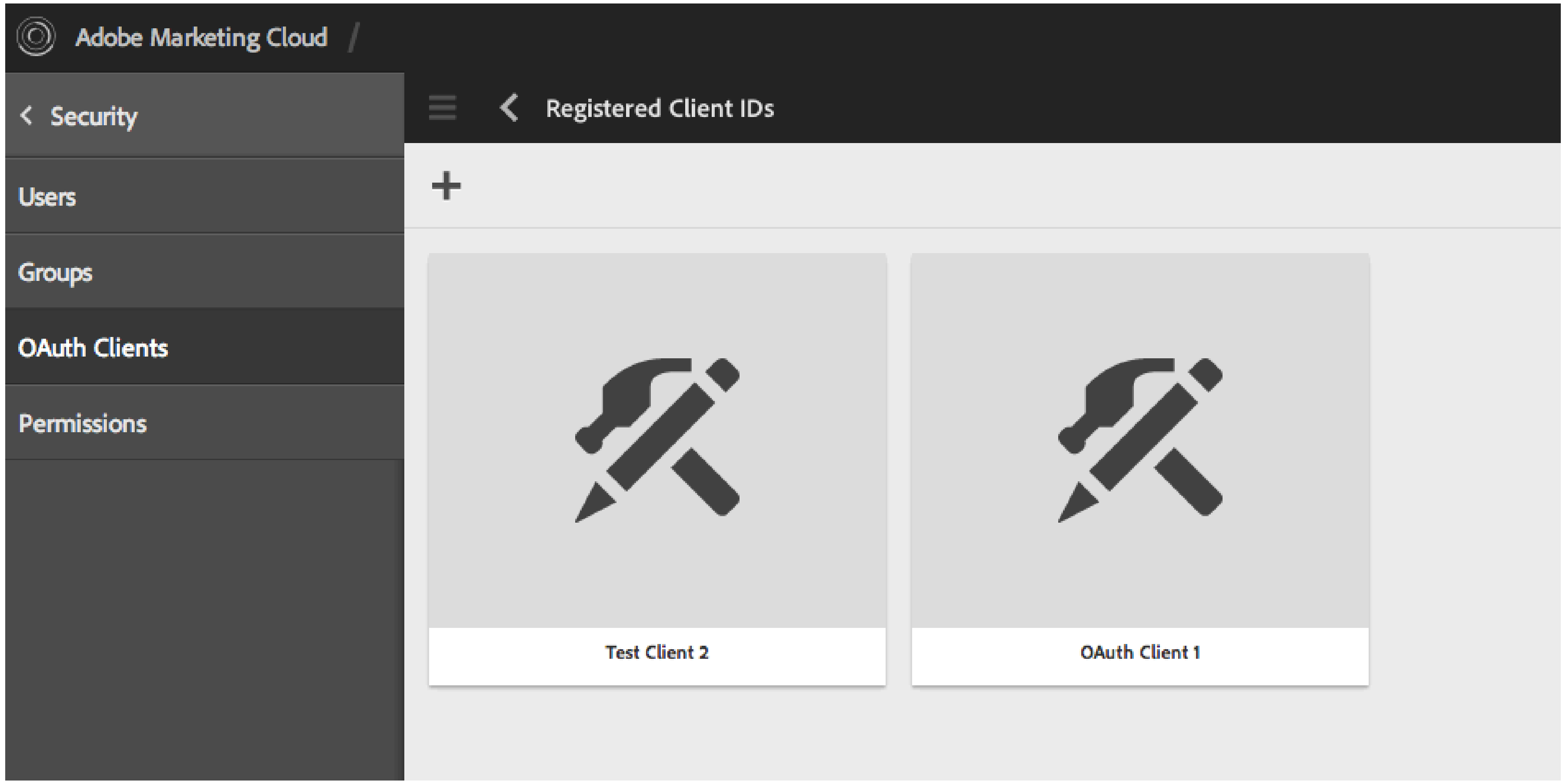
AEM: edit an OAuth client



Test Client 2

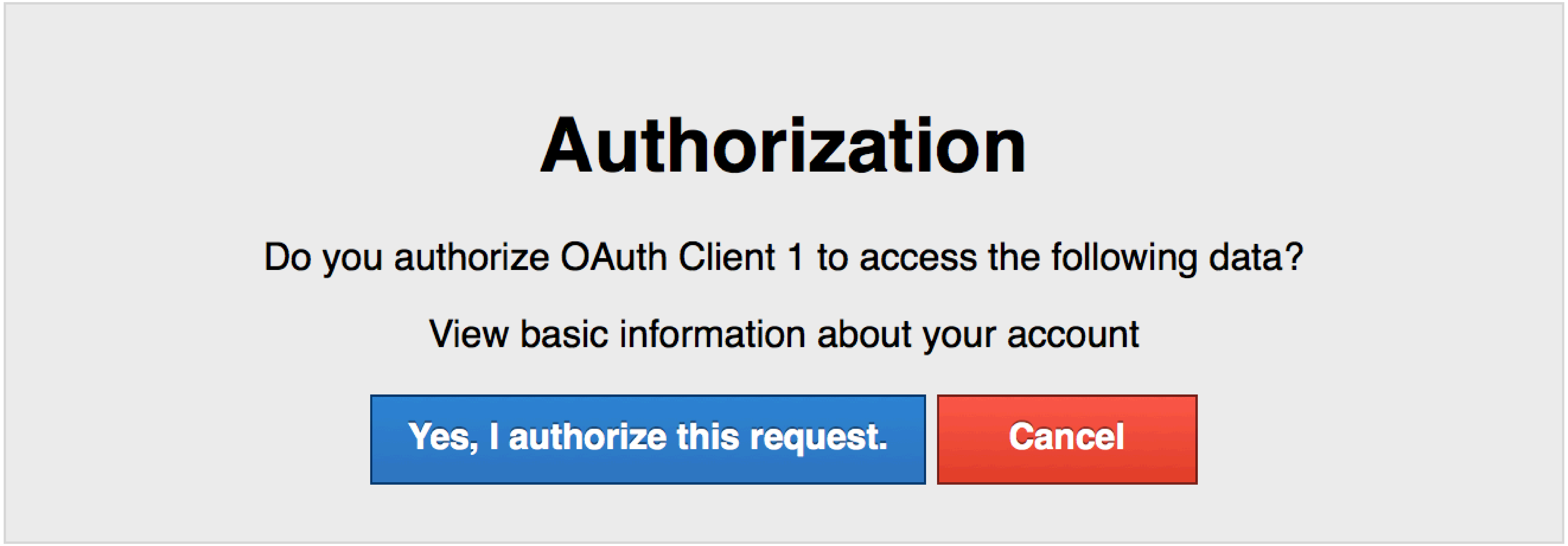
Client ID	3hp3gjumv1t51tdt8qnql3cb0u-ewt3wkjn
Redirect URI	http://localhost:8080/redirect
Client Secret	92h9td4vb95odkppen08c0g635

AEM: registered OAuth clients



http://<hostname>:<port>/libs/granite/oauth/content/clients.html

AEM: OAuth consent screen



Authorization

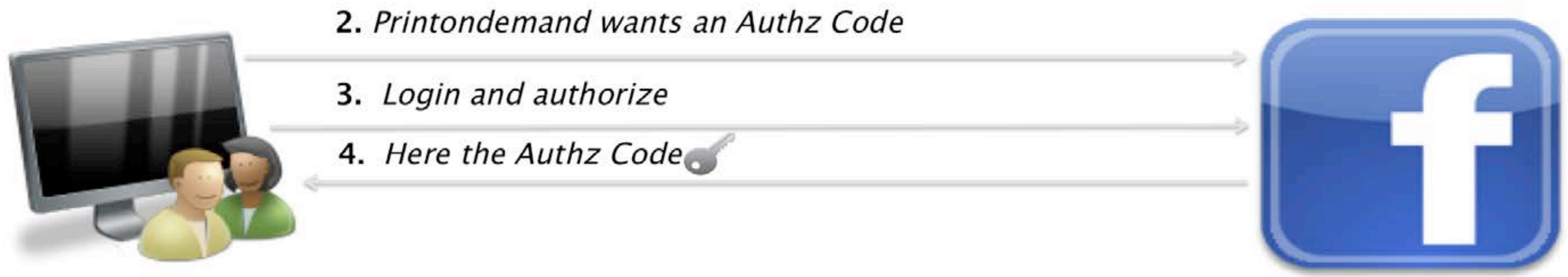
Do you authorize OAuth Client 1 to access the following data?

View basic information about your account

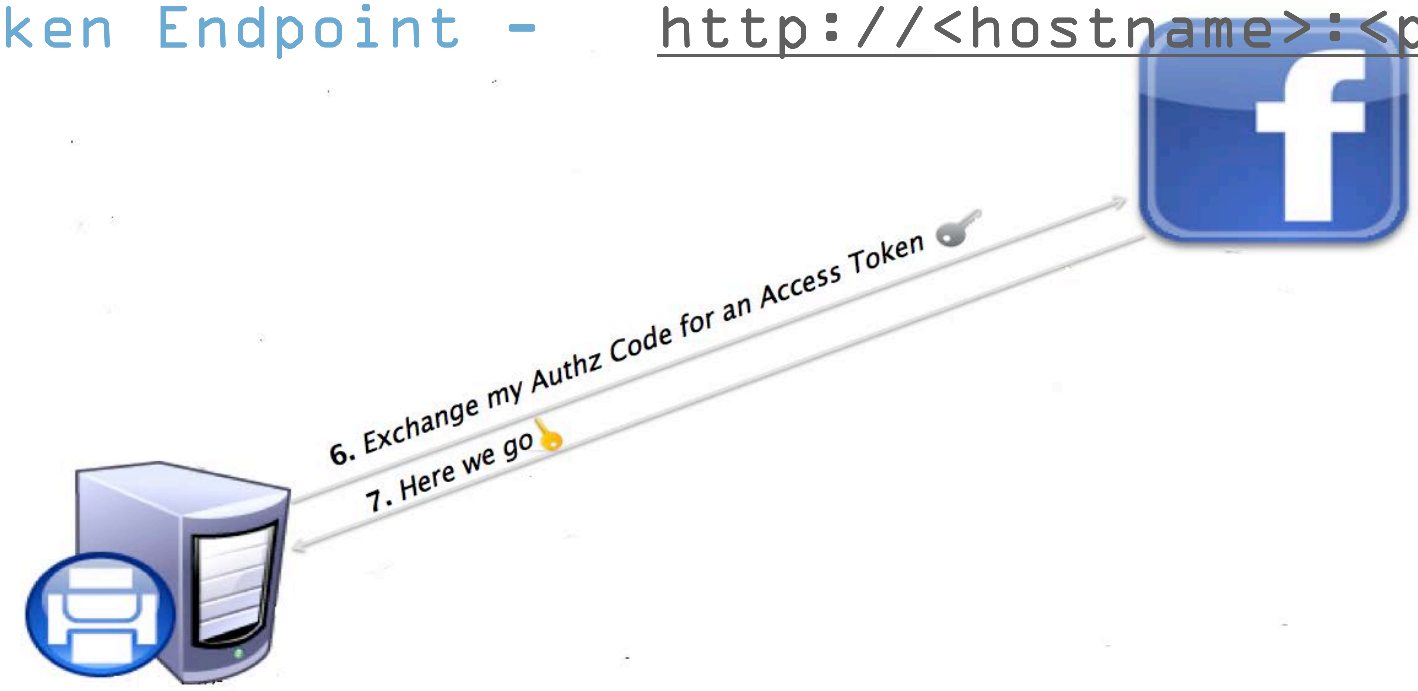
AEM: OAuth Endpoint

- Authorization Endpoint -

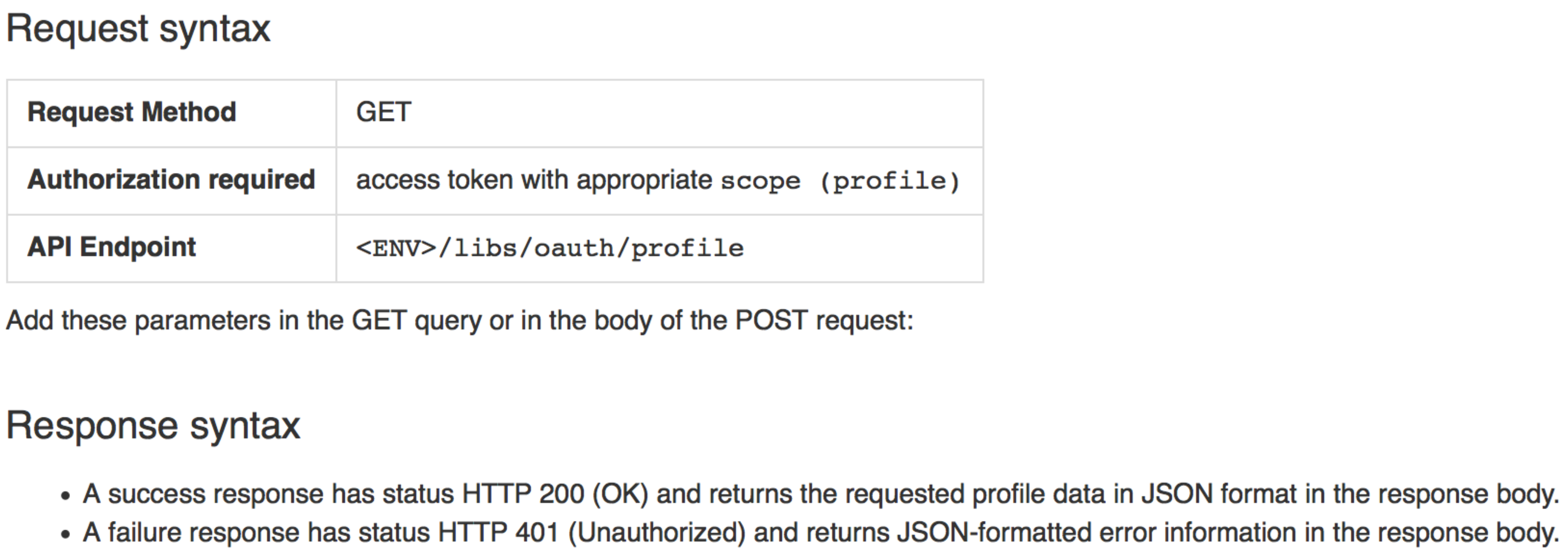
  ht



- Token Endpoint - http://<hostname>:<port>/oauth/token



AEM: OAuth APIs – Profile API



Request syntax

Request Method	GET
Authorization required	access token with appropriate `scope (profile)`
API Endpoint	`<ENV>/libs/oauth/profile`

Add these parameters in the GET query or in the body of the POST request:

Response syntax

- A success response has status HTTP 200 (OK) and returns the requested profile data in JSON format in the response body.
- A failure response has status HTTP 401 (Unauthorized) and returns JSON-formatted error information in the response body.

AEM: OAuth APIs – Profile API

http://<hostname>:<port>/oauth/authorize?response_type=code&client_id=<client_id>&redirect_uri=<redirect_uri>&**scope=profile**

GET Request

```
GET /libs/oauth/profile HTTP/1.1
Authorization: Bearer %ACCESS_TOKEN%
Host: localhost:4502
```

Curl example

```
curl -H "Authorization: Bearer  %ACCESS_TOKEN%" http://localhost:4502/libs/oauth/profile HTTP/1.1
```

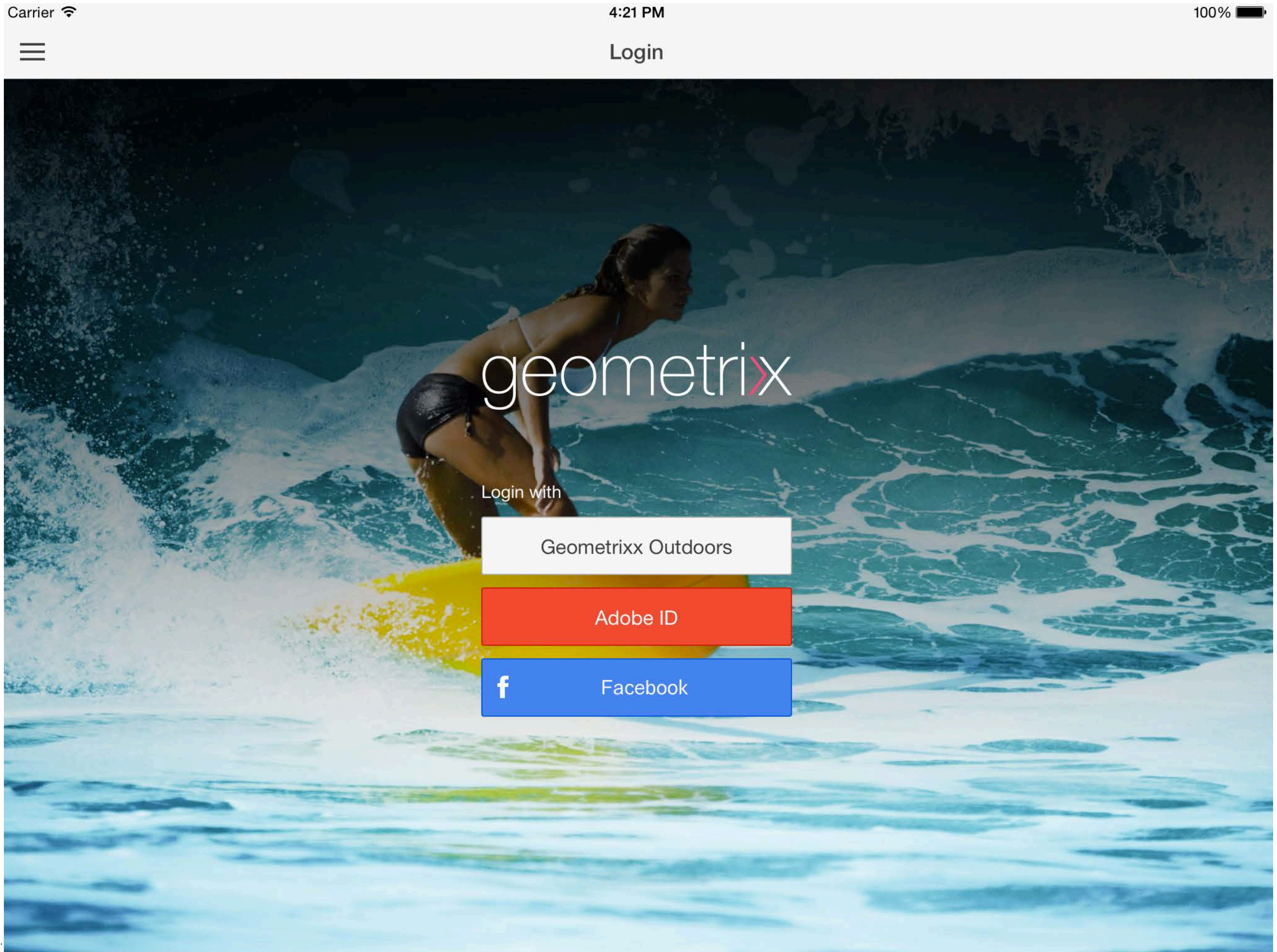
Success response

```
HTTP/1.1 200 OK
Date: Thu, 06 Mar 2014 13:56:03 GMT
Content-Type: application/json;charset=utf-8
Content-Length: 238
Server: Jetty(8.y.z-SNAPSHOT)
{"path":"/home/users/j/jdoe/profile","user":{"authorizableId":"jdoe","home":"/home/users/j/jdoe"},"familyName":"Doe","country":"United States","givenName":"John"}
```

Failure Response

```
HTTP/1.1 401 Unauthorized
Date: Thu, 06 Mar 2014 14:15:45 GMT
WWW-Authenticate: Bearer realm="Sling (Development)"
Server: Jetty(8.y.z-SNAPSHOT)
```

AEM: Profile API usage - Authent

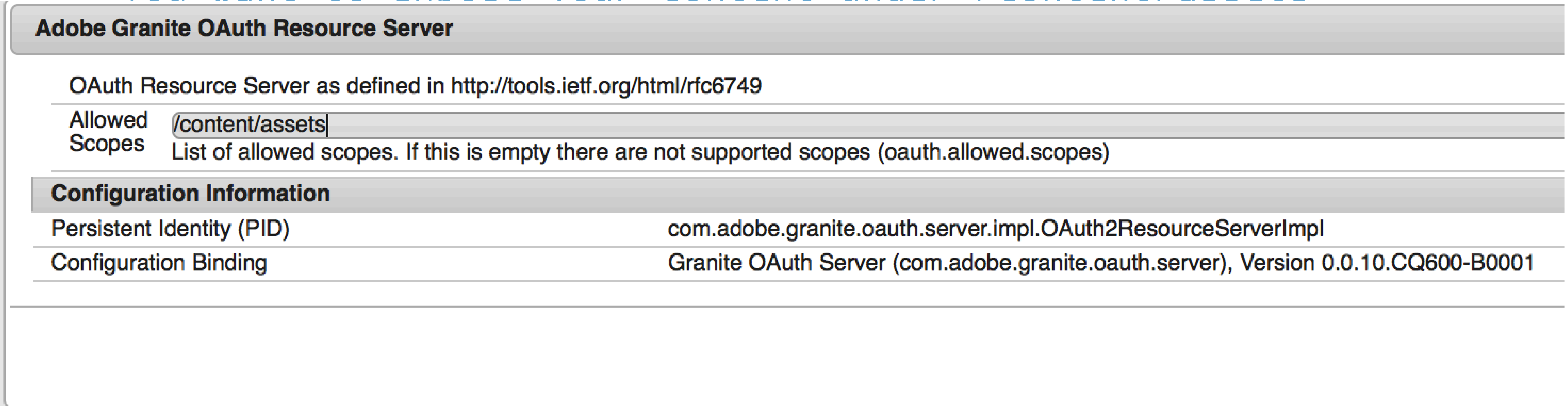


AEM: OAuth APIs - Extended

Assume you have an API with an endpoint /content/assets

OR

You want to expose your content under /content/assets



Adobe Granite OAuth Resource Server

OAuth Resource Server as defined in http://tools.ietf.org/html/rfc6749

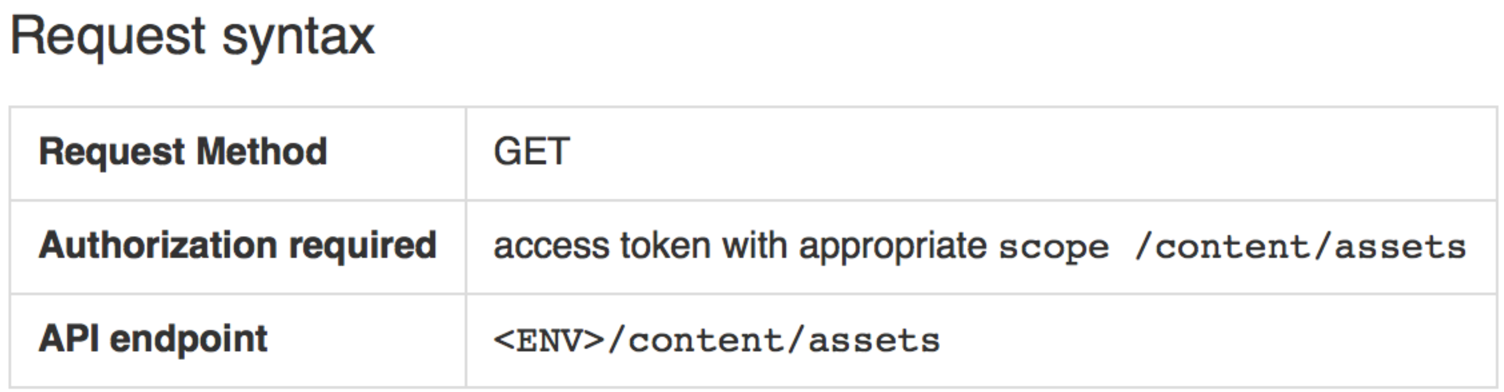
Allowed Scopes	/content/assets List of allowed scopes. If this is empty there are not supported scopes (oauth.allowed.scopes)

Configuration Information

Persistent Identity (PID)	com.adobe.granite.oauth.server.impl.OAuth2ResourceServerImpl
Configuration Binding	Granite OAuth Server (com.adobe.granite.oauth.server), Version 0.0.10.CQ600-B0001

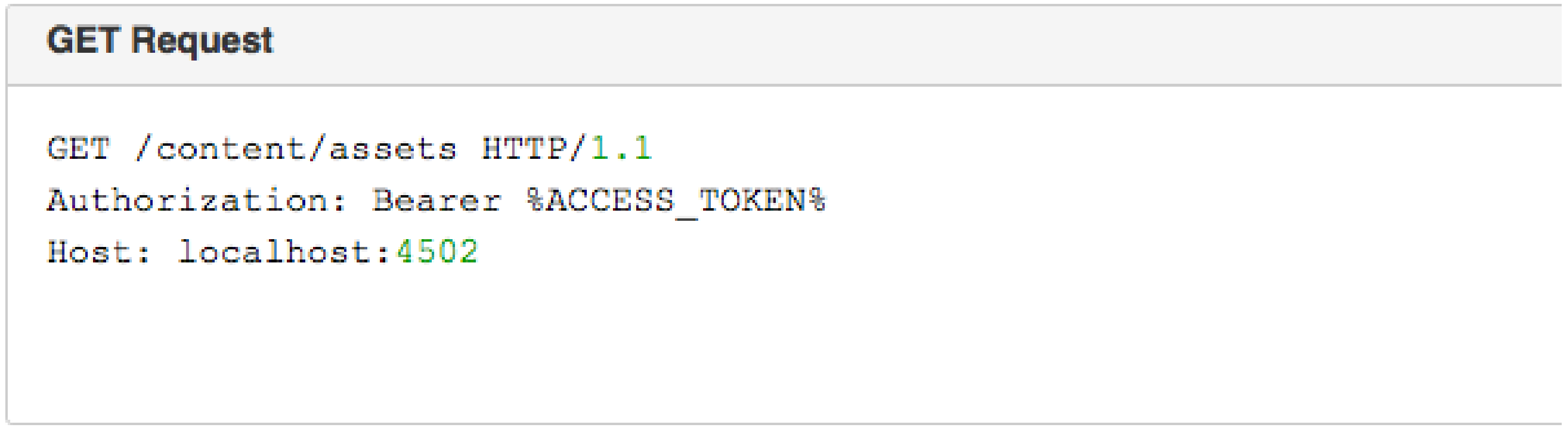
AEM: OAuth APIs – Extended

http://<hostname>:<port>/oauth/authorize?response_type=code&client_id=<client_id>&redirect_uri=<redirect_uri>&**scope=/content/assets**



Request syntax

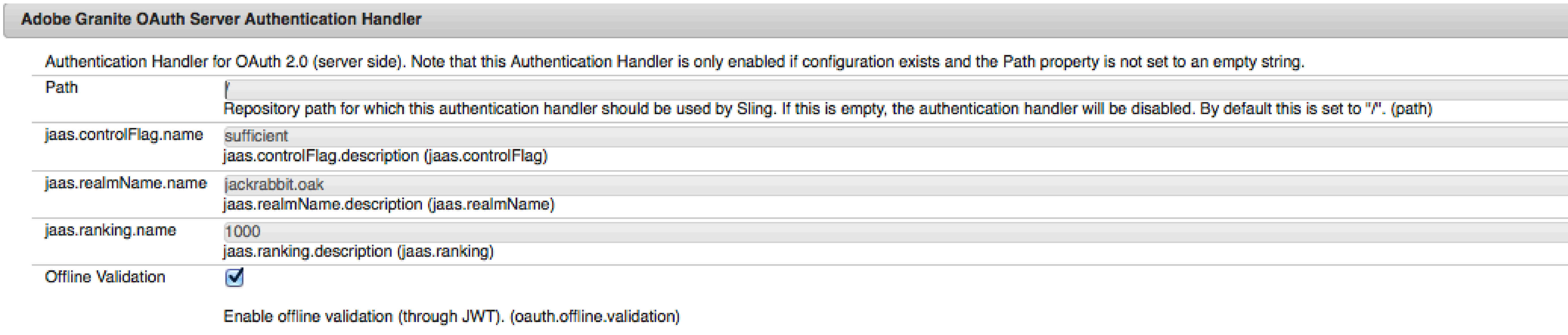
Request Method	GET
Authorization required	access token with appropriate `scope` `/content/assets`
API endpoint	`<ENV>/content/assets`



GET Request

```
GET /content/assets HTTP/1.1
Authorization: Bearer %ACCESS_TOKEN%
Host: localhost:4502
```

Enable OAuth Server in AEM

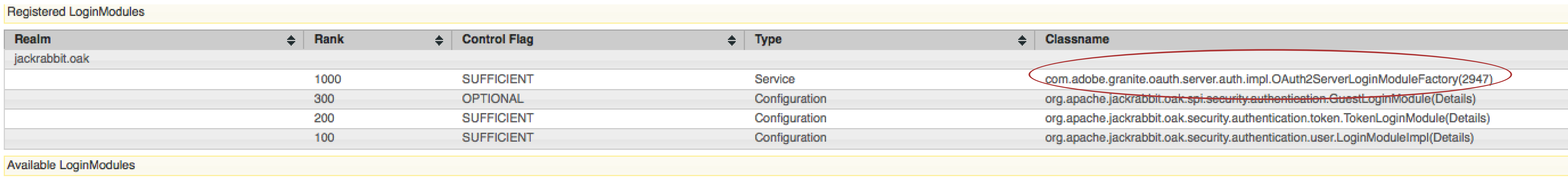


Adobe Granite OAuth Server Authentication Handler

Authentication Handler for OAuth 2.0 (server side). Note that this Authentication Handler is only enabled if configuration exists and the Path property is not set to an empty string.

Path	/ Repository path for which this authentication handler should be used by Sling. If this is empty, the authentication handler will be disabled. By default this is set to "/". (path)
jaas.controlFlag.name	sufficient jaas.controlFlag.description (jaas.controlFlag)
jaas.realmName.name	jackrabbit.oak jaas.realmName.description (jaas.realmName)
jaas.ranking.name	1000 jaas.ranking.description (jaas.ranking)
Offline Validation	✓ Enable offline validation (through JWT). (oauth.offline.validation)

http://<hostname>:<port>/system/console/jaas



Registered LoginModules

Realm	Rank	Control Flag	Type	Classname
jackrabbit.oak				
	1000	SUFFICIENT	Service	com.adobe.granite.oauth.server.auth.impl.OAuth2ServerLoginModuleFactory(2947)
	300	OPTIONAL	Configuration	org.apache.~~jackrabbit.oak.spi.security.authentication.GuestLoginModule~~(Details)
	200	SUFFICIENT	Configuration	org.apache.jackrabbit.oak.security.authentication.token.TokenLoginModule(Details)
	100	SUFFICIENT	Configuration	org.apache.jackrabbit.oak.security.authentication.user.LoginModuleImpl(Details)

Available LoginModules

Authentication in AEM

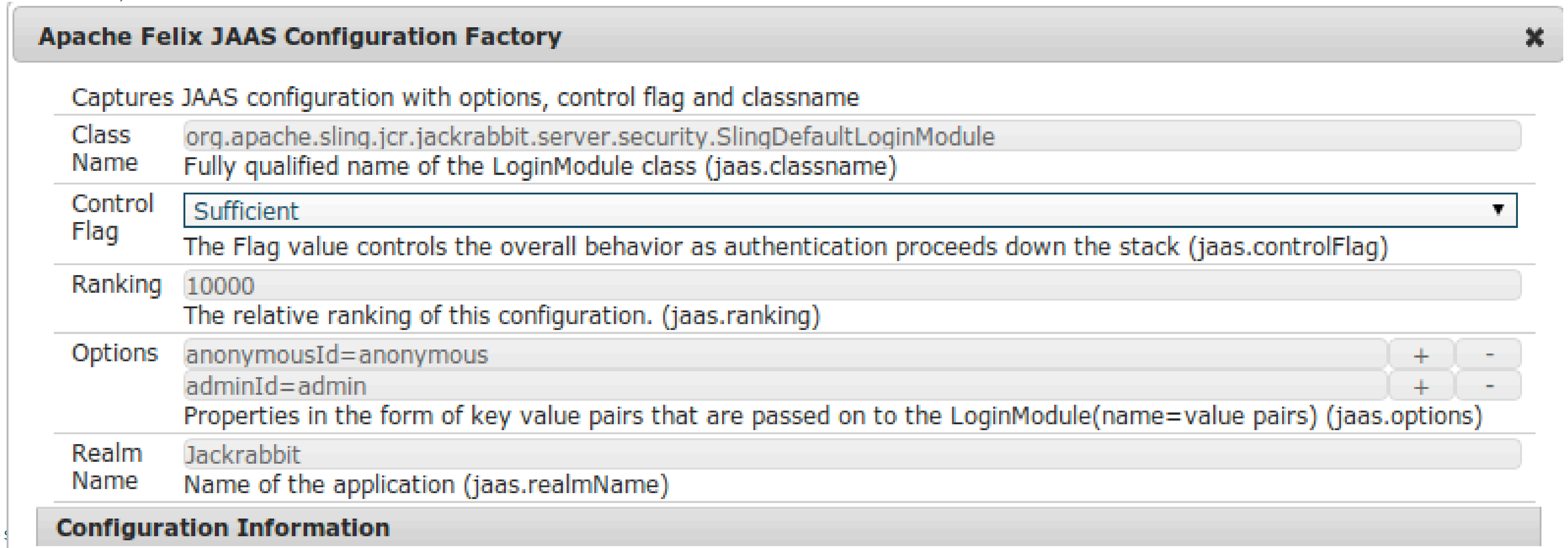
1. The client sends request with username and password
2. SlingAuthenticator calls the AuthenticationHandler (the CQ default is TokenAuthenticationHandler)
3. The AuthenticationHandler returns AuthenticationInfo with username and password
4. SlingAuthenticator calls RepositoryFactory with AuthenticationInfo to get resource resolver and validate the credentials (JackRabbit/Oak LoginModule)
5. SlingAuthenticator calls AuthenticationFeedbackHandler#authenticationSucceeded which may set cookies
6. request continues to be processed (or is redirected)

Extend Authentication in AEM

- Trusted Credential DEPRECATED
- com.day.crx.security.token.TokenUtil#createCredentials
- Custom (companion) LoginModule

⬅ Pre-AEM 6.0 Fragment bundle for the Login Module

➡ Post-AEM 6.0 Native JAAS-OSGi integration



Apache Felix JAAS Configuration Factory

Captures JAAS configuration with options, control flag and classname

Class Name	org.apache.sling.jcr.jackrabbit.server.security.SlingDefaultLoginModule Fully qualified name of the LoginModule class (jaas.classname)
Control Flag	Sufficient The Flag value controls the overall behavior as authentication proceeds down the stack (jaas.controlFlag)
Ranking	10000 The relative ranking of this configuration. (jaas.ranking)

Options	anonymousId=anonymous
	adminId=admin
	Properties in the form of key value pairs that are passed on to the LoginModule(name=value pairs) (jaas.options)

Realm Name	Jackrabbit Name of the application (jaas.realmName)

Configuration Information

server

Why?

Your application (**OAuth Client**) calls **OAuth Server** APIs on behalf of the service account, and user consent (**Resource Owner**) is not required (no human interaction).

How?

Register client

0. *Generate key pair and upload public key*

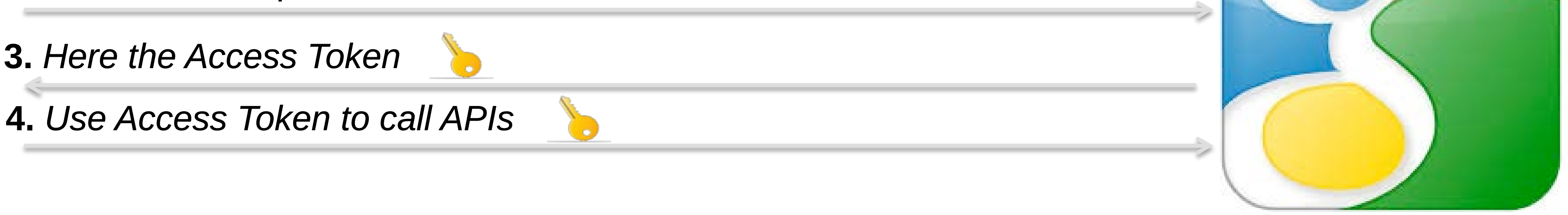
OAuth Server 2 Server Flow

1. *Create and sign JWT*

2. *Use JWT to request token*

3. *Here the Access Token*

4. *Use Access Token to call APIs*



server

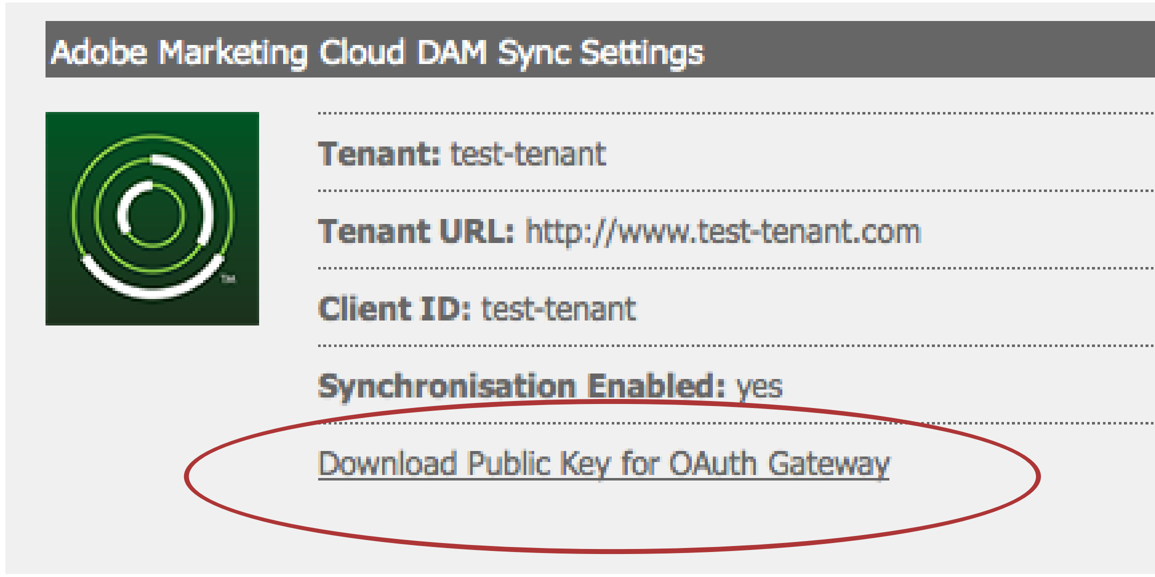
Why?

Your application (**OAuth Client**) calls **OAuth Server** APIs on behalf of the service account, and user consent (**Resource Owner**) is not required (no human interaction).

How?

Register client

0. *Generate key pair and upload public key*



Adobe Marketing Cloud DAM Sync Settings

Tenant: test-tenant

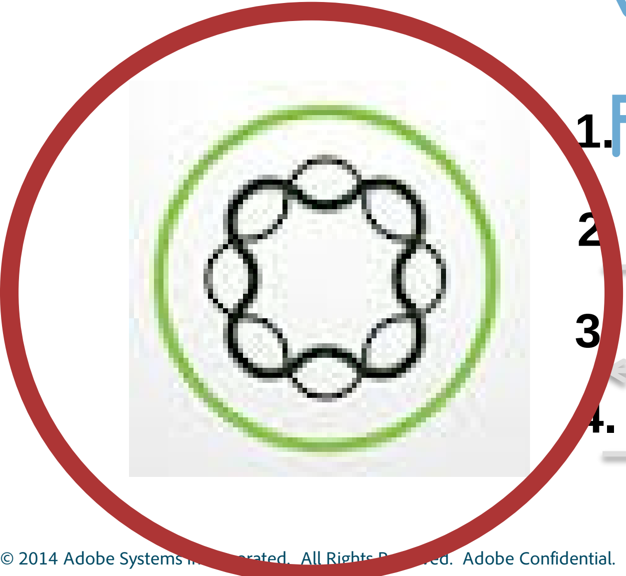
Tenant URL: http://www.test-tenant.com

Client ID: test-tenant

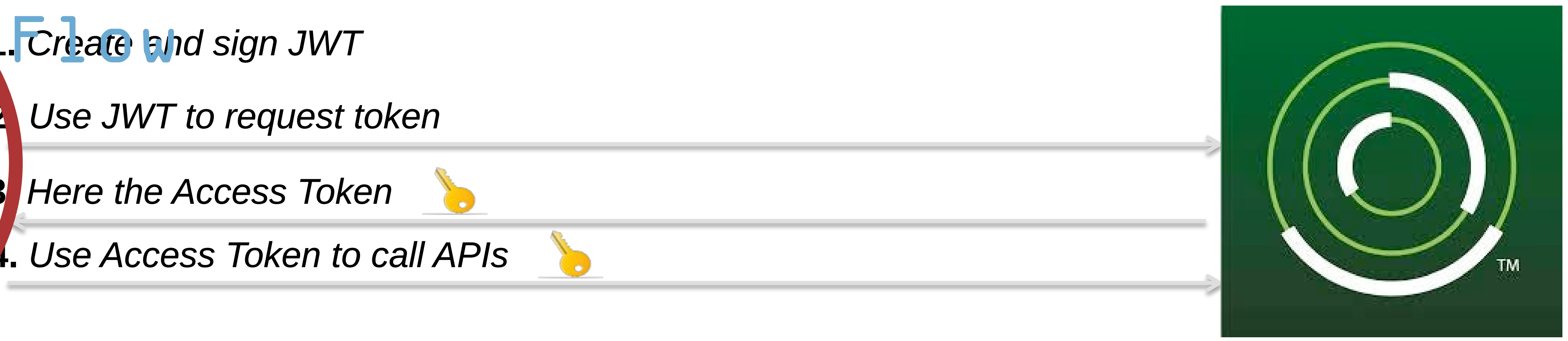
Synchronisation Enabled: yes

Download Public Key for OAuth Gateway

OAuth Server 2 Server Flow



1. *Create and sign JWT*



2. *Use JWT to request token*

3. *Here the Access Token*

4. *Use Access Token to call APIs*

(AEM)

0. *Generate key pair and upload public key* CryptoSupport

createKeyPair

```
KeyPair createKeyPair(String algorithm)
                      throws CryptoException
```

Generates a key pair. This will generate a new key pair every time it is called.

1. *Create and sign JWT* JwsBuilder

com.adobe.granite.oauth.jwt
Interface JwsBuilder

```
public interface JwsBuilder
```

The `JwsBuilder` provides a simple API to issue JWS formatted token as defined in http://tools.ietf.org/html/draft-ietf-oauth-json-web-token-16 and http://tools.ietf.org/html/draft-ietf-jose-json-web-signature-21

Method Summary	
`String`	`build()`
`JwsBuilder`	`setAudience(String aud)` Set the (Audience) Claim as for http://tools.ietf.org/html/draft-ietf-oauth-json-web-token-16#section-4.1.3
`JwsBuilder`	`setCustomClaimsSetField(String key, Object value)` Set a custom claim field
`JwsBuilder`	`setExpiresIn(long expiresIn)` Set the expiration time for the token expressed in seconds
`JwsBuilder`	`setIssuer(String iss)` Set the (Issuer) Claim as for http://tools.ietf.org/html/draft-ietf-oauth-json-web-token-16#section-4.1.1
`JwsBuilder`	`setScope(String scope)` Set the scope associate with the token
`JwsBuilder`	`setSubject(String sub)` Set the (Subject) Claim as for http://tools.ietf.org/html/draft-ietf-oauth-json-web-token-16#section-4.1.2

References

- OAuth 2.0 web site - http://oauth.net/2/
- OAuth 2.0 - http://tools.ietf.org/html/rfc6749
- Bearer Token - http://tools.ietf.org/html/rfc6750
- JWT - http://tools.ietf.org/html/draft-ietf-oauth-json-web-token-23
- http://docs.adobe.com/docs/en/aem/6-0/develop/ref/javadoc/com/adobe/granite/crypto/CryptoSupport.html
- http://docs.adobe.com/docs/en/aem/6-0/develop/ref/javadoc/com/adobe/granite/crypto/CryptoSupport.html
- Apache Oltu - http://oltu.apache.org/
- http://felix.apache.org/documentation/subprojects/apache-

Questions?