Introduction to OAuth2

Dominick Baier

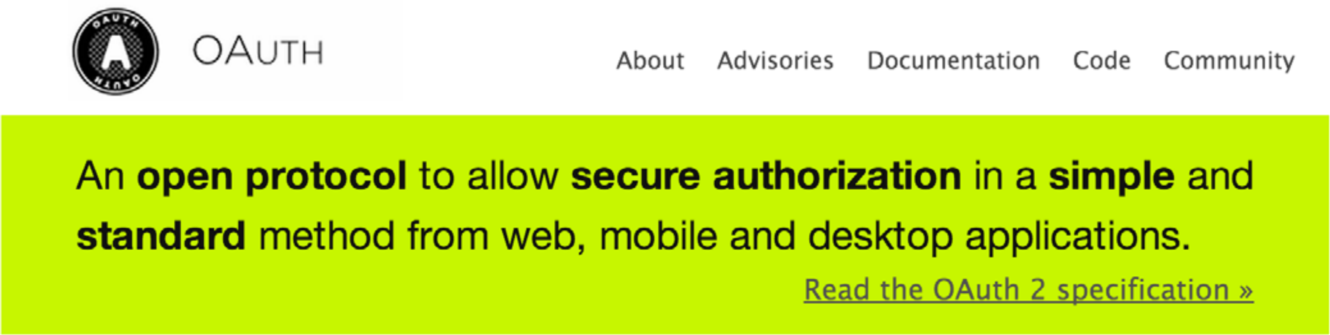
http://leastprivilege.com
@leastprivilege

pluralsight
hardcore developer training

Outline

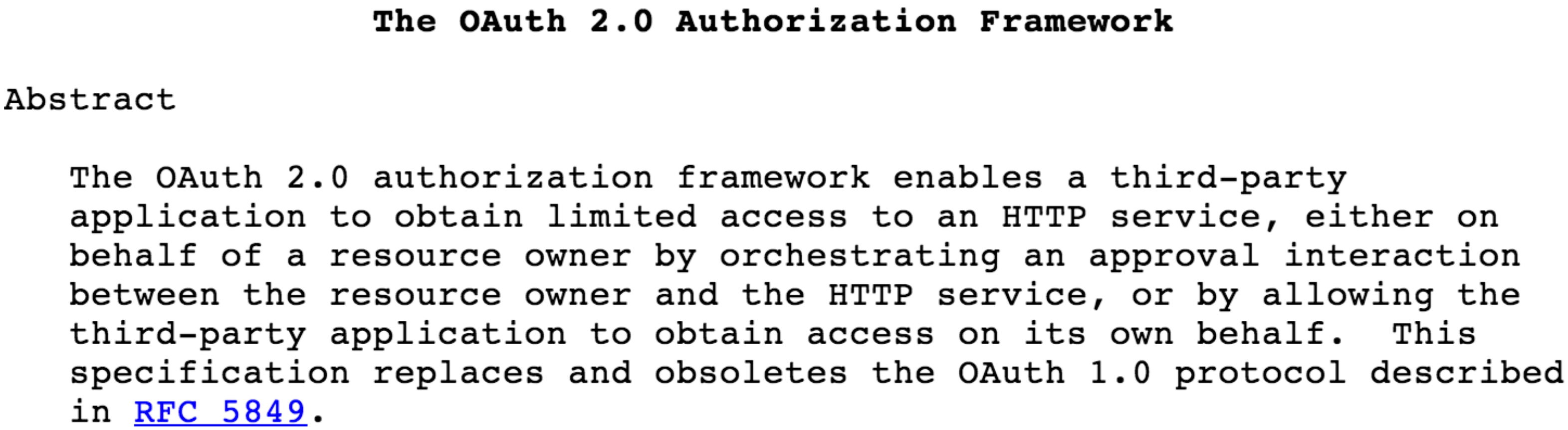
- Overview
- History
- Flows

What is OAuth2 ?



An **open protocol** to allow **secure authorization** in a **simple** and **standard** method from web, mobile and desktop applications.

Read the OAuth 2 specification »



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.

History

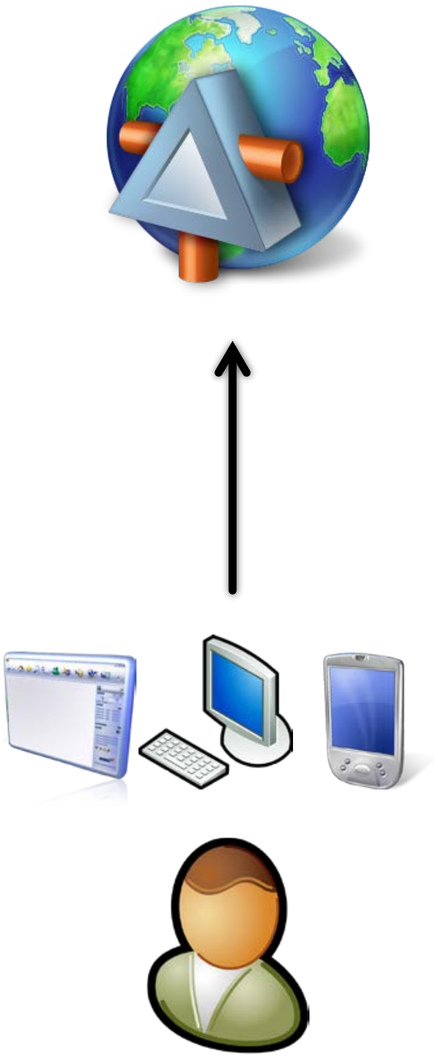
- OAuth started circa 2007
- 2008 - IETF normalization started in 2008
- 2010 - RFC 5849 defines OAuth 1.0
- 2010 - WRAP (Web Resource Authorization Profiles) proposed by Microsoft, Yahoo! And Google
- 2010 - OAuth 2.0 work begins in IETF

- Working deployments of various drafts & versions at Google, Microsoft, Facebook, Github, Twitter, Flickr, Dropbox…

- Mid 2012 – Lead author and editor resigned & withdraws his name from all specs

- October 2012 – RFC 6749, RFC 6750

High level overview



Resource Server

Client

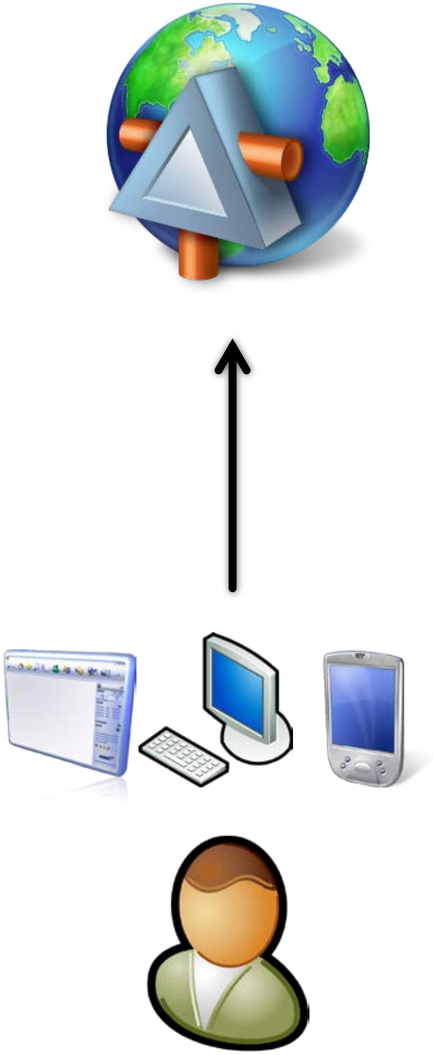
Resource Owner

http://hueniverse.com/2007/09/explaining-oauth/
http://amzn.com/1449311601

A-1 EZ
OK
PARK

No problem. Trust me.

High level overview

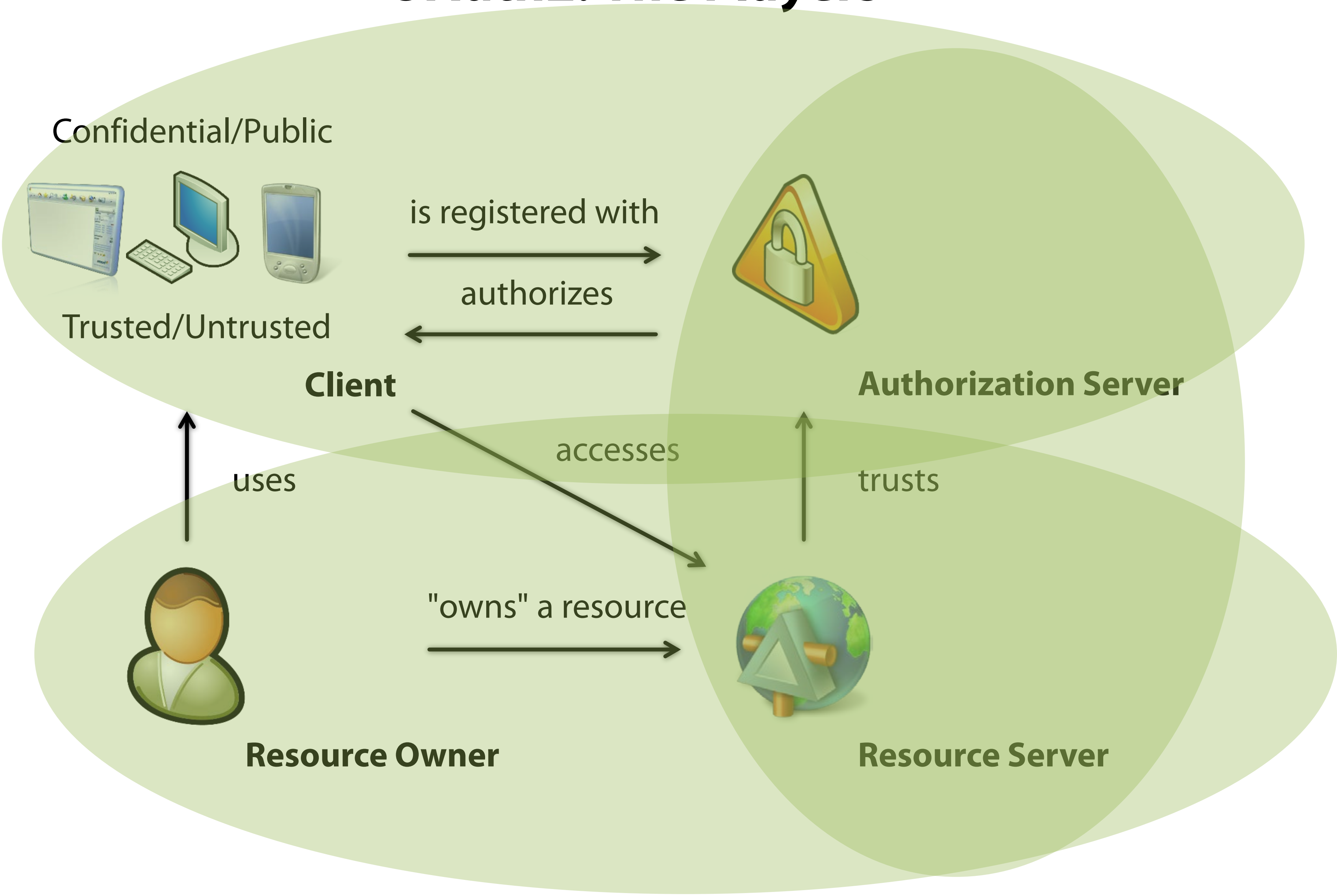


Resource Server

Client

Resource Owner

OAuth2: The Players



Confidential/Public

is registered with

authorizes

Trusted/Untrusted

Client

Authorization Server

accesses

uses

trusts

"owns" a resource

Resource Owner

Resource Server

OAuth2 Flows - with User Interaction

- **Authorization Code Flow**
  - Web application clients
    1. Request authorization
    2. Request token
    3. Access resource
- **Implicit Flow**
  - Native / local clients
    1. Request authorization & token
    2. Access resource

OAuth2 Flows -
no User Interaction

- **Resource Owner Password Credential Flow**
  - "Trusted clients"
    1. Request token with resource owner credentials
    2. Access resource
- **Client Credential Flow**
  - Client to Service communication
    1. Request token with client credentials
    2. Access resource

Summary

- **OAuth2 makes it HTTP/JSON friendly to request and transmit tokens**
  - typically for delegated authorization (access tokens)
- **Takes "multiple client" architectures into account**
  - clients can have varying trust levels

- **Since v2 of the spec is quite new, there's currently quite a discussion about its pros & cons. See Appendix A**