OAuth

OAuth

- Luxury cars often come with a valet key.
- The car can be driven for a short distance.
- Provides limited access to the car.

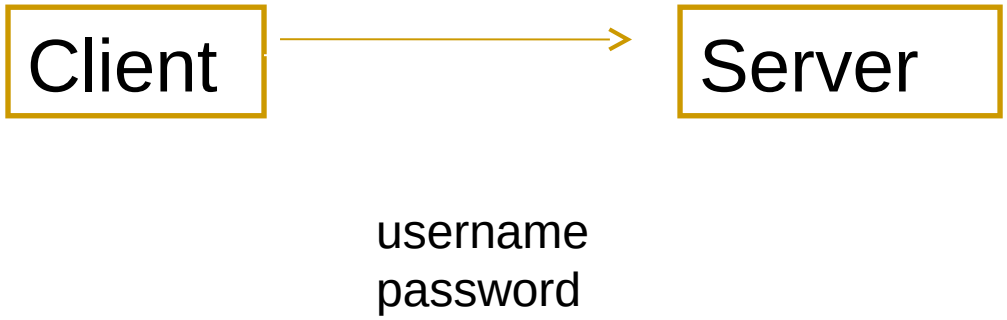
Printing Photos

- You might want to give the printing service limited access to you Flickr account.
- Or a social networking site limited access to your Google address book.
- "OAuth provides a method for users to grant third-party access to their resources without sharing their passwords.
- It also provides a way to grant limited access (in scope, duration, etc.).

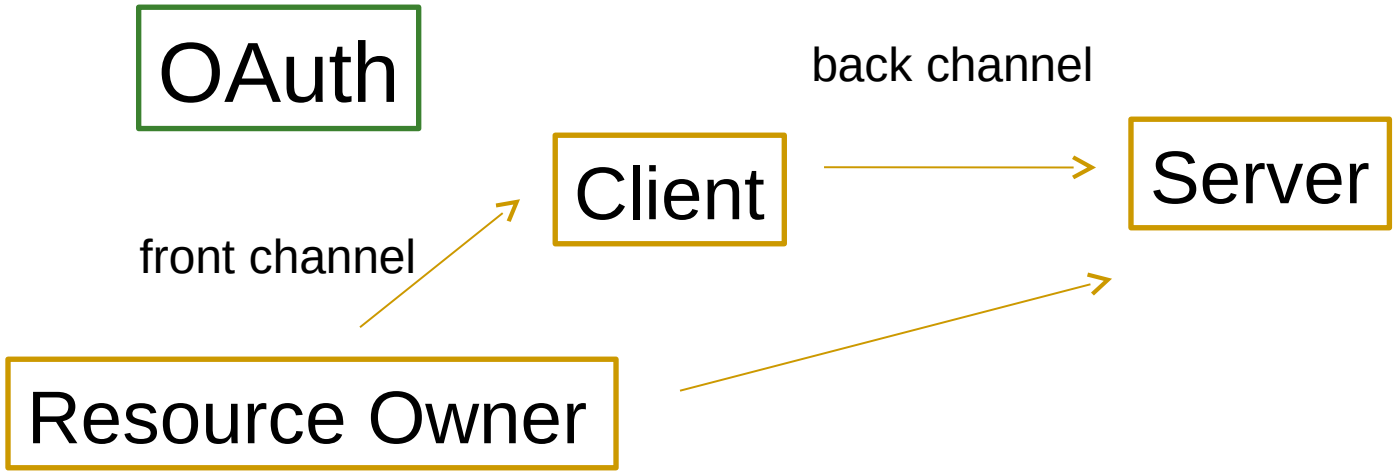
Client Server v. OAuth

- Client Server :
  - Client provides access credentials.
  - Allowed access to resources on the server.
- OAuth introduces a third role to this model, the resource owner.
- The client is now not the resource owner, but is acting on behalf of the resource owner.
- The client requests access to resources owned by the resource owner and hosted by the server.

Client Server Authentication



OAuth Authentication



OAuth

- The resource owner authorizes the client to access their resources on the Server.
- This provides limited access to the resouces.
- Limited access
  - restricted scope
  - limited lifetime

OAuth

- OAuth based on
  - Google AuthSub
  - Yahoo BBAuth
  - Flickr API
- OAuth has built-in support for desktop applications, mobile devices, set-top boxes, as well as websites.

Terminology

Terminology

- client (TwitterClient.java)
  - Application (HTTP client) capable of making OAuth-authenticated requests
- server (Twitter)
  - An HTTP server capable of accepting OAuth-authenticated requests.
- protected resource (users tweets)
  - An access-restricted resource that can be obtained from the server using an OAuth-authenticated request

Terminology

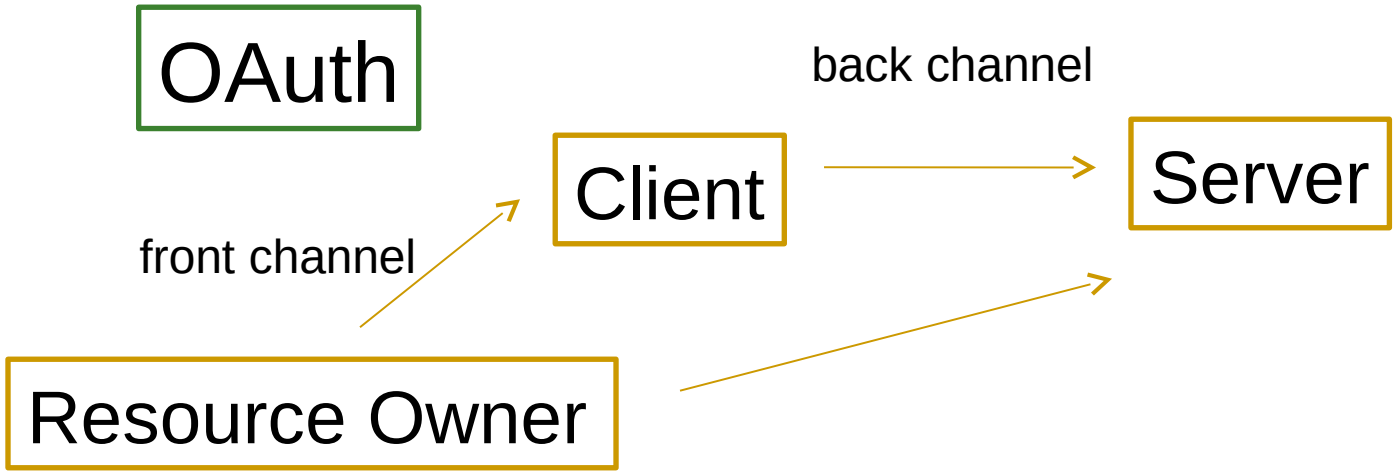
- resource owner (user)
  - An entity capable of accessing and controlling protected resources by using credentials to authenticate with the server.
- credentials
  - A pair of
    - a unique identifier (key or token).
    - a matching shared secret.

Three types of Credentials

- client credentials - identifies the application
  - Consumer (or API) Key and Secret
- request credentials (temporary credentials)
  - Request Key and Secret
- access credentials (token credentials)
  - Access Key and Secret

Redirection-Based Authorization

OAuth Authentication



Redirection-Based Authorization

- User is redirected by the client (application) to the server (Twitter)
- Users authenticate directly with the server, instructing the server to given credentials to the client for use with the authentication method.

Redirection-Based Authorization

- Visit photo printing site
  - printer.example.com
- Request some photos to be printed and mailed to someone.
- Specify that photos are stored at
  - photos.example.net (server)
- User - paul

Redirection-Based Authorization

- printer.example.com has previously registered with photos.example.net and obtained client credentials.
- printer.example.com requests request credentials from photos.example.net and obtains these.
- Customer redirected to photos.example.net.
- Logon to photos.example.net.

Redirection-Based Authorization

- Message that some printing service has request access to your photos. Do you want to allow such access. Yes.
- Request credentials of printer.example.com marked as resource-owner-authorized.
- Customer redirected back to Printing site.
- printer.example.com obtains access credentials.
- printer.example.com downloads the photos from photos.example.net and you are prompted which ones to print.

Access URLs

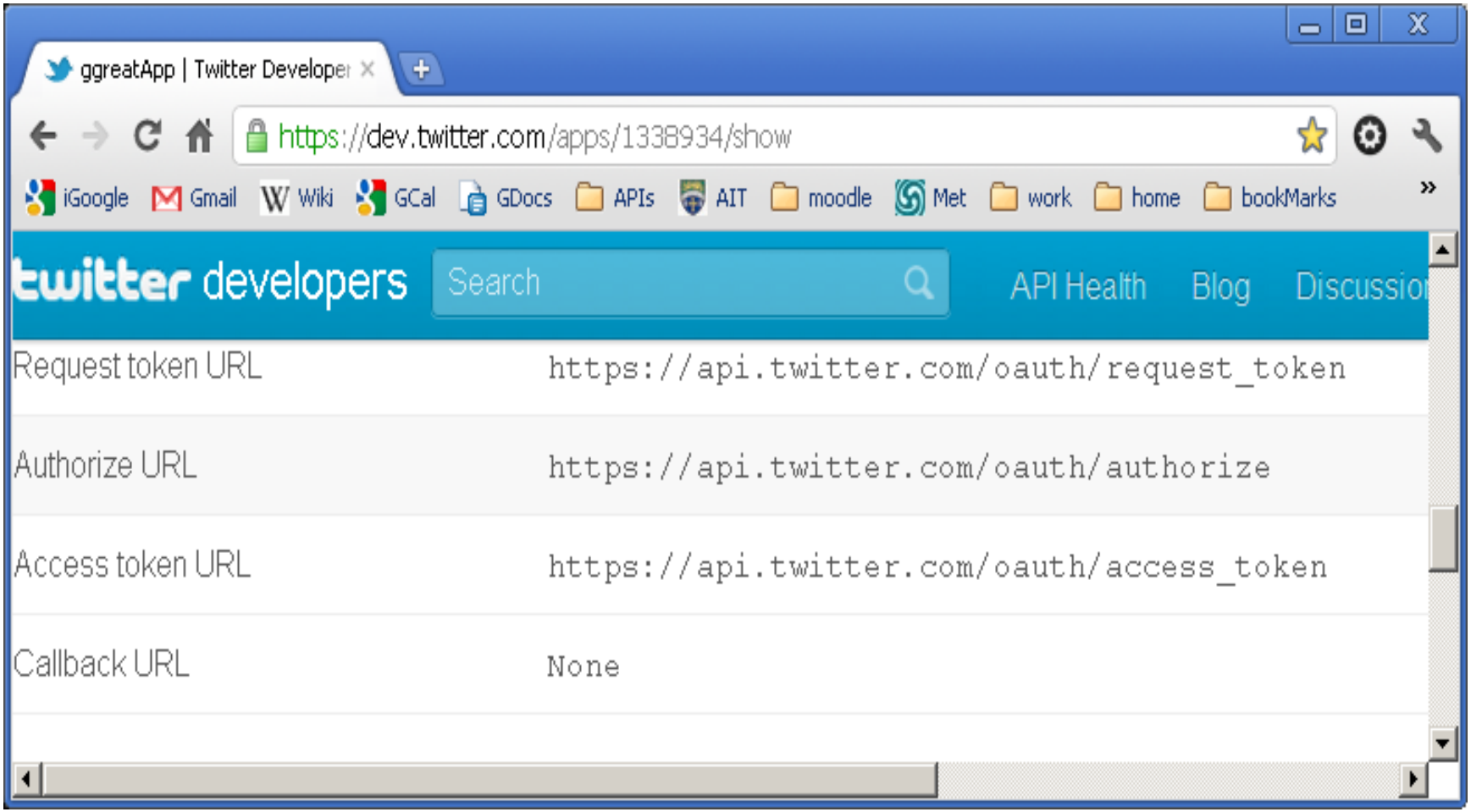
- A number of (server) URLs required to make OAuth requests.

Access URLs - Example

- Request URL
  - Ask for <u>request credentials</u>, e.g.
  - https://photos.example.net/initiate
- Authorize URL
  - For user authorization e.g.
  - https://photos.example.net/authorize
- Access URL
  - Ask for <u>access credentials</u>, e.g.
  - https://photos.example.net/token

Twitter Example

Twitter Access URLs



Register an Application (Twitter)

- You will be provided with (over HTTPS)
  - consumer key (API key)
  - consumer secret (API secret)
- You will specify a callback URL if you are accessing Twitter from a Web application.

Scenario – Web Application Client

- Client asks for <u>request credentials</u>
- Client redirects user to Server (authorize URL)
- Server redirects user to client callback URL.
- Client asks for <u>access credentials</u>.
- Client requests resources.

Scenario – Desktop Client

- Client asks for request credentials
- Client asks user to go to Server (authorize URL) and get a PIN.
- User logs on and gets PIN.
- User enters it in to client.
- Client asks for access credentials (sends PIN)
- Client requests resources.

Example Requests

Client asks for Request Credentials

```
POST /initiate HTTP/1.1
    Host: photos.example.net
    Authorization: OAuth realm="Photos",
      oauth_consumer_key="dpf43f3p2l4k3l03",
      oauth_signature_method="HMAC-SHA1",
      oauth_timestamp="137131200",
      oauth_nonce="wIjqoS",
      oauth_callback="http%3A%2F%2Fprinter.example.com%2Fready",
      oauth_signature="74KNZJeDHnMBp0EMJ9ZHt%2FXKycU%3D"
```

Client asks for Request Credentials

- Sends the consumer (API) key.
- Authenticates the request by sending the HMAC signature calculated with consumer secret.
- Timestamp and nonce sent to prevent replay attacks.
-

Server Responds

```
HTTP/1.1 200 OK
    Content-Type: application/x-www-form-urlencoded
    oauth_token=hh5s93j4hdidpola&oauth_token_secret=hdhd0244k9j7ao03&
    oauth_callback_confirmed=true
```

- Server responds with <u>request credentials</u>
  - request key - hh5s93j4hdidpola
  - request secret - hdhd0244k9j7ao03

Client requests Access Credentials

```
POST /token HTTP/1.1
   Host: photos.example.net
   Authorization: OAuth realm="Photos",
      oauth_consumer_key="dpf43f3p2l4k3l03",
      oauth_token="hh5s93j4hdidpola",
      oauth_signature_method="HMAC-SHA1",
      oauth_timestamp="137131201",
      oauth_nonce="walatlh",
      oauth_verifier="hfdp7dh39dks9884",
      oauth_signature="gKgrFCywp7rO0OXSjdot%2FIHF7IU%3D"
```

Client asks for Request Credentials

- Sends the consumer (API) key.
- Also sends the request key.
- Authenticates the request by sending the HMAC signature calculated with these secrets.

Server supplies access credentials

```
HTTP/1.1 200 OK
    Content-Type: application/x-www-form-urlencoded
    oauth_token=nnch734d00sl2jdk&oauth_token_secret=pfkkdhi9sl3r4s00
```

- These are the access (token) credentials.
  - access key: nnch734d00sl2jdk
  - access secret: pfkkdhi9sl3r4s00

Client requests resources using Access Credentials

```
GET /photos?file=vacation.jpg&size=original HTTP/1.1
    Host: photos.example.net
    Authorization: OAuth realm="Photos",
      oauth_consumer_key="dpf43f3p2l4k3l03",
      oauth_token="nnch734d00sl2jdk",
      oauth_signature_method="HMAC-SHA1",
      oauth_timestamp="137131202",
      oauth_nonce="chapoH",
      oauth_signature="MdpQcU8iPSUjWoN%2FUDMsK2sui9I%3D"
```

Client asks for Request Credentials

- Sends the consumer (API) key.
- Also sends the access key.
- Authenticates the request by sending the HMAC signature calculated with the corresponding secrets.