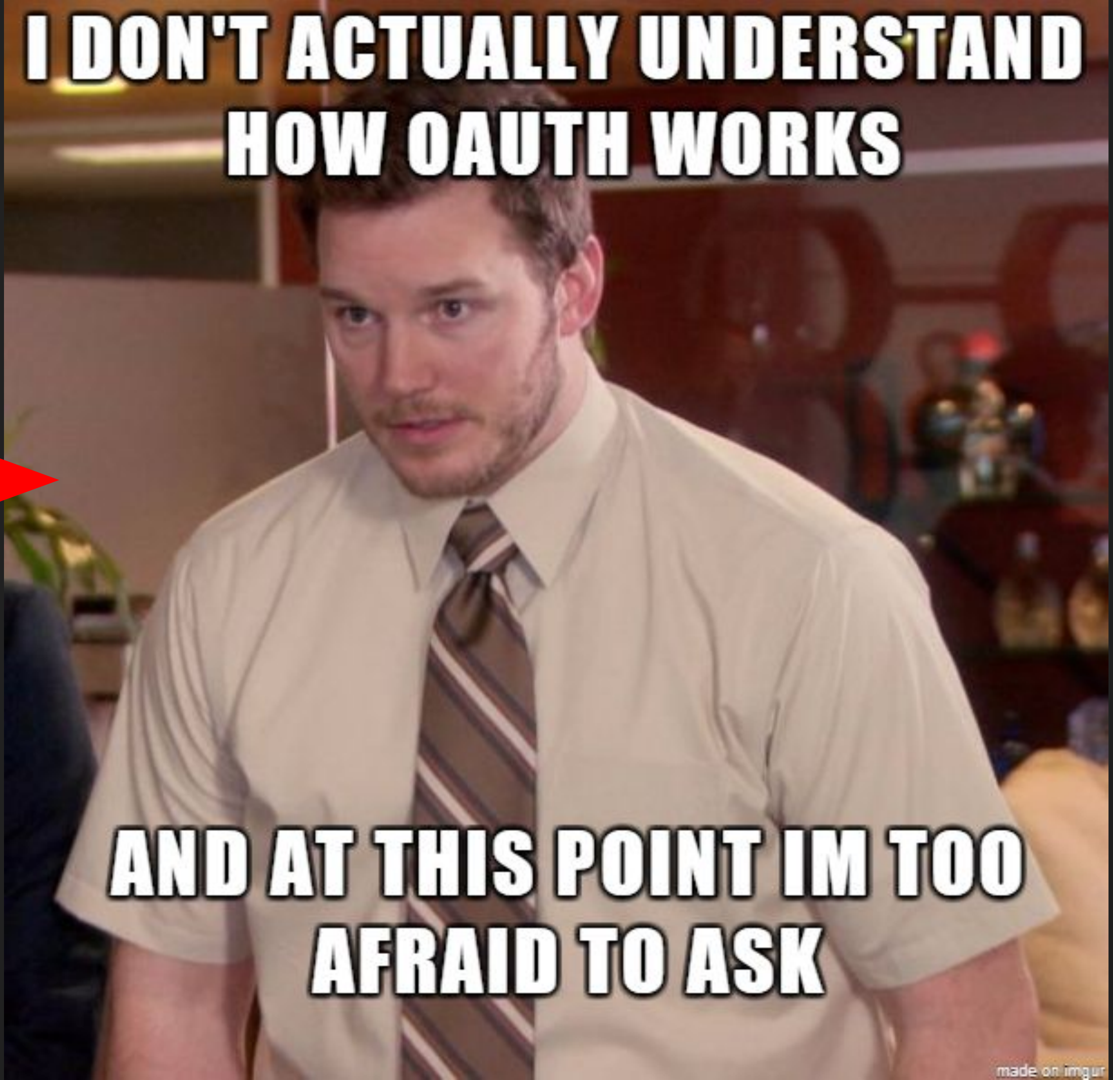
OAuth

Me two days before the end of the sprint


I DON'T ACTUALLY UNDERSTAND HOW OAUTH WORKS
AND AT THIS POINT IM TOO AFRAID TO ASK
made on imgur

1. The Theory of OAuth
2. REDACTED

Auth in General

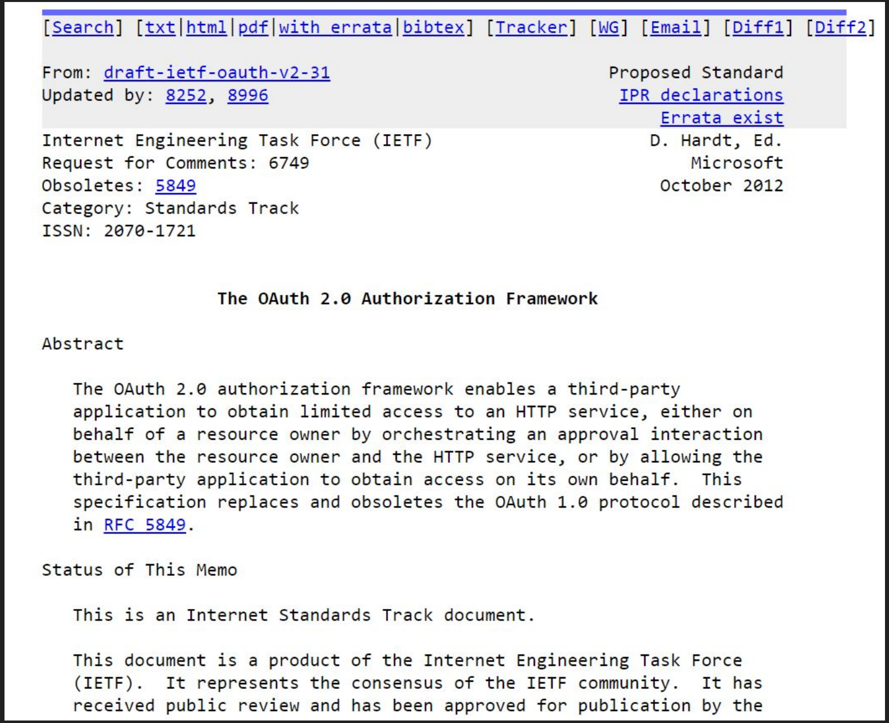
- Very important to get auth right!

- Don’t want to make any mistakes. How do we do that? Make Azure, Google and trusted open-source libraries do as much as possible.

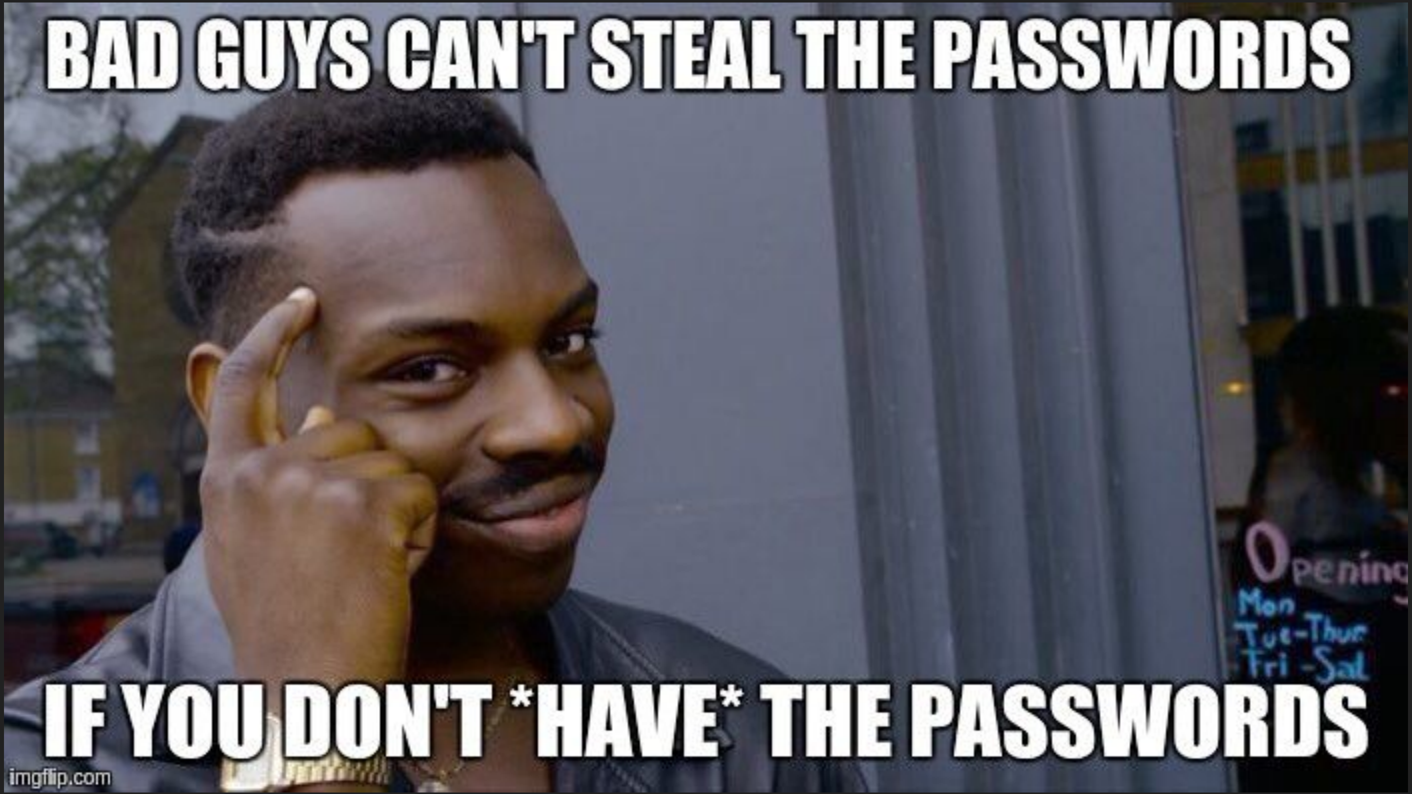
- Limit surface area of exposure. Don’t pass tokens around more than necessary.

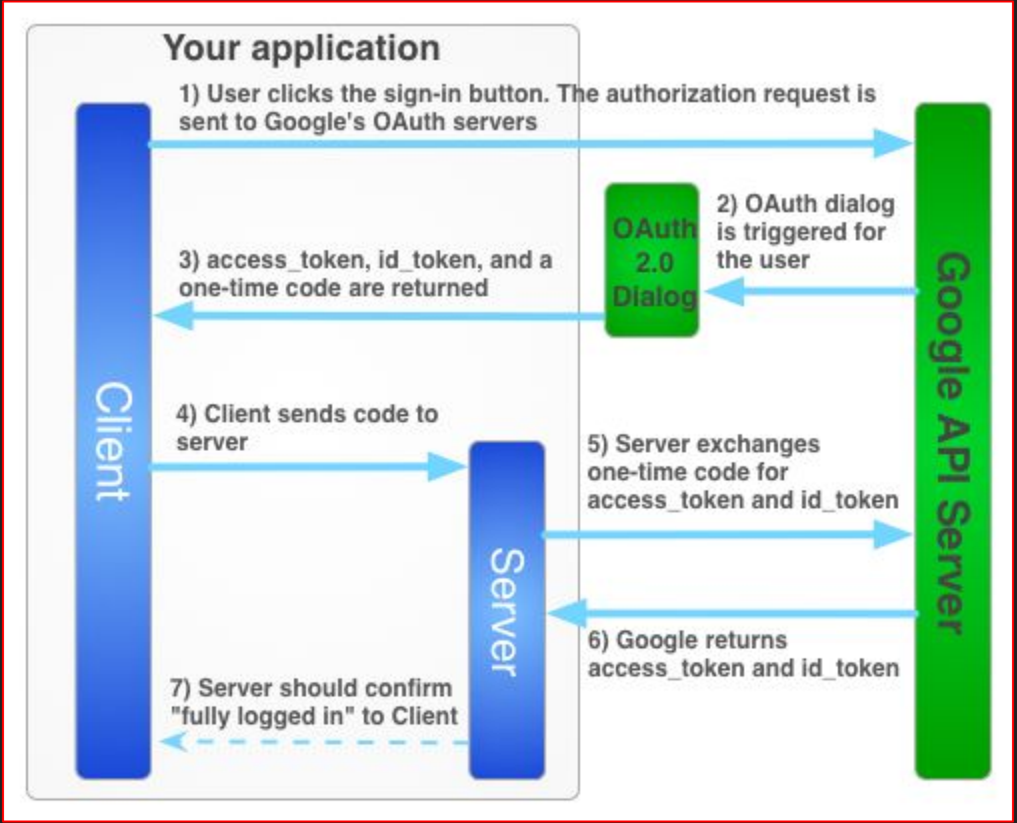
What is OAuth 2.0?

- OAuth is a protocol for authorization
- Offload the tricky parts of auth to a trusted auth provider eg Azure AD, Google OAuth etc.
- Different OAuth providers' implementations may differ a little.



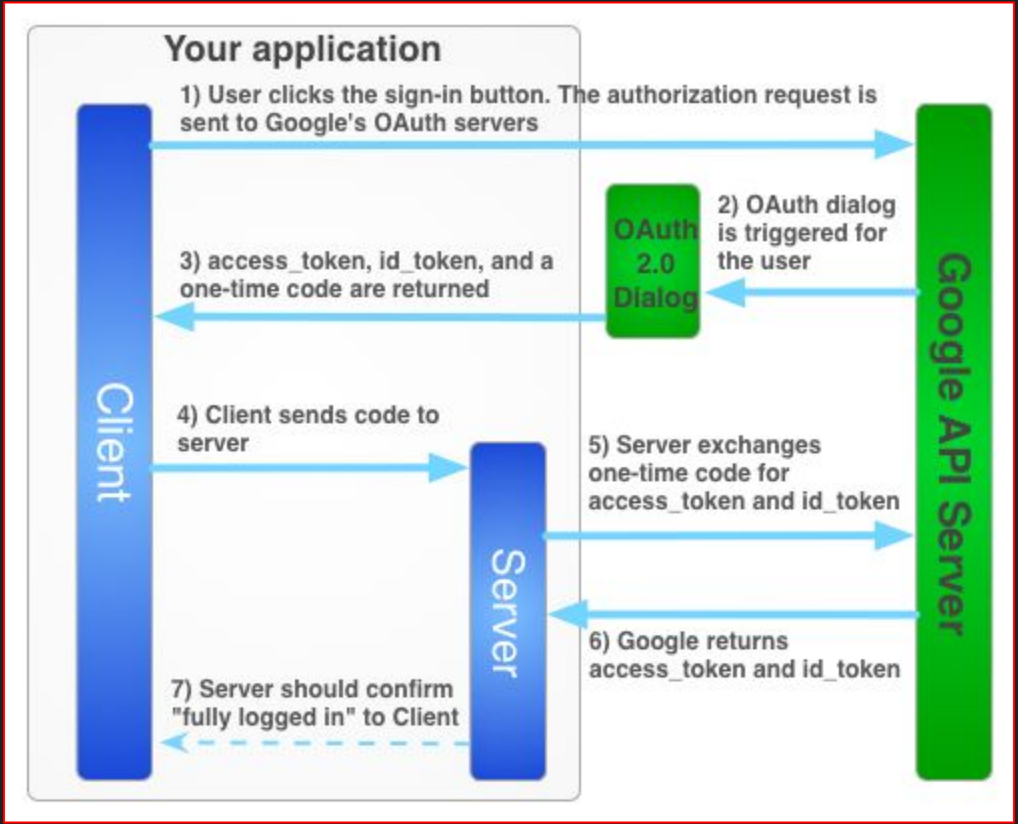
```
[Search] [txt|html|pdf|with errata|bibtex] [Tracker] [WG] [Email] [Diff1] [Diff2]

From: draft-ietf-oauth-v2-31                                Proposed Standard
Updated by: 8252, 8996                                       IPR declarations
                                                                Errata exist
Internet Engineering Task Force (IETF)                         D. Hardt, Ed.
Request for Comments: 6749                                         Microsoft
Obsoletes: 5849                                                 October 2012
Category: Standards Track
ISSN: 2070-1721


                 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.

Status of This Memo

   This is an Internet Standards Track document.

   This document is a product of the Internet Engineering Task Force
   (IETF).  It represents the consensus of the IETF community.  It has
   received public review and has been approved for publication by the
```


BAD GUYS CAN'T STEAL THE PASSWORDS
IF YOU DON'T *HAVE* THE PASSWORDS
imgflip.com


Your application
1) User clicks the sign-in button. The authorization request is sent to Google's OAuth servers
Client
OAuth 2.0 Dialog
2) OAuth dialog is triggered for the user
3) access_token, id_token, and a one-time code are returned
Google API Server
4) Client sends code to server
Server
5) Server exchanges one-time code for access_token and id_token
6) Google returns access_token and id_token
7) Server should confirm "fully logged in" to Client

- Diagram is from Google but principles apply to any OAuth provider eg Azure
- There are many different ways to implement/use OAuth depending on system architecture etc.
- Understanding what is going on in this diagram is key to understanding why/how our OAuth approach works, so it's quite important!

Image from https://developers.google.com/identity/sign-in/web/server-side-flow ;



Your application
1) User clicks the sign-in button. The authorization request is sent to Google's OAuth servers
Client
OAuth 2.0 Dialog
2) OAuth dialog is triggered for the user
3) access_token, id_token, and a one-time code are returned
Google API Server
4) Client sends code to server
Server
5) Server exchanges one-time code for access_token and id_token
6) Google returns access_token and id_token
7) Server should confirm "fully logged in" to Client

- Diagram is from Google but principles apply to any OAuth provider eg Azure

- Questions you might be asking:
  - What is the auth request in (1) ?
  - How does the OAuth dialog return the one-time code in (2) ?
  - What do we know about users who sign in with OAuth?
  - What does (7) mean?

Image from https://developers.google.com/identity/sign-in/web/server-side-flow ;