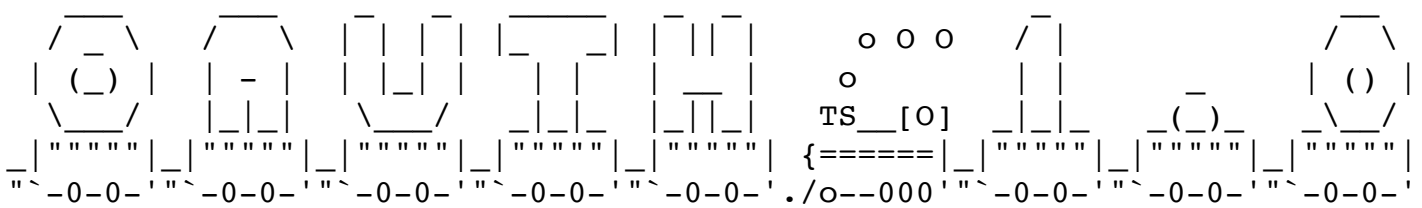


History

- Late 2006
- Blaine Cook of Twitter, Chris Messina and Larry Halff of Ma.gnolia all wanted something like OpenID and Flickr Auth / Google AuthSub / and Yahoo! BBAuth
- They wanted an open standard for auth without passwords or copy/paste tokens
- This is different then OpenAuth (AOL)
- Mid July a bunch more people were involved and the standard took shape

History

- December 4th 2007 - OAuth Core 1.0 specification was declared final at the Internet Identity Workshop
- March 9th 2009 – OAuth Core 1.0 Editor's Edition released - a complete rewrite of the entire specification including many fixes and clarifications.
- April 15th 2009 – first published exploit that affected all systems
- April 23rd 2009 – first security advisory
- June 24th 2009 – OAuth 1.0 rev A released
- April 2010 - IETF published RFC 5849 as an informational RFC, replacing the OAuth Core 1.0 Revision A specification

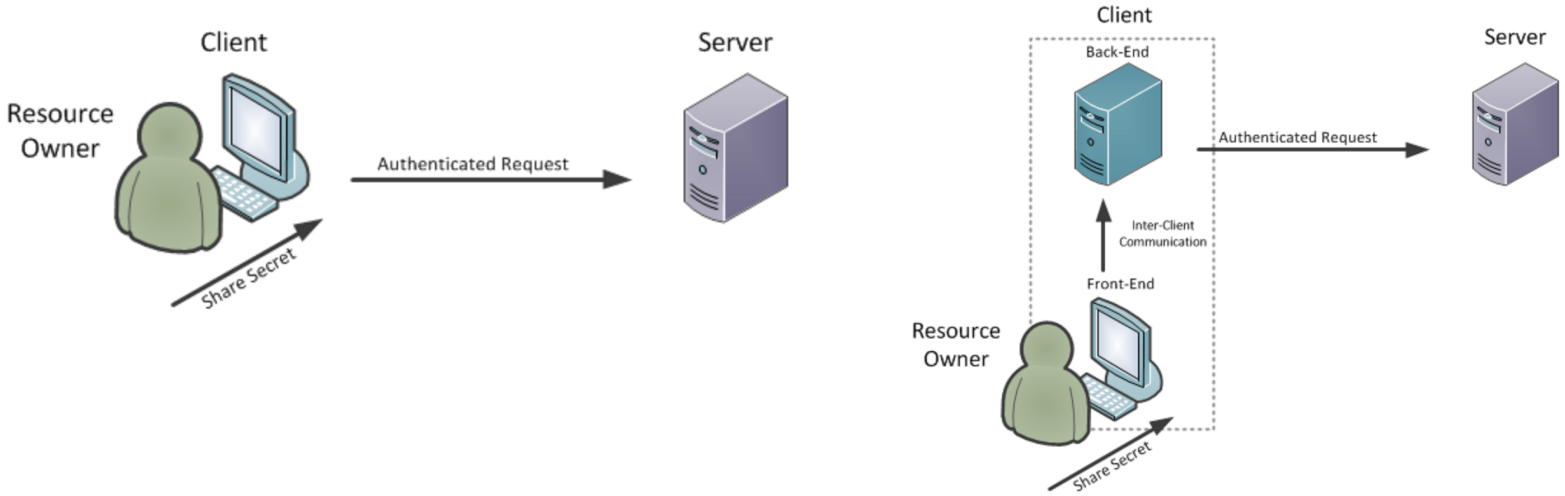
Terminology

- Three roles:
  - Client (consumer)
  - Server (service provider)
  - Resource Owner (user)
  - Note: In some cases the client is the resource owner
- Protected Resources
  - A resource stored on a server and which requires authentication in order to access
  - Controlled/owned by the resource owner
  - Can be data or services

Terminology

- Traditionally a client uses its credentials to access its resources on a server

- Sometimes a client is acting on a user's behalf, and in this case they are using a resource owner's credentials to make requests



Terminology

- Three kinds of credentials:
  - Client credentials (consumer key and secret)
    - Used to authenticate the client
  - Temporary credentials (request token and secret)
    - Used to identify the authorization request
    - Helpful when accommodating different clients (web, desktop, mobile, etc.)
  - Token credentials (access token and secret)
    - Consists of a token identifier and a shared secret
    - Limited in scope and duration
    - Can be revoked at any time by the resource owner without affecting other outstanding tokens

Protocol Workflow

- This example was taken from here: https://hueniverse.com/oauth/guide/workflow/
- Written by one of the original authors of the specification

Protocol Workflow

- Jane went on vacation and took some photos
- She gets home and wants to share them with friends
- She uses a photo sharing site called Faji
- She logs into her Faji account and uploads 2 photos which are then marked private
- Who is the resource owner? Who is the server?


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://faji.com/photos
faji
foto journeys
Hi Jane,,
Your have uploaded 2 photos:
Share with Friends

Protocol Workflow

- Jane wants to have some of the photos printed and decides to use a photo printing service called Beppa
- Beppa must use OAuth credentials to access the private photos hosted on Faji
- We have three roles now:
  - Jane the resource owner
  - Faji the server
  - Beppa the client


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://beppa.com
eco friendly printing
beppa
Where are your photos located?
Flickr
Fotolog
Faji
Continue

Protocol Workflow

- When Beppa added support for Faji they received a client identifier and secret from Faji to use with their OAuth-enabled API
- When Jane first clicks continue to use the Faji integration on Beppa a set of temporary credentials are requested from Faji (by Beppa)
  - These credentials are not resource-owner specific at this point
- She is then redirected to Faji and asked to sign in using her Faji credentials
- OAuth requires that the server first authenticate the user, then ask them to grant access to the client
- At no point are her username and password shared with Beppa


Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://faji.com/oauth/authorize
faji
foto journeys
Please sign-in to your account:
Username: janedoe
Password: ••••••••
Continue

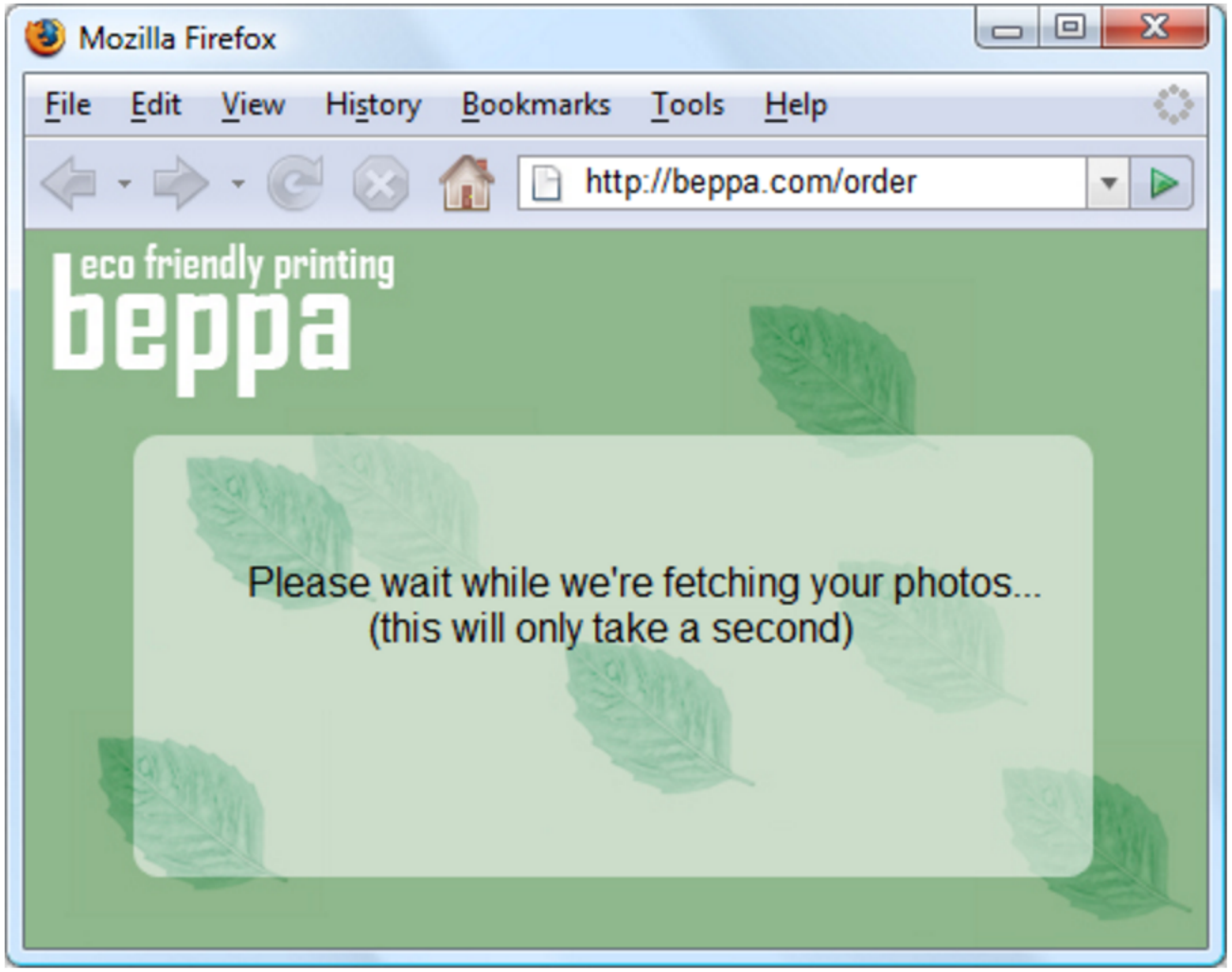
Protocol Workflow

- Faji informs Jane who is requesting access to the resource and what the privileges/conditions of access are
- She can approve or deny access at this point


Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://faji.com/oauth/confirm
faji
foto journeys
beppa.com is requesting access to your private photo albums:
eco friendly printing
beppa
If approved, beppa.com will have read only access for the next 1 hour
Approve
Deny

Protocol Workflow

- If Jane approves the request then Faji marks the temporary credentials as resource-owner authorized
- She is redirected back to Beppa with the temporary credential identifier


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://beppa.com/order
eco friendly printing
beppa
Please wait while we're fetching your photos...
(this will only take a second)

Protocol Workflow

- Before the resource is accessed by the client it must trade the authorized request token for an access token
- Request tokens are only good for during the user approval stage
- Access tokens are used for accessing protected resources
- After access token is obtained then Beppa (client) can access the photos (protected resources)


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://beppa.com/order
eco friendly printing
beppa
2 Photos found. Please select the ones you would like to print:
Print