COMP6443 : Topic 2(Week 3)

Authorization and Session magic

sec edu

A NOTE ON ETHICS / LEGALITY

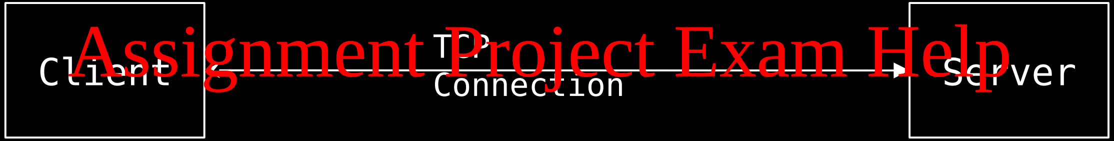
- UNSW hosting this course is an extremely important step forward.
- We expect a high standard of professionalism from you, meaning:
  - Respect the property of others and the university
  - Always abide by the law and university regulations
  - Be considerate of others to ensure everyone has an equal learning experience
  - Always check that you have written permission before performing a security test on a system

Always err on the side of caution. If you are unsure about

“NOT-A-HOMEWORK”

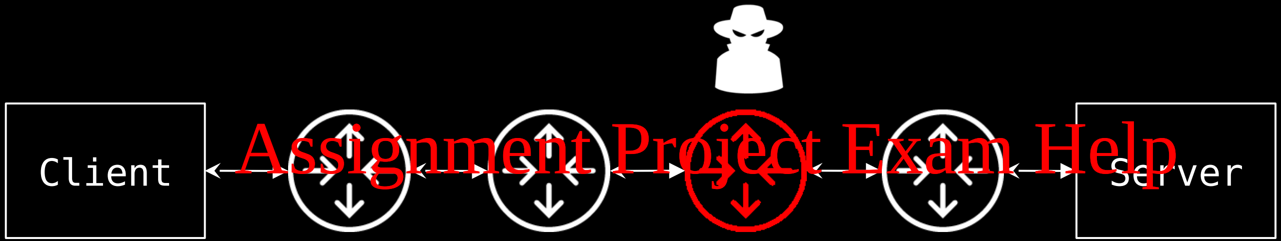
HTTP vs HTTPs

HTTP



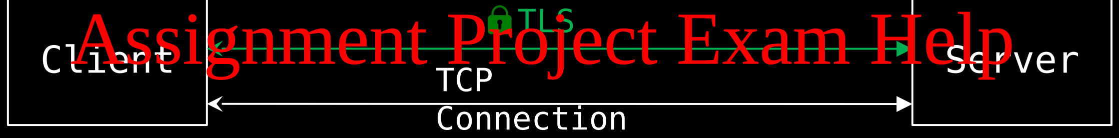
- Client opens a TCP connection to the Server
- Client and server now have a bidirectional communication stream
- Requests and responses sent over this channel

The Problem with HTTP



- The requests and responses are sent in cleartext
- A malicious party (Man in the Middle) in the path of the requests/responses can read and even modify them
- This could be:
    - A router routing the packet
    - An attacker on the client’s local network

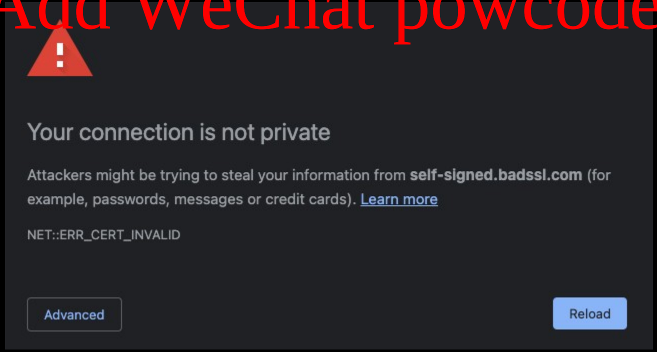
HTTPS



- Transport Layer Security (TLS) implemented on top of TCP connection
- Client and server now have an encrypted communication stream
- TLS was previously known as Secure Sockets Layer (SSL)

HTTPS

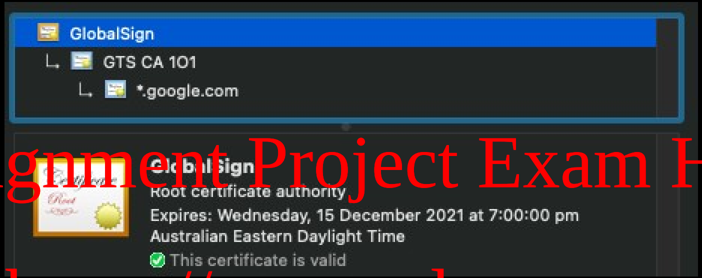
- But how do we know the server is the one we intend to connect to?
- During the TLS Handshake, the server sends a certificate indicating that it has control of the intended domain
- The browser (client) verifies the certificate, showing a privacy warning if it looks s


Your connection is not private

Attackers might be trying to steal your information from **self-signed.badssl.com** (for example, passwords, messages or credit cards). Learn more

NET::ERR_CERT_INVALID

How do certificates work?

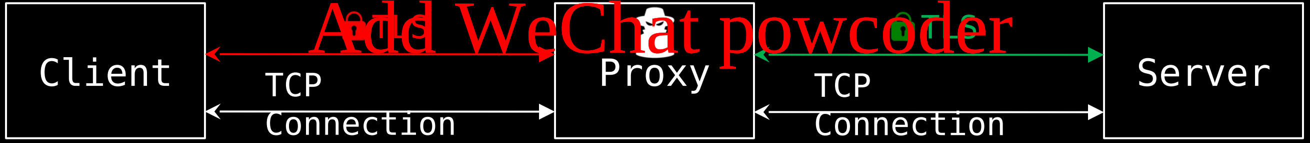

GlobalSign
GTS CA 101
*.google.com

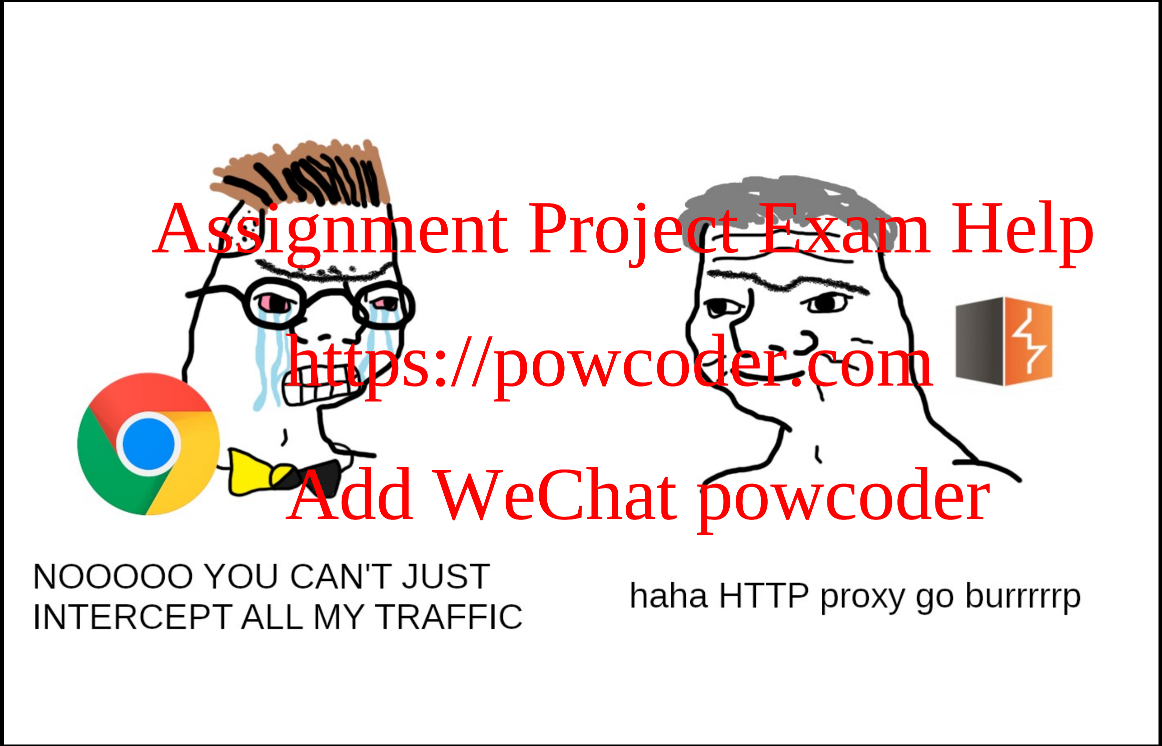
GlobalSign
Root certificate authority
Expires: Wednesday, 15 December 2021 at 7:00:00 pm Australian Eastern Daylight Time
This certificate is valid

- A certificate authority verifies that a server has control of a domain
- The CA issues the server a public key certificate and corresponding private key
- CAs themselves may be signed by another CA, resulting in a “certificate chain” with the root certificate authority at the top
- Operating Systems come loaded with a set of trusted root CAs

Can HTTPS be MiTM’d?

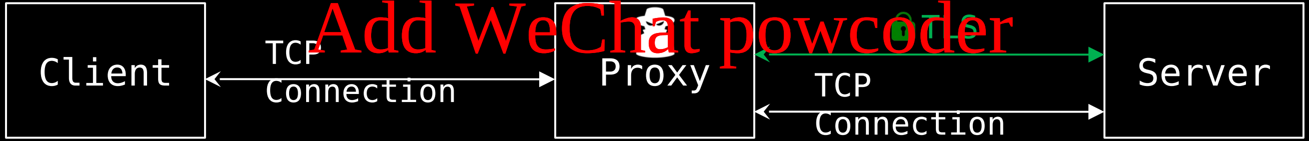
- A malicious attacker can sit between client/server as a proxy
- However, it will not be able to present a certificate signed by a trusted CA
- Best it can do is present a “self-signed” certificate, which will result in a privacy warning




NOOOOO YOU CAN'T JUST INTERCEPT ALL MY TRAFFIC
haha HTTP proxy go burrrrrp

How to be secure?

- Don’t serve any content over HTTP, redirect to HTTPs
- HOWEVER, this is still problematic as the initial HTTP request is still susceptible to MITM



OVERVIEW

Authentication → Session Management → Access Control (Authorisation)		
Is the user who they claim to be?	Is it still that user?	Is the user allowed to access this thing?

SESSION MANAGEMENT in 1999

CLIENT

```
GET /bestcms.php?page=supersecretpls&sessionid=123 HTTP/1.0
Host: www.lol.com
Cookie: username=uid0123456
Cookie: usertype=admin
```

SESSION MANAGEMENT in 2021

CLIENT

```
GET /1 HTTP/1.1
Host: www.lol.com
Cookie: SESSIONID=1234567890
```

SERVER

```
I recognize this SESSIONID. You are user ABCD
You have 1 item in your cart, item XYZ
You are not currently logged in.
```

ANATOMY OF A COOKIE

The main way we store session information is in a cookie.

Server → Client

```
Set-Cookie: SSID=abcdef; Domain=lol.com; Expires=Mon, 20 Jan 2020 20:20:20 GMT;
Secure; HttpOnly
```

name=value	the data to store
Domain	specifies the (sub)domain that the cookie belongs to
Expires	date when the cookie should be deleted
Secure	only send the cookie over secure connections (i.e. HTTPS)
HttpOnly	disable access to the cookie from JavaScript

Client → Server

```
Cookie: country=aus; SSID=abcdef
```

ATTACKING SESSIONS

- Session Creation
  - How are sessions created? Can I fake my own session?
  - Can I attack the PRNG, and generate my own cookie?
  - Can I “fixate” a session?

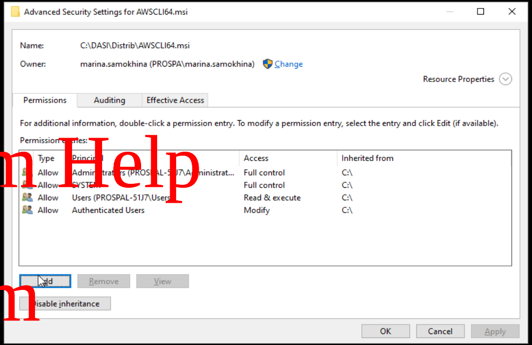
- Session Handling / Transfer / Usage
  - Can I steal the cookie through XSS (No ‘HttpOnly’ flag?)
  - Can I steal the cookie through redirecting to HTTPS.
  - What information does the site trust the user to provide?

- Session Cleanup
  - What happens when I click “log out”?
  - Under what conditions is a session actually destroyed? What happens then?
  - Do sessions time out correctly?

Authentication → Session Management → Access Control(Authorization)		
Is the user who they claim to be?	Is it still that user?	Is the user allowed to access this thing?

ACCESS CONTROL TYPES

- DAC (NTFS)
- RuleBAC and RoleBAC (Attribute)
- Parameter-based



Advanced Security Settings for AWSCLI64.msi

Name: C:\DASI\Distrib\AWSCLI64.msi

Owner: marina.samokhina (PROSPA\marina.samokhina) Change

Resource Properties

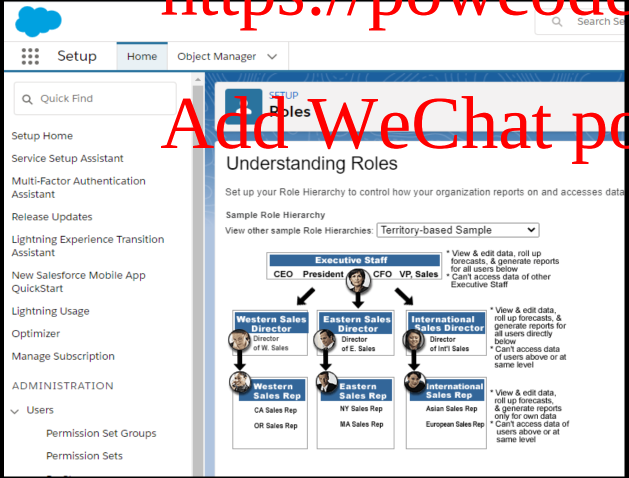
Permissions | Auditing | Effective Access

For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).

Permission entries:

Type	Principal	Access	Inherited from
Allow	Administrators (PROSPAL-51J7\Administrat...	Full control	C:\
Allow	SYSTEM	Full control	C:\
Allow	Users (PROSPAL-51J7\Users)	Read & execute	C:\
Allow	Authenticated Users	Modify	C:\

Add
Remove
View
Disable inheritance
OK
Cancel
Apply



Search Se
Setup
Home
Object Manager
Quick Find
Setup Home
Service Setup Assistant
Multi-Factor Authentication Assistant
Release Updates
Lightning Experience Transition Assistant
New Salesforce Mobile App QuickStart
Lightning Usage
Optimizer
Manage Subscription
ADMINISTRATION
Users
Permission Set Groups
Permission Sets
SETUP
Roles

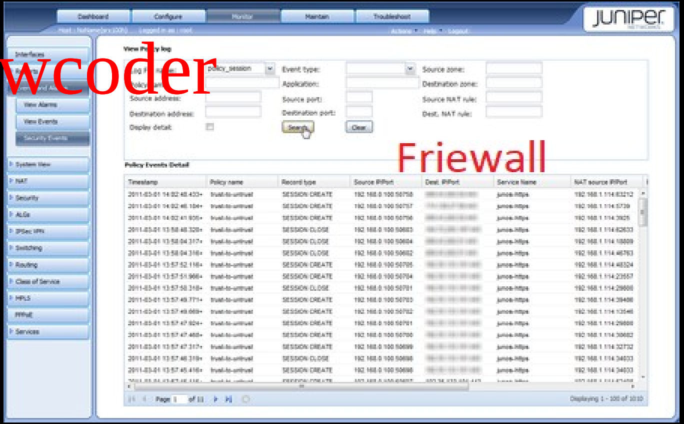
Understanding Roles

Set up your Role Hierarchy to control how your organization reports on and accesses data

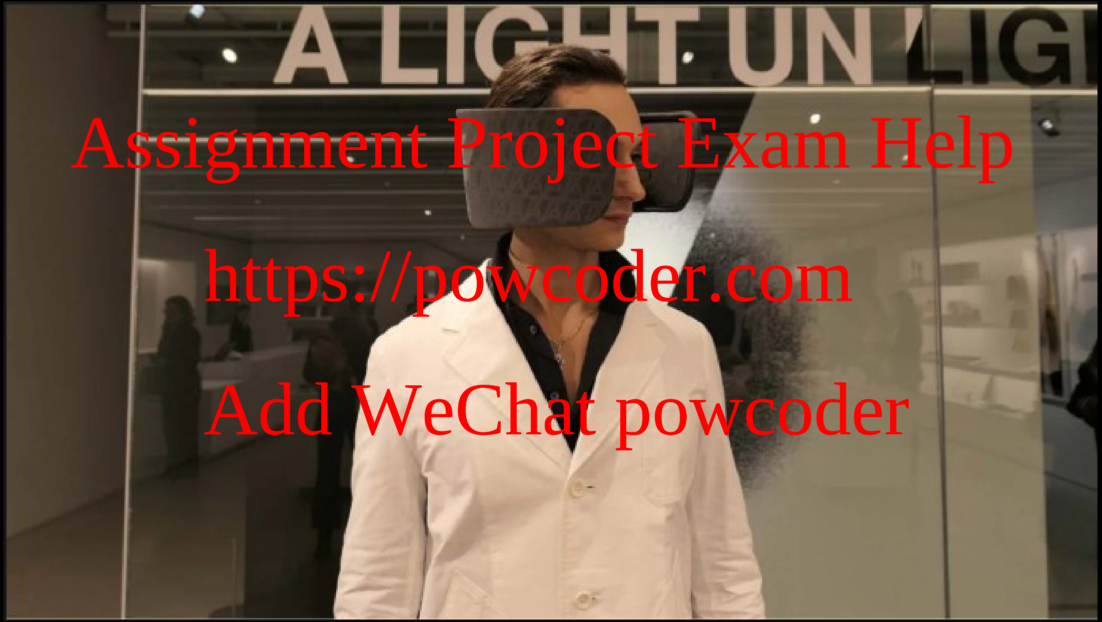
Sample Role Hierarchy

View other sample Role Hierarchies: Territory-based Sample

Executive Staff
CEO President CFO VP, Sales
* View & edit data, roll up forecasts, & generate reports for all users below
* Can't access data of other Executive Staff
Western Sales Director
Director of W. Sales
Eastern Sales Director
Director of E. Sales
International Sales Director
Director of Int'l Sales
* View & edit data, roll up forecasts, & generate reports for all users directly below
* Can't access data of users above or at same level
Western Sales Rep
CA Sales Rep
OR Sales Rep
Eastern Sales Rep
NY Sales Rep
MA Sales Rep
International Sales Rep
Asian Sales Rep
European Sales Rep
* View & edit data, roll up forecasts, & generate reports only for own data
* Can't access data of users above or at same level



ON /admin AND OTHER THINGS...



Your user context is enough to get your ssh privkey.

TYPES OF (WEB) ACCESS CONTROL

Security through obscurity

One-off access control

Rule-based access control

RBAC: HORIZONTAL vs VERTICAL

- Horizontal access control is making sure one user can’t access a different user’s data without permission
  - http://bank.com/statement.php?user_id=12078
- Vertical access control is making sure only administrative users can access administrative content
- Attacking vertical access control is commonly known as privilege escalation
  - http://bank.com/admin.php

ATTACKING ACCESS CONTROL

METHOD 1: BYPASS ENTIRELY

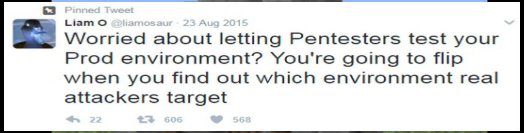
www.website.com

shop.website.com

blog.website.com

stage.website.com

db.website.com



Pinned Tweet

Liam O @liamosaur · 23 Aug 2015

Worried about letting Pentesters test your Prod environment? You're going to flip when you find out which environment real attackers target

22 606 568

api.website.com

dev.website.com

backup-syd.website.com

archive.website.com

s3 Buckets

github

pastebin

third party providers

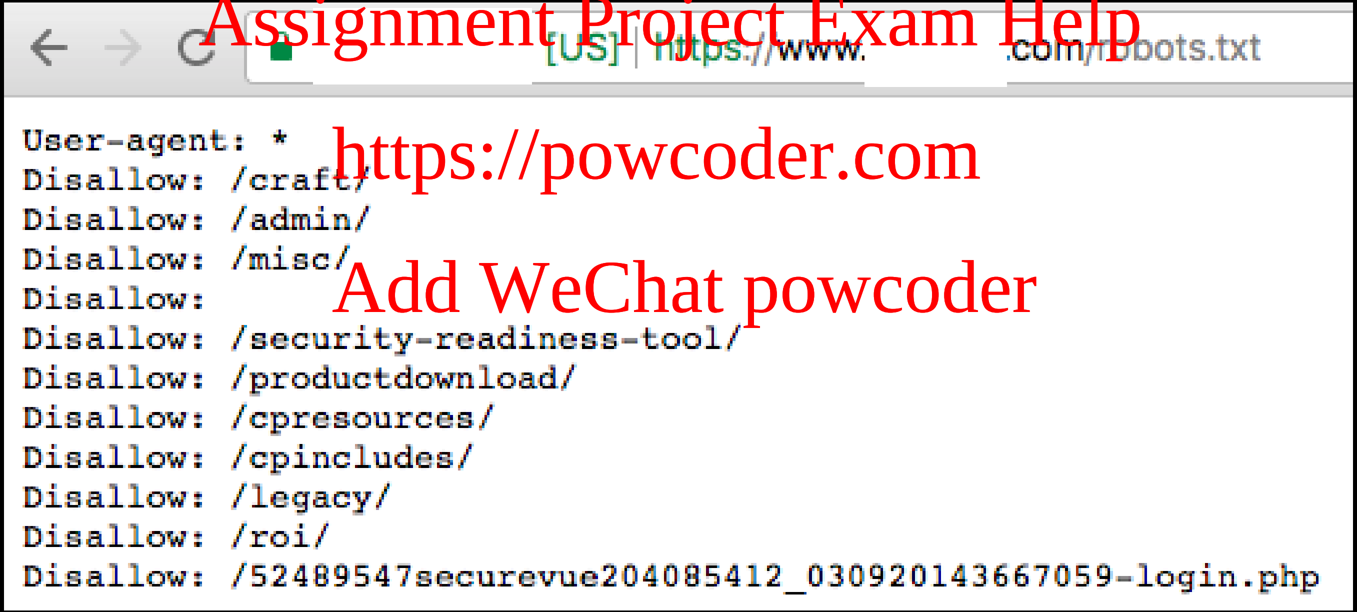
mobile applications

analytics

etc etc…

ATTACKING ACCESS CONTROL

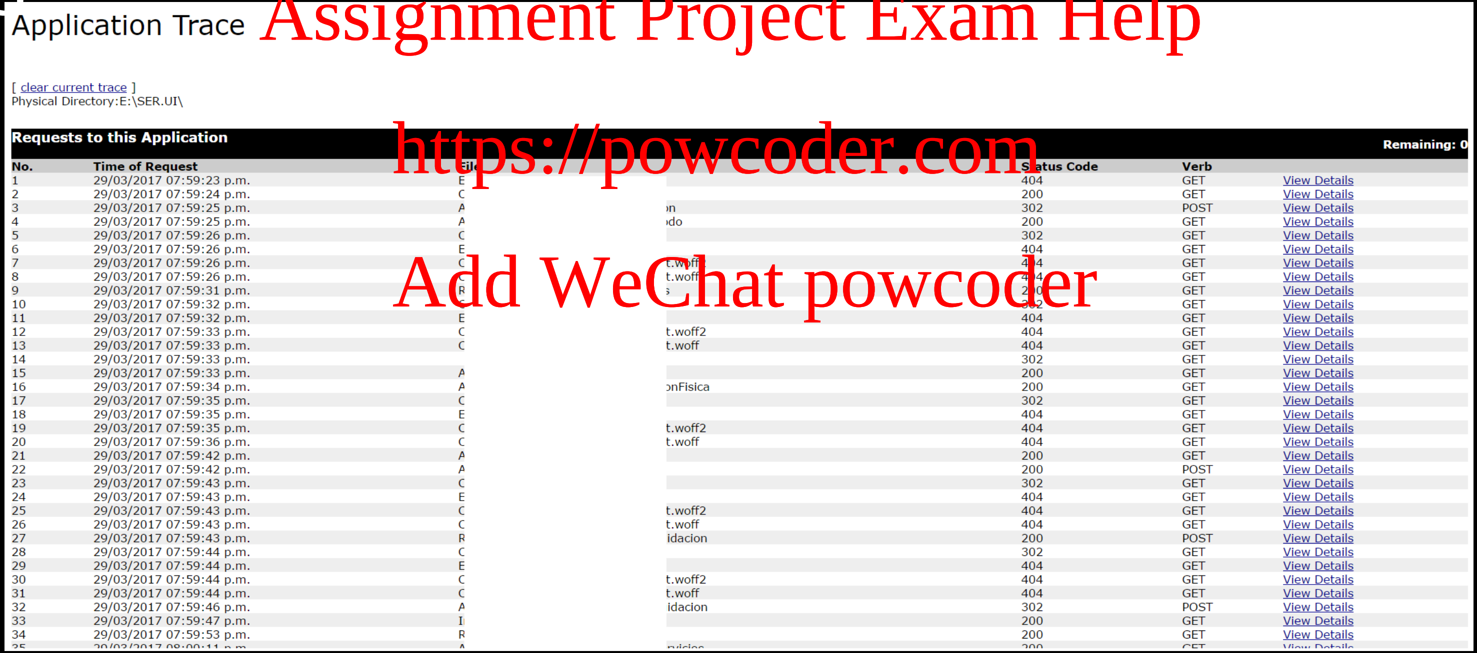
METHOD 1.5: ROBOTS.TXT



```
User-agent: *
Disallow: /craft/
Disallow: /admin/
Disallow: /misc/
Disallow:
Disallow: /security-readiness-tool/
Disallow: /productdownload/
Disallow: /cpresources/
Disallow: /cpincludes/
Disallow: /legacy/
Disallow: /roi/
Disallow: /52489547securevue204085412_030920143667059-login.php
```

ATTACKING ACCESS CONTROL

METHOD 2: COPY LEGITIMATE USERS



Application Trace

[clear current trace]
Physical Directory:E:\SER.UI\

Requests to this Application **Remaining: 0**

No.	Time of Request	File	Status Code	Verb	
1	29/03/2017 07:59:23 p.m.	E [redacted]	404	GET	View Details
2	29/03/2017 07:59:24 p.m.	C [redacted]	200	GET	View Details
3	29/03/2017 07:59:25 p.m.	A [redacted] on	302	POST	View Details
4	29/03/2017 07:59:25 p.m.	A [redacted] odo	200	GET	View Details
5	29/03/2017 07:59:26 p.m.	C [redacted]	302	GET	View Details
6	29/03/2017 07:59:26 p.m.	E [redacted]	404	GET	View Details
7	29/03/2017 07:59:26 p.m.	C [redacted] t.woff2	404	GET	View Details
8	29/03/2017 07:59:26 p.m.	C [redacted] t.woff	404	GET	View Details
9	29/03/2017 07:59:31 p.m.	R [redacted] s	200	GET	View Details
10	29/03/2017 07:59:32 p.m.	C [redacted]	302	GET	View Details
11	29/03/2017 07:59:32 p.m.	E [redacted]	404	GET	View Details
12	29/03/2017 07:59:33 p.m.	C [redacted] t.woff2	404	GET	View Details
13	29/03/2017 07:59:33 p.m.	C [redacted] t.woff	404	GET	View Details
14	29/03/2017 07:59:33 p.m.	[redacted]	302	GET	View Details
15	29/03/2017 07:59:33 p.m.	A [redacted]	200	GET	View Details
16	29/03/2017 07:59:34 p.m.	A [redacted] onFisica	200	GET	View Details
17	29/03/2017 07:59:35 p.m.	C [redacted]	302	GET	View Details
18	29/03/2017 07:59:35 p.m.	E [redacted]	404	GET	View Details
19	29/03/2017 07:59:35 p.m.	C [redacted] t.woff2	404	GET	View Details
20	29/03/2017 07:59:36 p.m.	C [redacted] t.woff	404	GET	View Details
21	29/03/2017 07:59:42 p.m.	A [redacted]	200	GET	View Details
22	29/03/2017 07:59:42 p.m.	A [redacted]	200	POST	View Details
23	29/03/2017 07:59:43 p.m.	C [redacted]	302	GET	View Details
24	29/03/2017 07:59:43 p.m.	E [redacted]	404	GET	View Details
25	29/03/2017 07:59:43 p.m.	C [redacted] t.woff2	404	GET	View Details
26	29/03/2017 07:59:43 p.m.	C [redacted] t.woff	404	GET	View Details
27	29/03/2017 07:59:43 p.m.	R [redacted] idacion	200	POST	View Details
28	29/03/2017 07:59:44 p.m.	C [redacted]	302	GET	View Details
29	29/03/2017 07:59:44 p.m.	E [redacted]	404	GET	View Details
30	29/03/2017 07:59:44 p.m.	C [redacted] t.woff2	404	GET	View Details
31	29/03/2017 07:59:44 p.m.	C [redacted] t.woff	404	GET	View Details
32	29/03/2017 07:59:46 p.m.	A [redacted] idacion	302	POST	View Details
33	29/03/2017 07:59:47 p.m.	I [redacted]	200	GET	View Details
34	29/03/2017 07:59:53 p.m.	R [redacted]	200	GET	View Details
35	29/03/2017 08:00:11 p.m.	A [redacted] rvicios	200	GET	View Details

ATTACKING ACCESS CONTROL

METHOD 3: ACTUAL TESTING

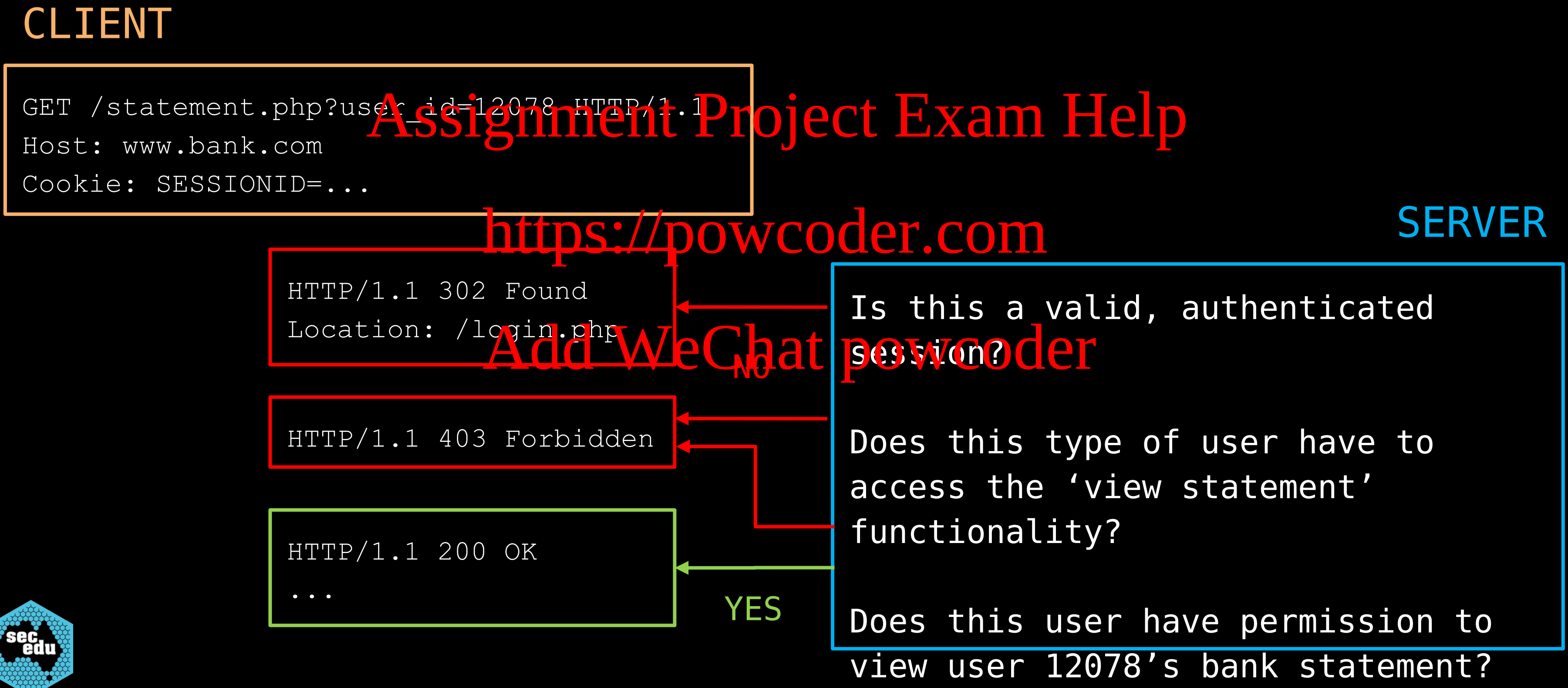
How does the application know what user role I am?
Are checks applied consistently throughout the application?
When a check fails, what happens?

What aspects of this information can I control?
Can I impersonate another user, or role?
What about content which has zero access control?

CYBER

SUCCESS

TOWARDS BETTER ACCESS CONTROL



```
GET /statement.php?user_id=12078 HTTP/1.1
Host: www.bank.com
Cookie: SESSIONID=...
```

```
HTTP/1.1 302 Found
Location: /login.php
```

```
HTTP/1.1 403 Forbidden
```

```
HTTP/1.1 200 OK
...
```

Is this a valid, authenticated session?

Does this type of user have to access the 'view statement' functionality?

Does this user have permission to view user 12078's bank statement?

CLIENT-SIDE ACCESS CONTROL

HTTPS Downgrade

Insecure Sites

XSS

User Identity & Trust

Plugins

Cookies

Content Integrity

HTTP/HTTPS

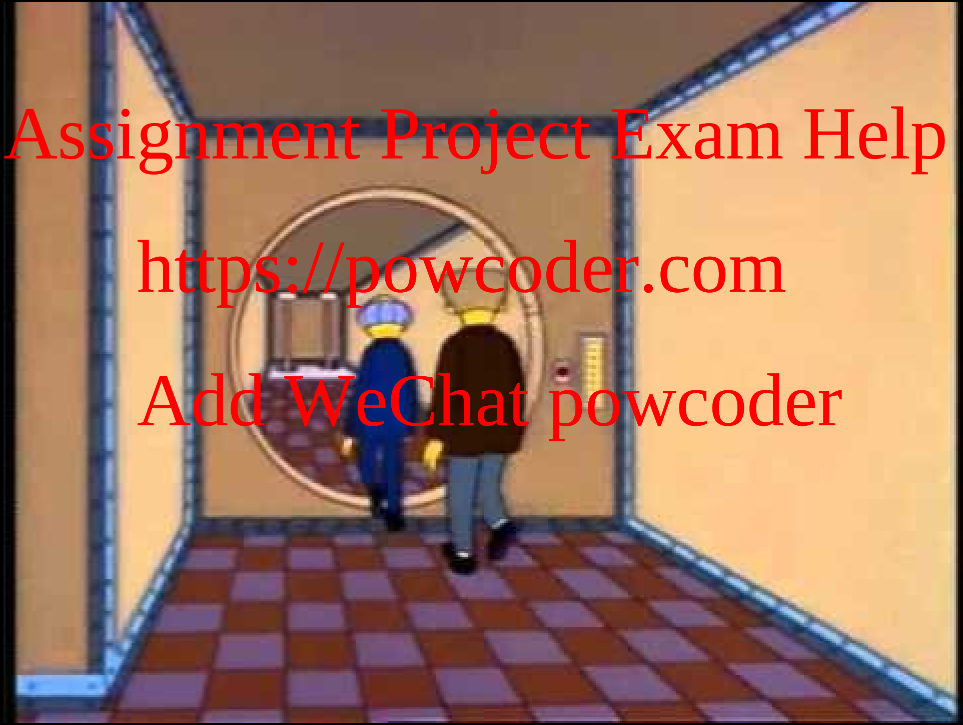
Mixed Content

Phishing Websites

Compromises of Privacy
(e.g. friends list)

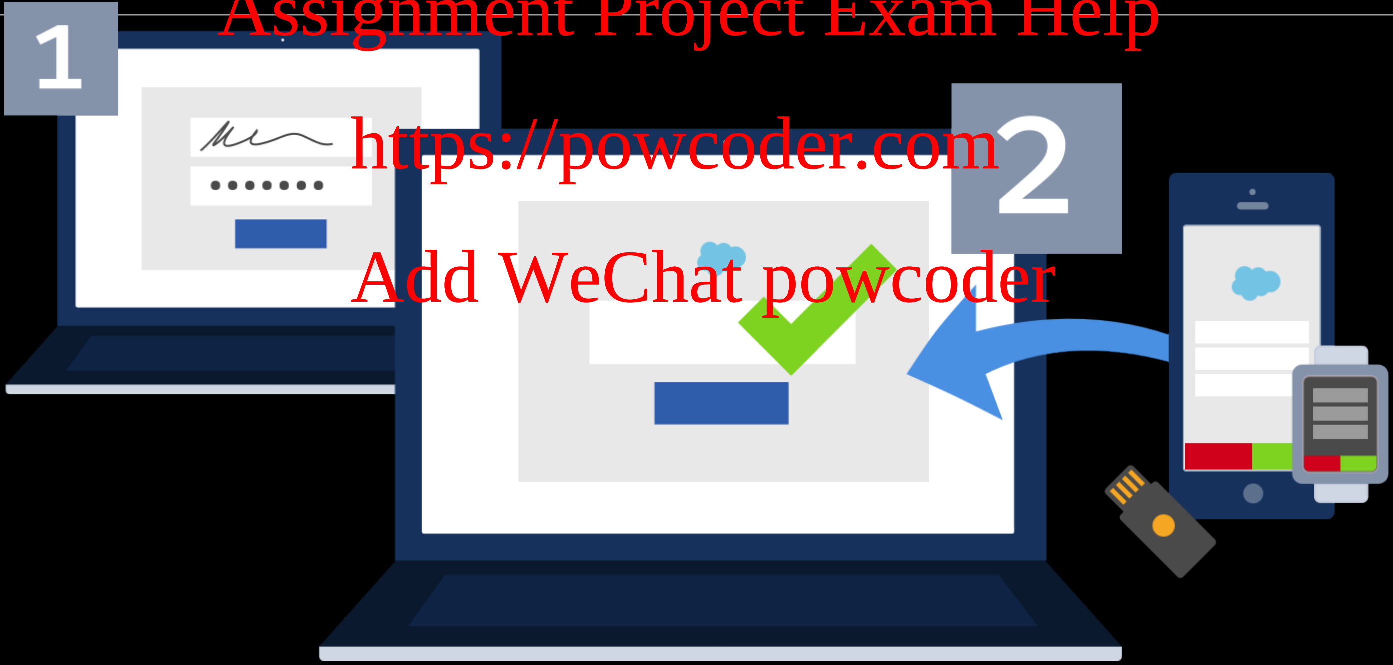
Phishing / CSRF

THE WEAKEST LINK



HOW TO PROTECT?

2FA



2FA

SMS

Mobile app

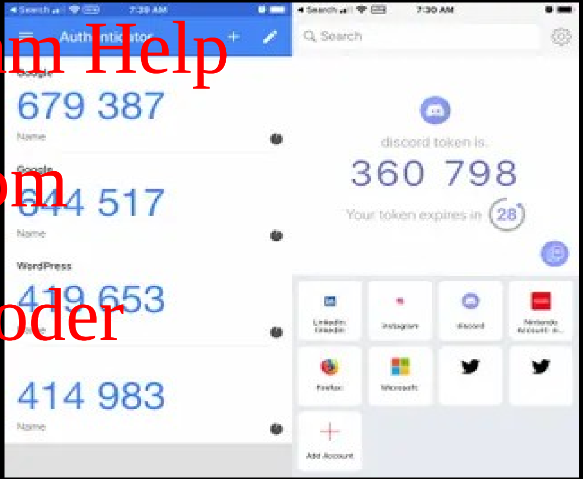
Key generator

Fingerprint

Channel-based

Location-based

Biometric



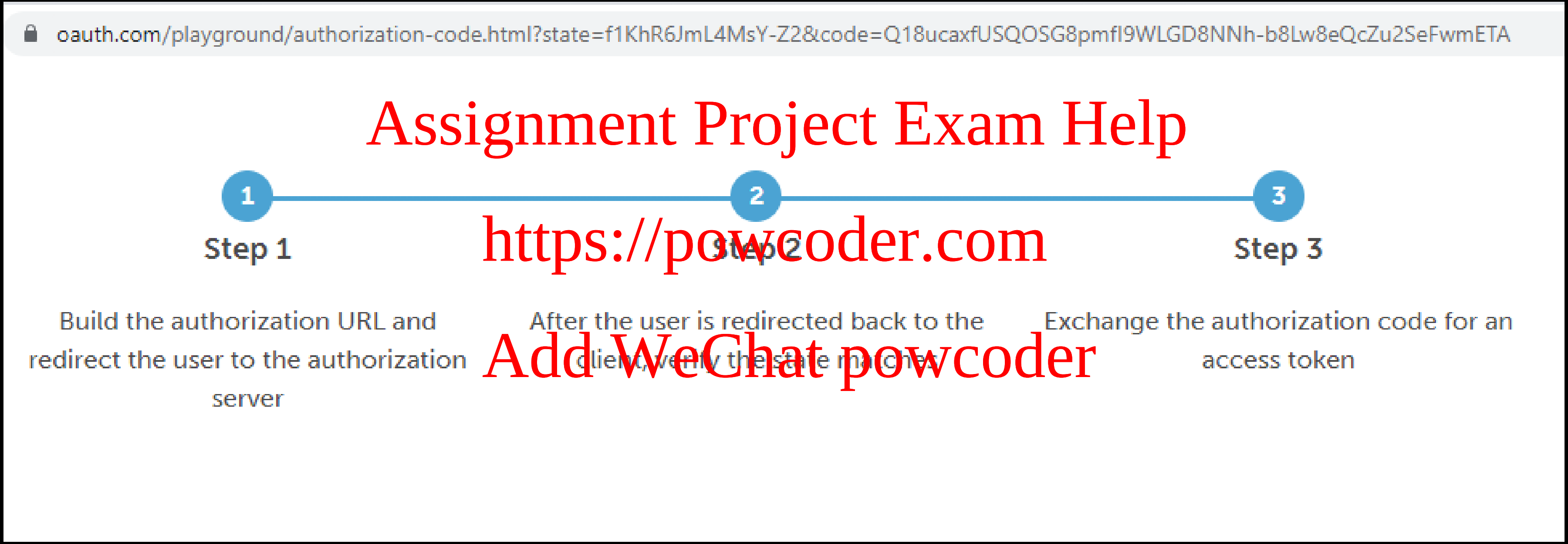
OAUTH

- Open id – Authentication Protocol
- Oauth – Authorization Framework(Oauth 1.0 & 2.0)
- OpenID Connect – built on top of Oauth 2.0

4 Types of Oauth grants:

- Authorization Grant
- Implicit Grant
- Resource Owner Credentials
- Client Credentials

OAUTH



oauth.com/playground/authorization-code.html?state=f1KhR6JmL4MsY-Z2&code=Q18ucaxfUSQOSG8pmfl9WLGD8NNh-b8Lw8eQcZu2SeFwmETA

1 Step 1

Build the authorization URL and redirect the user to the authorization server

2 Step 2

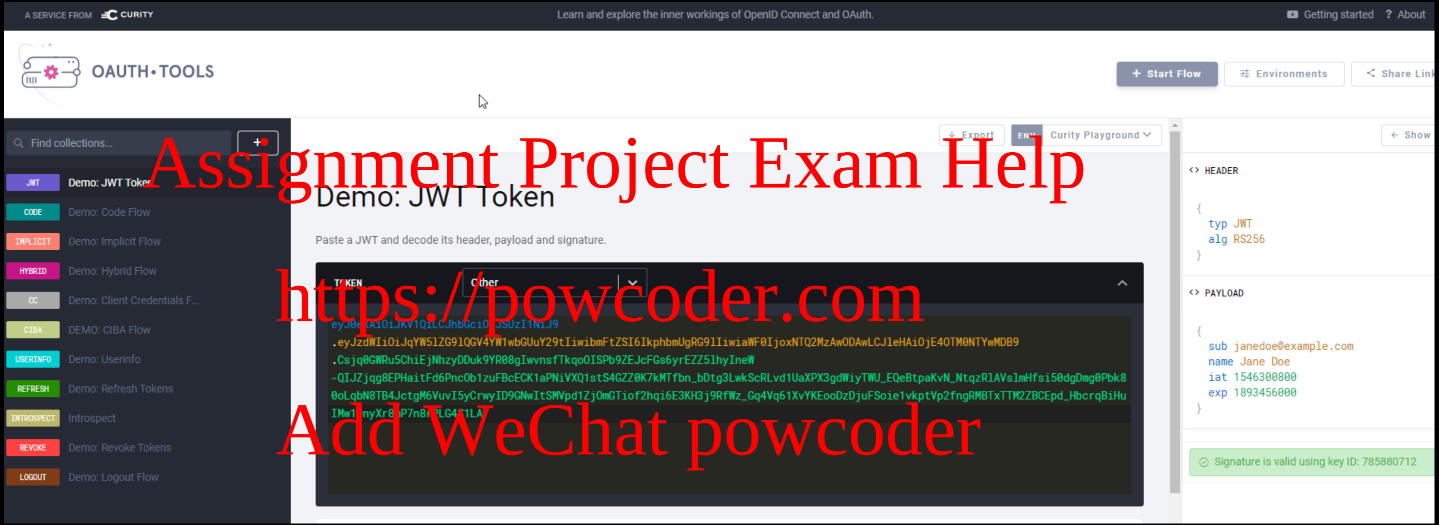
After the user is redirected back to the client, verify the state matches

3 Step 3

Exchange the authorization code for an access token

https://www.oauth.com/playground/index.html

OAUTH



Demo: JWT Token

Paste a JWT and decode its header, payload and signature.

TOKEN Other

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9
.eyJzdWIiOiJqYW5lZG9lQGV4YW1wbGUuY29tIiwibmFtZSI6IkphbmUgRG9lIiwiaWF0IjoxNTQ2MzAwODAwLCJleHAiOjE4OTM0NTYwMDB9
.Csjq0GWRu5ChiEjNhzyDDuk9YR08gIwvnsfTkqoOISPb9ZEJcFGs6yrEZZ5lhyIneW
-QIJZjqg8EPHaitFd6PncOb1zuFBcECK1aPNiVXQ1stS4GZZ0K7kMTfbn_bDtg3LwkScRLvd1UaXPX3gdWiyTWU_EQeBtpaKvN_NtqzRlAVslmHfsi50dgDmg0Pbk8
0oLqbN8TB4JctgM6VuvI5yCrwyID9GNwItSMVpd1ZjOmGTiof2hqi6E3KH3j9RfWz_Gq4Vq61XvYKEooDzDjuFSoie1vkptVp2fngRMBTxTTM2ZBCEpd_HbcrqBiHu
IMw1[illegible]nyXr8[illegible]P7n8[illegible]LG4[illegible]1LA
```

HEADER

```
{
  typ JWT
  alg RS256
}
```

PAYLOAD

```
{
  sub janedoe@example.com
  name Jane Doe
  iat 1546300800
  exp 1893456000
}
```

Signature is valid using key ID: 785880712

https://oauth.tools/

[DEFENSIVE] SECURING YOUR SESSIONS

- Minimize your attack surface
  - Your session should be managed on the server side where possible, using a single session token.
- Mostly mechanical fixes
  - Don’t let people steal your tokens (URLs, HTTP, etc)
  - Don’t let people re-use tokens (expire them properly, log out)
  - Don’t let people generate tokens (secure PRNG, avoid rolling your own crypto, don’t allow users to supply tokens).
- Attention to detail is key.

READING MATERIAL (REFERENCE)

- OAuth2 Simplified https://aaronparecki.com/oauth-2-simplified/
- What the heck is Oauth https://stormpath.com/blog/what-the-heck-is-oauth
- How Anand hacked Tinder https://medium.com/free-code-camp/hacking-tinder-accounts-using-facebook-accountkit-d5cc813340d1

WEEK 2-3 ASSESSMENT

- Hash collisions
- Don't forget about <u>automation</u>!
- Don't forget about writing a <u>report</u>!!

Please call out if you get stuck.
Support one another, your tutors are here to help!

THANKS FOR LISTENING TO US RANT!

questions? slack / email / come talk to us