SPRING SUMMIT 2017

开放共享 原生共融

北京 | 2017年8月26日

Next Generation Oauth 2.0 Support With Spring Security 5.0

Joe Grandja
Spring Security Team

Github.com/jgrandja
@joe_grandja

Today's Objective

To provide a detailed overview
of the new **OAuth 2.0 Login** feature

Let's Define…

- OAuth 2.0 Authorization Framework
- OpenID Connect Core 1.0

Outline

1. OAuth 2.0 Overview
2. Requesting Authorization
3. Obtaining Access Token / ID Token
4. Accessing UserInfo Resource

1. OAuth 2.0 Overview

2. Requesting Authorization
3. Obtaining Access Token / ID Token
4. Accessing UserInfo Resource

Roles

- Authorization Server / OpenID Provider (OP)
- Resource Server
- Client / Relying Party (RP)
- Resource Owner

Client Types

- Confidential (server-based application)
- Public (native or browser-based application)

Authorization Grant Types

- Authorization Code
- Implicit
- Resource Owner Password Credentials
- Client Credentials

The “***authentication flow***”
is realized using the
Authorization Code Grant

Authorization Code Grant

- Redirection-based flow
- Targeted for Confidential Clients
- Returns **Access Token** and optional **Refresh Token**
- Returns **ID Token** for OpenID Connect flow

Protocol Endpoints

- Authorization Endpoint
- Token Endpoint
- Redirection Endpoint

OAuth 2.0 Overview
Recap

- 4x Roles
- Confidential vs. Public Clients
- 4x Authorization Grant Types
- *"Authentication Flow"* implemented using Authorization Code Grant
- Protocol Endpoints

1. OAuth 2.0 Overview

2. Requesting Authorization

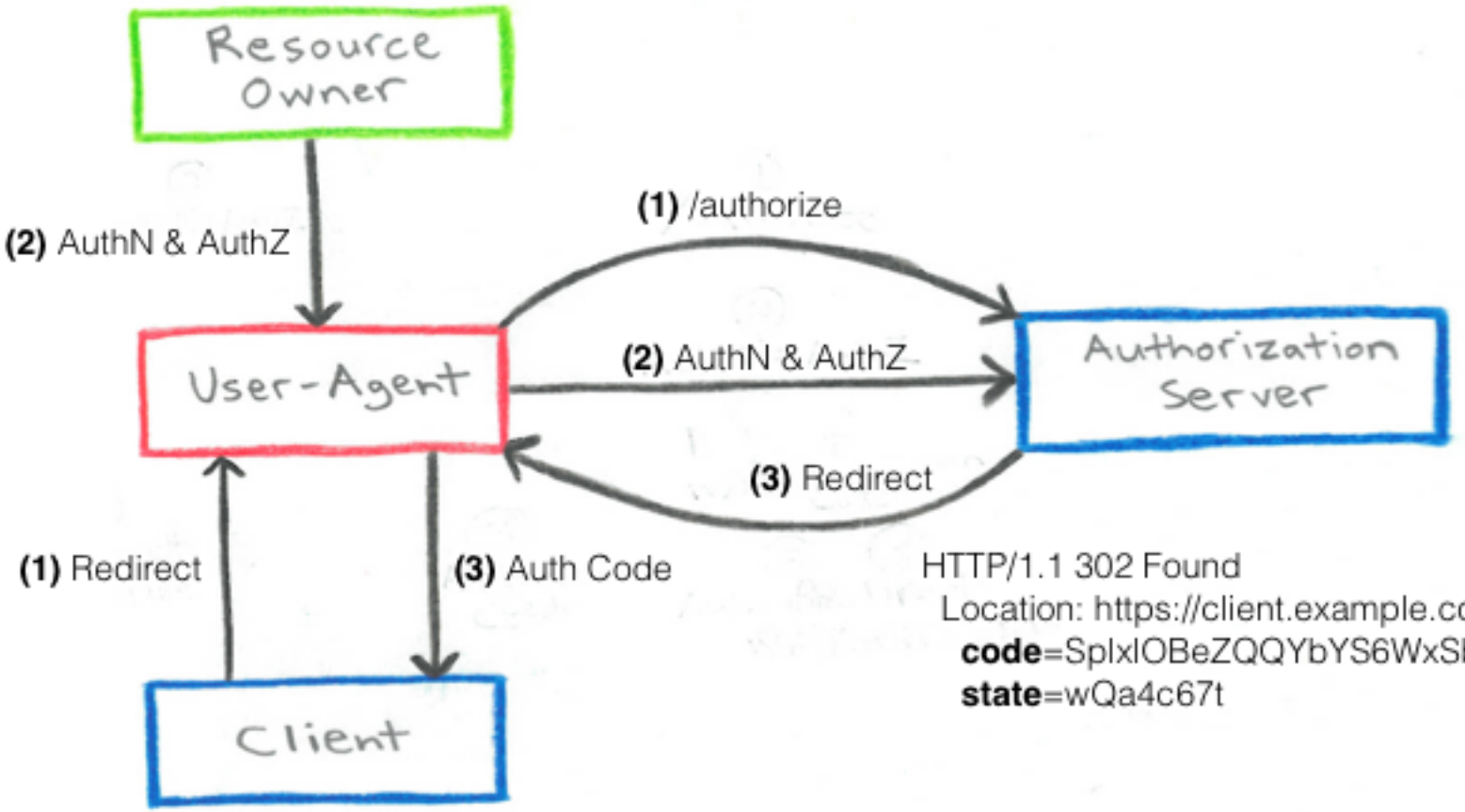
3. Obtaining Access Token / ID Token

4. Accessing UserInfo Resource

Requesting Authorization

Requesting Authorization

OAuth 2.0



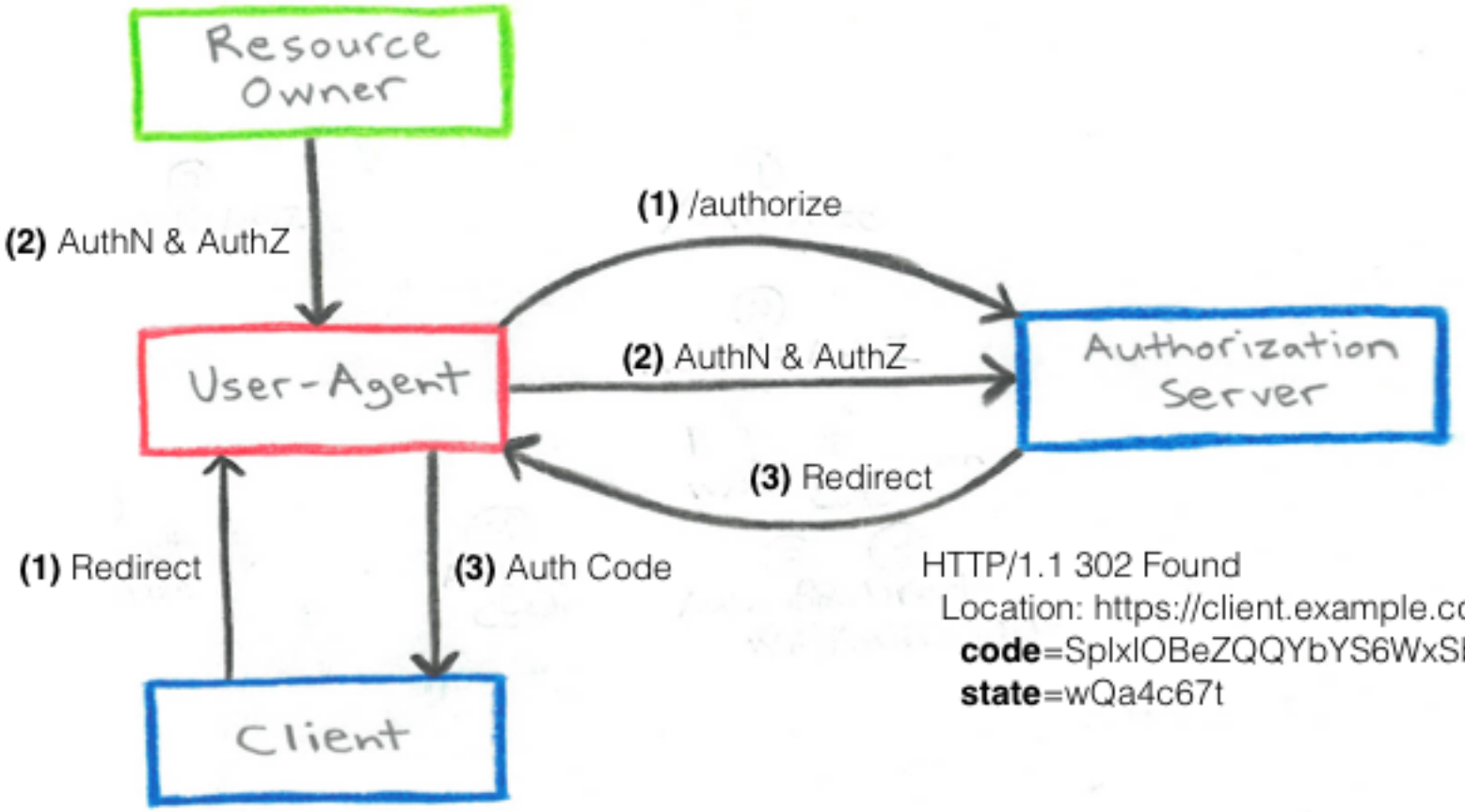
HTTP/1.1 302 Found
Location: https://client.example.com?
code=SplxlOBeZQQYbYS6WxSbIA&
state=wQa4c67t

HTTP/1.1 302 Found
Location: https://identity.example.com/authorize?
response_type=code&**scope**=profile email&
client_id=client1234&**state**=wQa4c67t&
redirect_uri=https://client.example.com

Requesting Authentication

Requesting Authentication

OpenID Connect 1.0



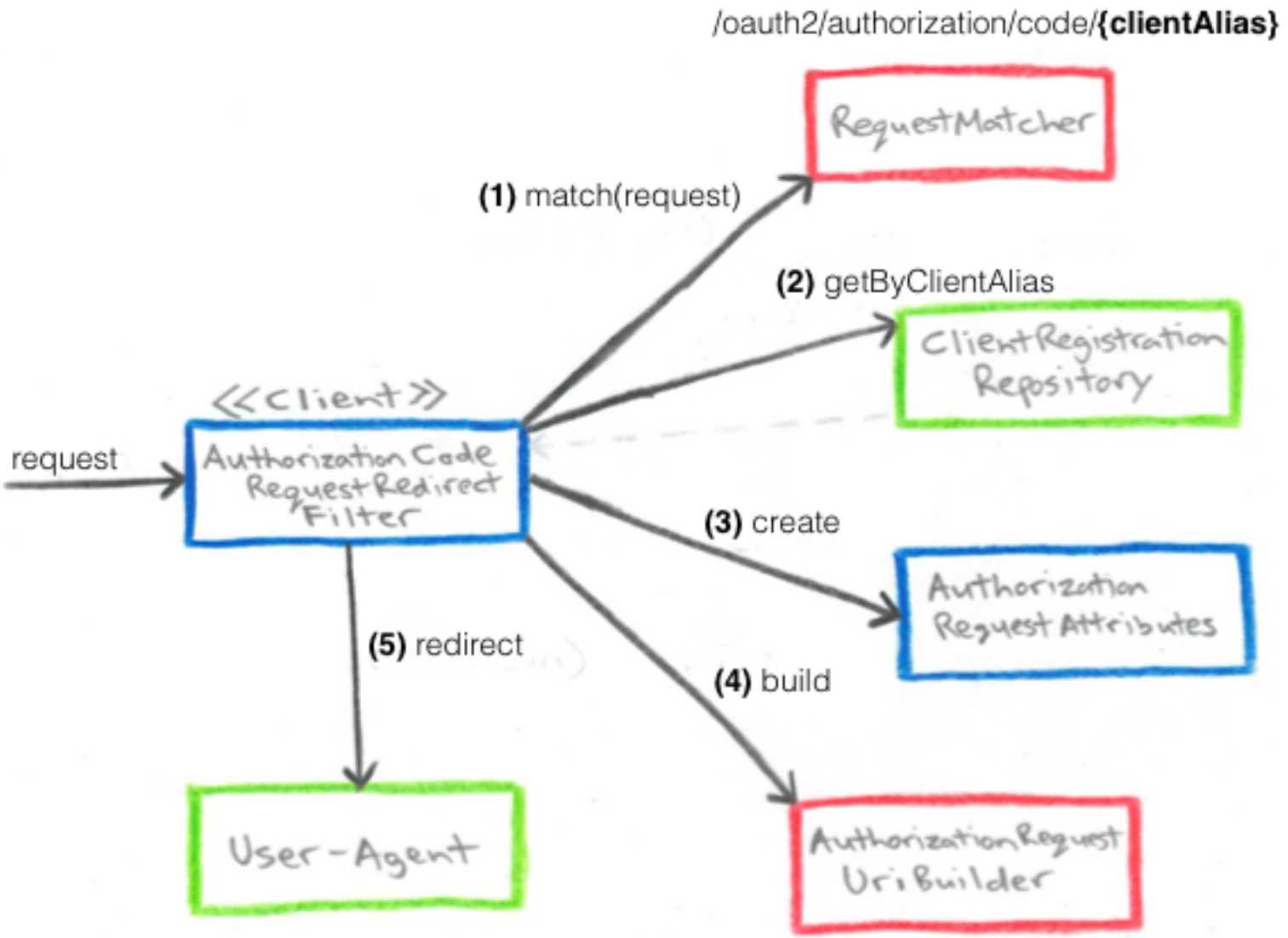
HTTP/1.1 302 Found
Location: https://client.example.com?
code=SplxlOBeZQQYbYS6WxSbIA&
state=wQa4c67t

HTTP/1.1 302 Found
Location: https://identity.example.com/authorize?
response_type=code&**scope**=**openid** profile email&
client_id=client1234&**state**=wQa4c67t&
redirect_uri=https://client.example.com

Requesting Authorization

Requesting Authorization

Implementation Artifacts



Requesting Authorization

Requesting Authorization
Recap

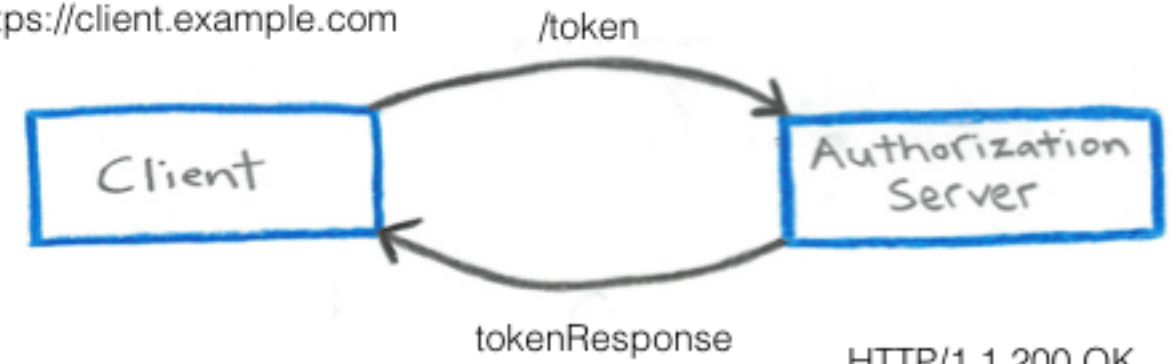
- Client redirects to Authorization Server
- End-User authorizes Client
- Authorization Server redirects back to Client with Authorization Code

1. OAuth 2.0 Overview
2. Requesting Authorization
3. **Obtaining Access Token**
4. Accessing the UserInfo Resource

Obtaining Access Token

Obtaining Access Token
OAuth 2.0

```
POST /token HTTP/1.1
  Host: identity.example.com
  Authorization: Basic czZCaGRSa3F0

  grant_type=authorization_code&
  code=SplxlOBeZQQYbYS6WxSbIA&
  redirect_uri=https://client.example.com
```

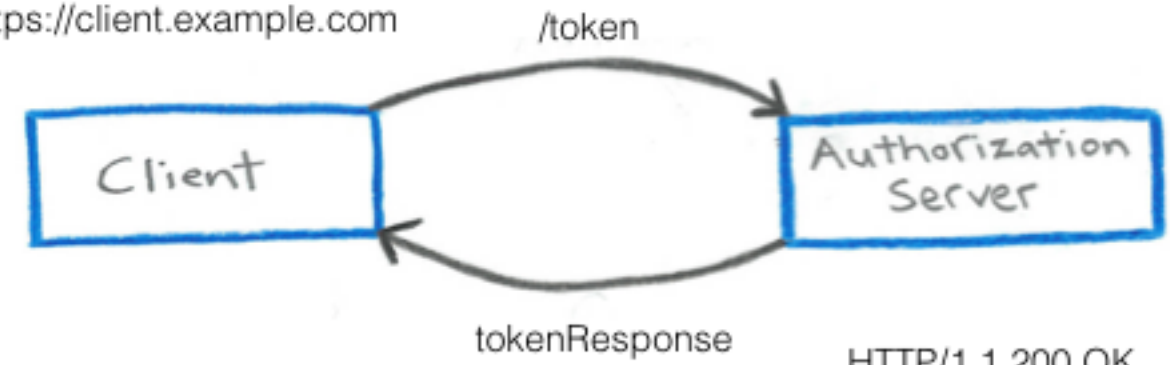



```
HTTP/1.1 200 OK
  Content-Type: application/json
  Cache-Control: no-store
  Pragma: no-cache

  {
   "access_token": "SlAV32hkKG",
   "token_type": "Bearer",
   "refresh_token": "8xLOxBtZp8",
   "expires_in": 3600
  }
```

Obtaining Access Token

Obtaining Access Token
OpenID Connect 1.0

```
POST /token HTTP/1.1
  Host: identity.example.com
  Authorization: Basic czZCaGRSa3F0

  grant_type=authorization_code&
  code=SplxlOBeZQQYbYS6WxSbIA&
  redirect_uri=https://client.example.com
```




```
HTTP/1.1 200 OK
  Content-Type: application/json
  Cache-Control: no-store
  Pragma: no-cache

  {
   "access_token": "SlAV32hkKG",
   "token_type": "Bearer",
   "refresh_token": "8xLOxBtZp8",
   "expires_in": 3600,
   "id_token": "eyJhbGciOiJSUzI1N"
  }
```

ID Token

ID Token

Contains ***"Claims"*** about the **Authentication of an End-User** by an Authorization Server


```
{
  "iss": "https://identity.example.com",
  "sub": "24400320",
  "aud": "client1234",
  "nonce": "n-0S6_WzA2Mj",
  "exp": 1311281970,
  "iat": 1311280970,
  "auth_time": 1311280969,
  "azp": "client1234"
}
```


Obtaining Access Token

Receiving the Authorization Code

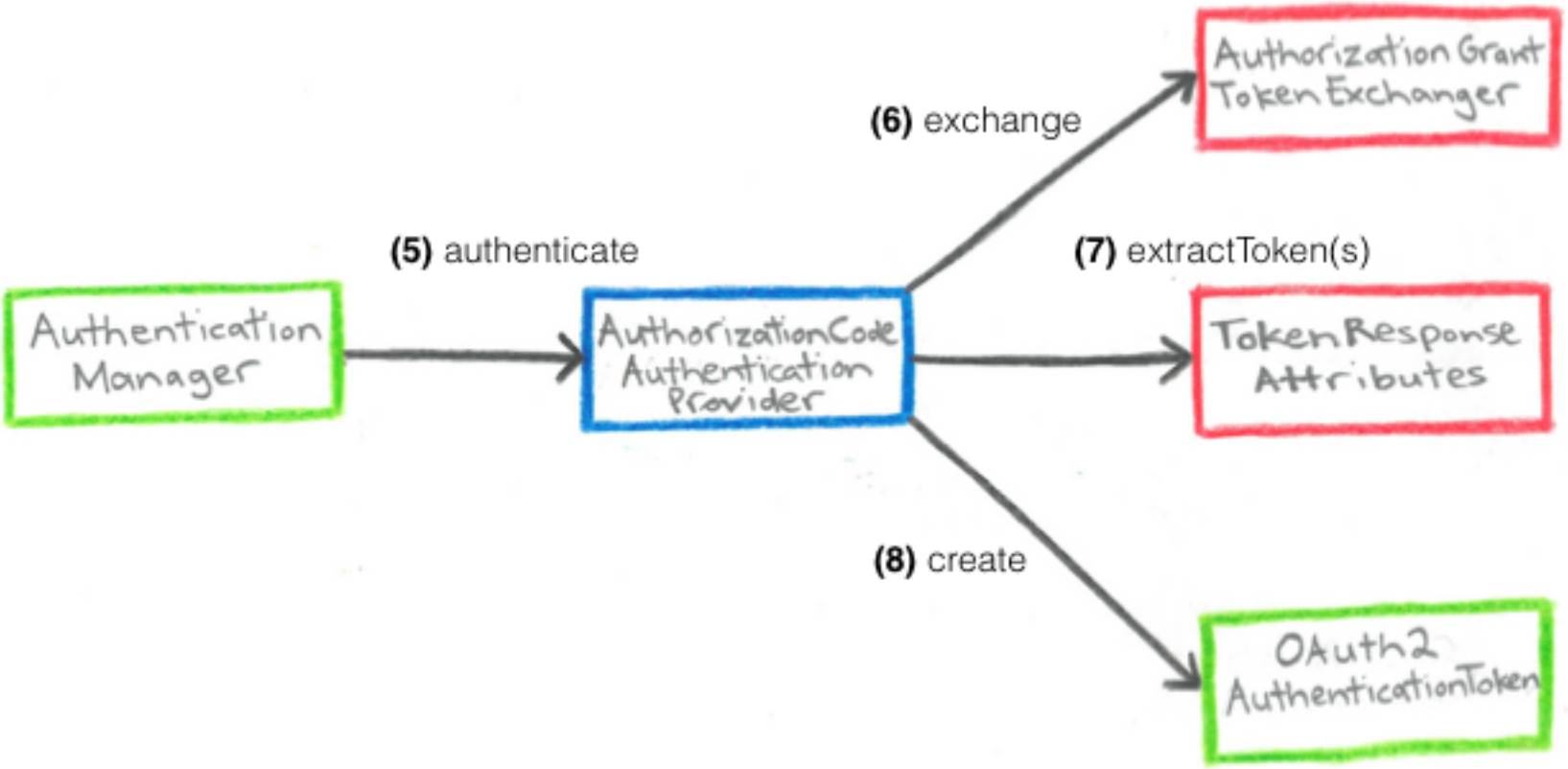
Authorization Response

HTTP/1.1 302 Found
Location: https://client.example.com?
code=SplxlOBeZQQYbYS6WxSbIA&
state=wQa4c67t

Obtaining Access Token

Exchanging the Authorization Code

Exchanging the Authorization Code



Obtaining Access Token

Recap

- Client sends Access Token request
- Authorization Server responds with Access Token and optional Refresh Token
- ID Token is also returned for OpenID Connect 1.0 flow

1. OAuth 2.0 Overview
2. Requesting Authorization
3. Obtaining Access Token / ID Token
4. **Accessing UserInfo Resource**

UserInfo Endpoint

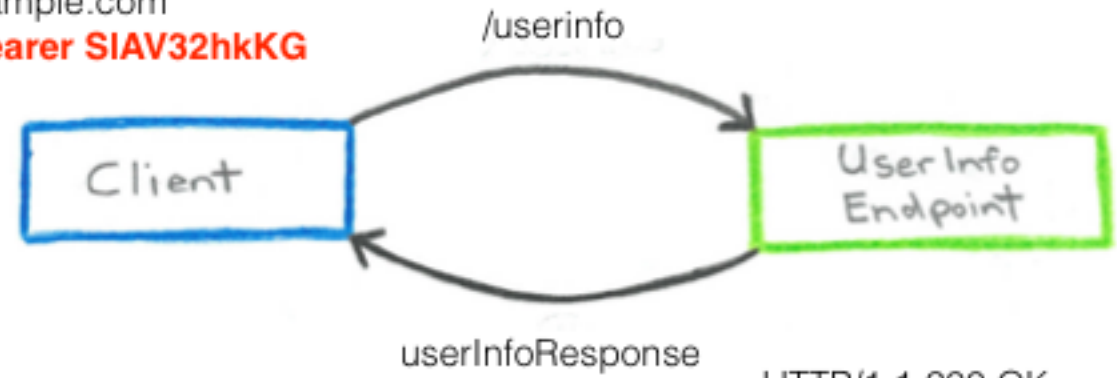
Is a **Protected Resource**
that returns **claims (or attributes)**
about the **Authentication of an End-User**

- **Standard** for OpenID Connect Provider's
- **Non-standard** for OAuth 2.0 Provider's

Accessing UserInfo Resource

OpenID Connect 1.0

```
GET /userinfo HTTP/1.1
  Host: identity.example.com
  Authorization: Bearer SIAV32hkKG
```




```
HTTP/1.1 200 OK
  Content-Type: application/json

  {
   "sub": "248289761001",
   "name": "Jane Doe",
   "given_name": "Jane",
   "family_name": "Doe",
   "preferred_username": "j.doe",
   "email": "janedoe@example.com",
   "picture": "http://example.com/janedoe/me.jpg"
  }
```

Standard Claims

OpenID Connect 1.0

Returned in the **UserInfo Response** and/or the **ID Token**

Member	Type
sub	string
name	string
given_name	string
family_name	string
preferred_username	string
profile	string
email	string
gender	string
locale	string
updated_at	number

Accessing UserInfo Resource

OAuth 2.0

OAuth 2.0

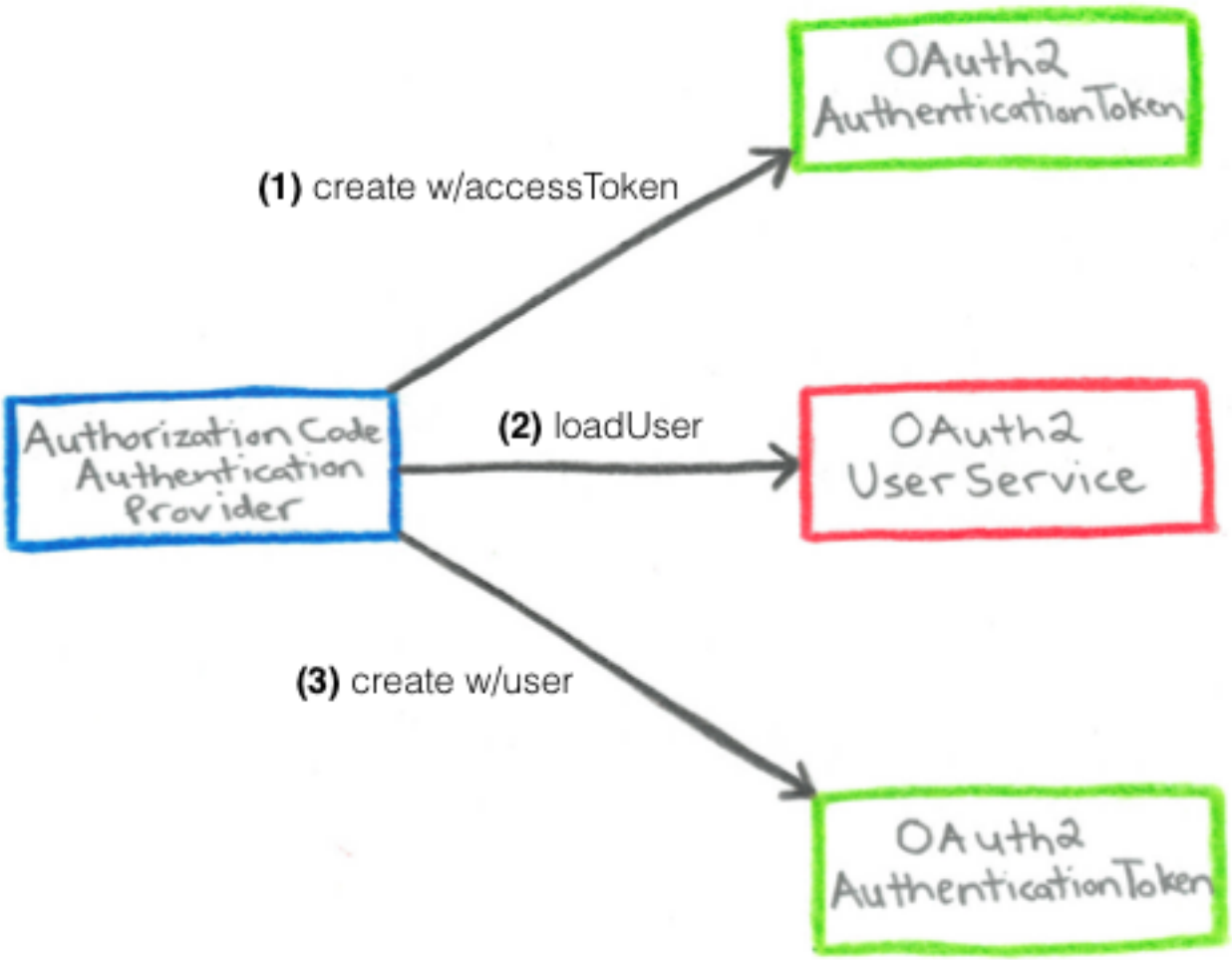
GitHub

https://api.github.com/user


```
{
  "login": "octocat",
  "id": 1,
  "avatar_url": "https://github.com/images/error/octocat_happy.gif",
  "gravatar_id": "",
  "url": "https://api.github.com/users/octocat",
  "html_url": "https://github.com/octocat",
  "followers_url": "https://api.github.com/users/octocat/followers",
  "following_url": "https://api.github.com/users/octocat/following{/other_user}",
  "gists_url": "https://api.github.com/users/octocat/gists{/gist_id}",
  "starred_url": "https://api.github.com/users/octocat/starred{/owner}{/repo}",
  "subscriptions_url": "https://api.github.com/users/octocat/subscriptions",
  "organizations_url": "https://api.github.com/users/octocat/orgs",
  "repos_url": "https://api.github.com/users/octocat/repos",
  "type": "User",
  "name": "monalisa octocat",
  "company": "GitHub",
  "blog": "https://github.com/blog",
  "location": "San Francisco",
  "email": "octocat@github.com",
  "hireable": false,
  "bio": "There once was...",
  "public_repos": 2,
  "public_gists": 1,
  "created_at": "2008-01-14T04:33:35Z",
  "updated_at": "2008-01-14T04:33:35Z"
}
```


Accessing UserInfo Resource

Implementation Artifacts



Accessing UserInfo Resource

Recap

Recap

- UserInfo Endpoint is a Protected Resource
- UserInfo Endpoint returns *"Standard Claims"* about the Authentication of an End-User
- Non-standard for OAuth 2.0 Provider's

Summary

- OAuth 2.0 Overview
- Requesting Authorization
- Obtaining Access Token / ID Token
- Accessing UserInfo Resource

Demo

OAuth 2.0 Login Sample

github.com/spring-projects/spring-security/
tree/master/
samples/boot/oauth2login

Pivotal | InfoQ

Thanks!