

PRACTICUM OF ATTACK AND DEFENSE OF NETWORK SECURITY

LECTURE-06 : WEB SECURITY

Spring 2021

Outline

- OWASP TOP 10
- Web vulnerability
 - CSRF, XSS, CRLF injection, SSRF
- Tool
 - Burp, DirBuster
- Privacy & Anonymity
 - VPN, DNS Over HTTPS, TOR

OWASP TOP 10

Open Web Application Security Project

OWASP Top 10

- Most critical security risks to web applications.
- 2003, 2004, 2007, 2010, 2013, and 2017

OWASP Top 10 - 2013	→	OWASP Top 10 - 2017
A1 – Injection	→	A1:2017-Injection
A2 – Broken Authentication and Session Management	→	A2:2017-Broken Authentication
A3 – Cross-Site Scripting (XSS)	↳	A3:2017-Sensitive Data Exposure
A4 – Insecure Direct Object References [Merged+A7]	↳	A4:2017-XML External Entities (XXE) [NEW]
A5 – Security Misconfiguration	↳	A5:2017-Broken Access Control [Merged]
A6 – Sensitive Data Exposure	↗	A6:2017-Security Misconfiguration
A7 – Missing Function Level Access Contr [Merged+A4]	↳	A7:2017-Cross-Site Scripting (XSS)
A8 – Cross-Site Request Forgery (CSRF)	☒	A8:2017-Insecure Deserialization [NEW, Community]
A9 – Using Components with Known Vulnerabilities	→	A9:2017-Using Components with Known Vulnerabilities
A10 – Unvalidated Redirects and Forwards	☒	A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

OWASP Top 10 2021 proposal

- Proposal 1. Add SSRF as a new category.
- Proposal 2. Merge XXE and Insecure Deserialization
- Proposal 3. Introduce Overall Risk Score

OWASP Top 10 2021

Statistics-based proposal.

- <https://lab.wallarm.com/owasp-top-10-2021-proposal-based-on-a-statistical-data/>
- Overall Risk = Avg. CVSS x Amount of Bulletins (<https://vulners.com/>)

OWASP Top 10 2017		change	OWASP Top 10 2021 proposal	
A1	Injections	as is	A1	Injections
A2	Broken Authentication	as is	A2	Broken Authentication
A3	Sensitive Data Exposure	down 1	A3	Cross-Site Scripting (XSS)
A4	XML eXternal Entities (XXE)	down 1 + A8	A4	Sensitive Data Exposure
A5	Broken Access Control	down 1	A5	Insecure Deserialization (merged with XXE)
A6	Security Misconfiguration	down 4	A6	Broken Access Control
A7	Cross-Site Scripting (XSS)	up 4	A7	Insufficient Logging & Monitoring
A8	Insecure Deserialization	up 3 + A4	A8	NEW: Server Side Request Forgery (SSRF)
A9	Known Vulnerabilities	as is	A9	Known Vulnerabilities
A10	Insufficient Logging & Monitoring	up 3	A10	Security Misconfiguration

Avg. CVSS	# of bulletins	Overall score
4.83	34061	164514.63
4.08	13735	56038.8
0.1	433353	43335.3
3.55	5990	21264.5
5.33	2985	15910.05
0.72	16967	12216.24
3.35	2309	7735.15
3.8	1139	4328.2
5.38	376	2022.88
2.27	480	1089.6

WEB VULNERABILITY

CSRF

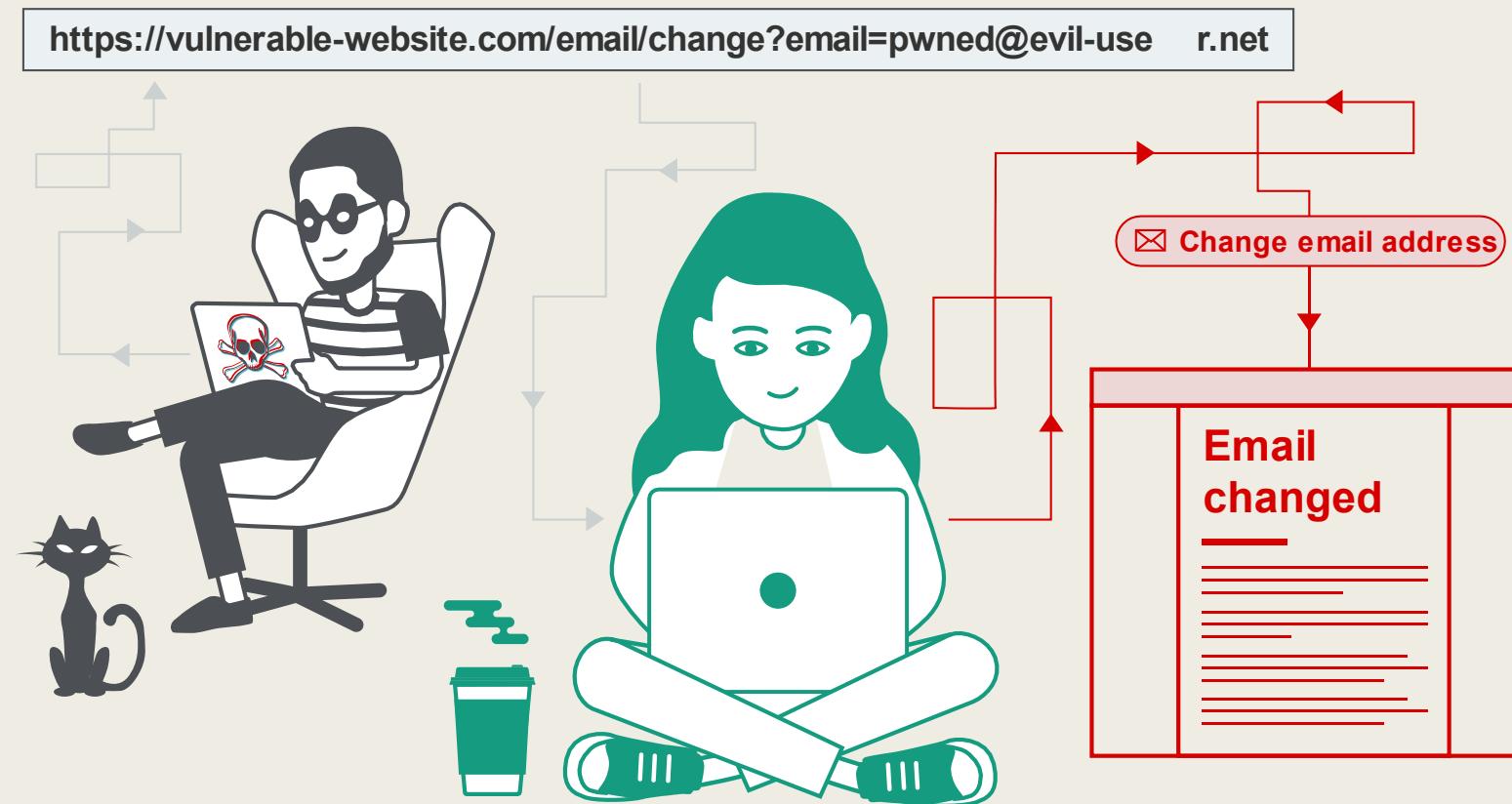
XSS

CRLF injection

SSRF

CSRF (Cross-Site Request Forgery)

- 跨站請求偽造
 - 攻擊者讓用戶在當前已登入的網站上，執行非本意操作

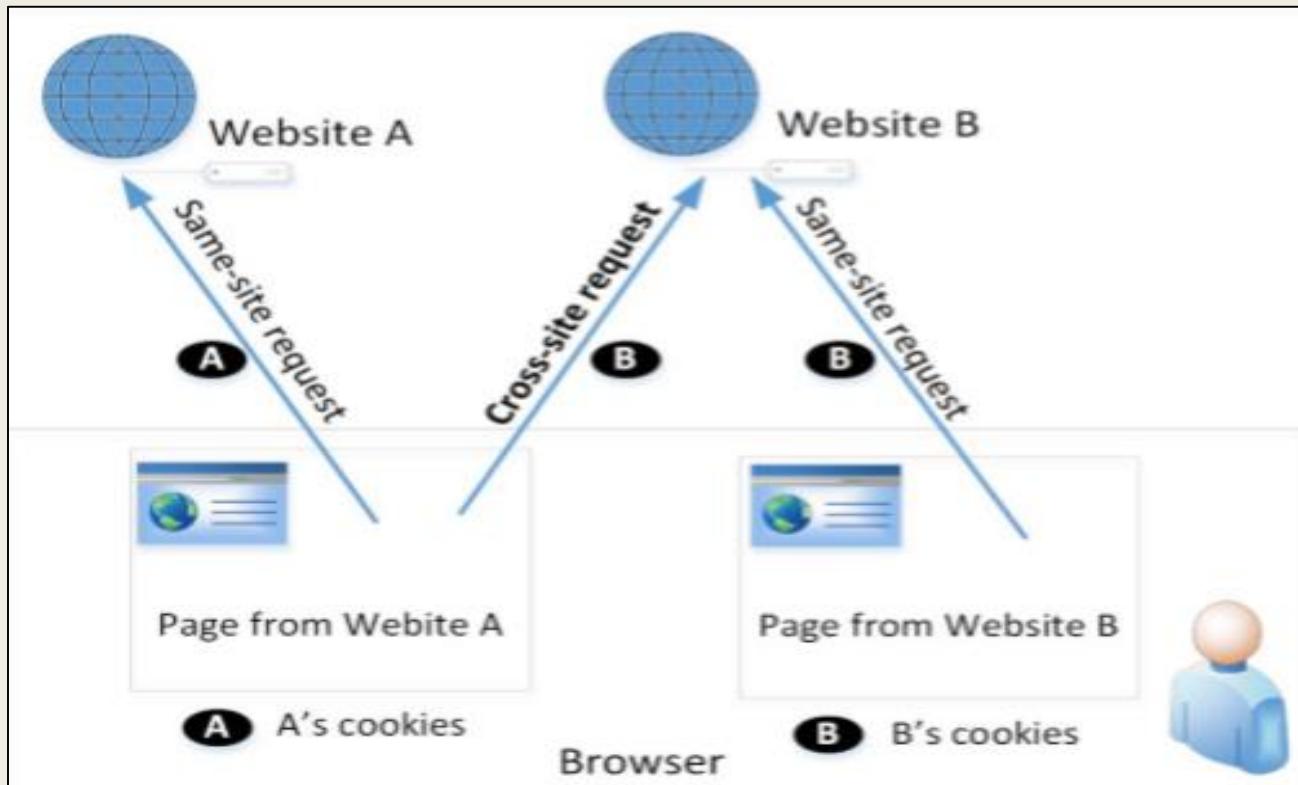


HTML form

```
<h1>Change email</h1>
<section>
  <form class="login-form" action="/email/change-email" method="POST">
    <label>Email</label>
    <input required type="email" name="email" value="">
    <button class='button' type='submit'> Update email </button>
  </form>
</section>
```

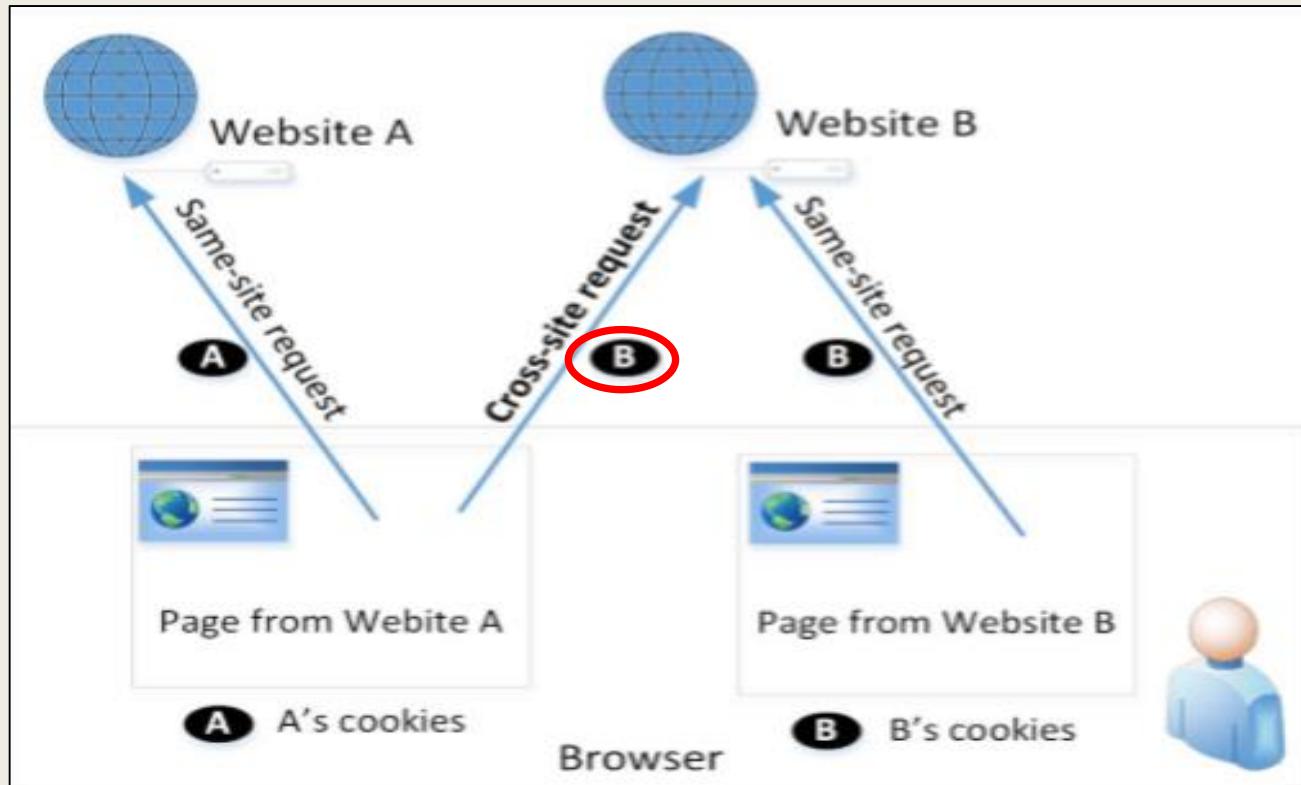
The screenshot shows a user interface for changing an email address. At the top, the title "Change email" is displayed in a large blue font. Below the title is a light gray input field labeled "Email" with a white text area for entering the new email address. At the bottom of the interface is a green button with the text "Update email" in white.

Definition: Cross-Site Requests & Same-Site Request



- same-site request:
一個網站的頁面向自己發送請求
`action="/email/change-email"`
- cross-site request:
A 網站的頁面向 B 網站發送請求
`action="http://www.example.com/action_post.php" method="post">`

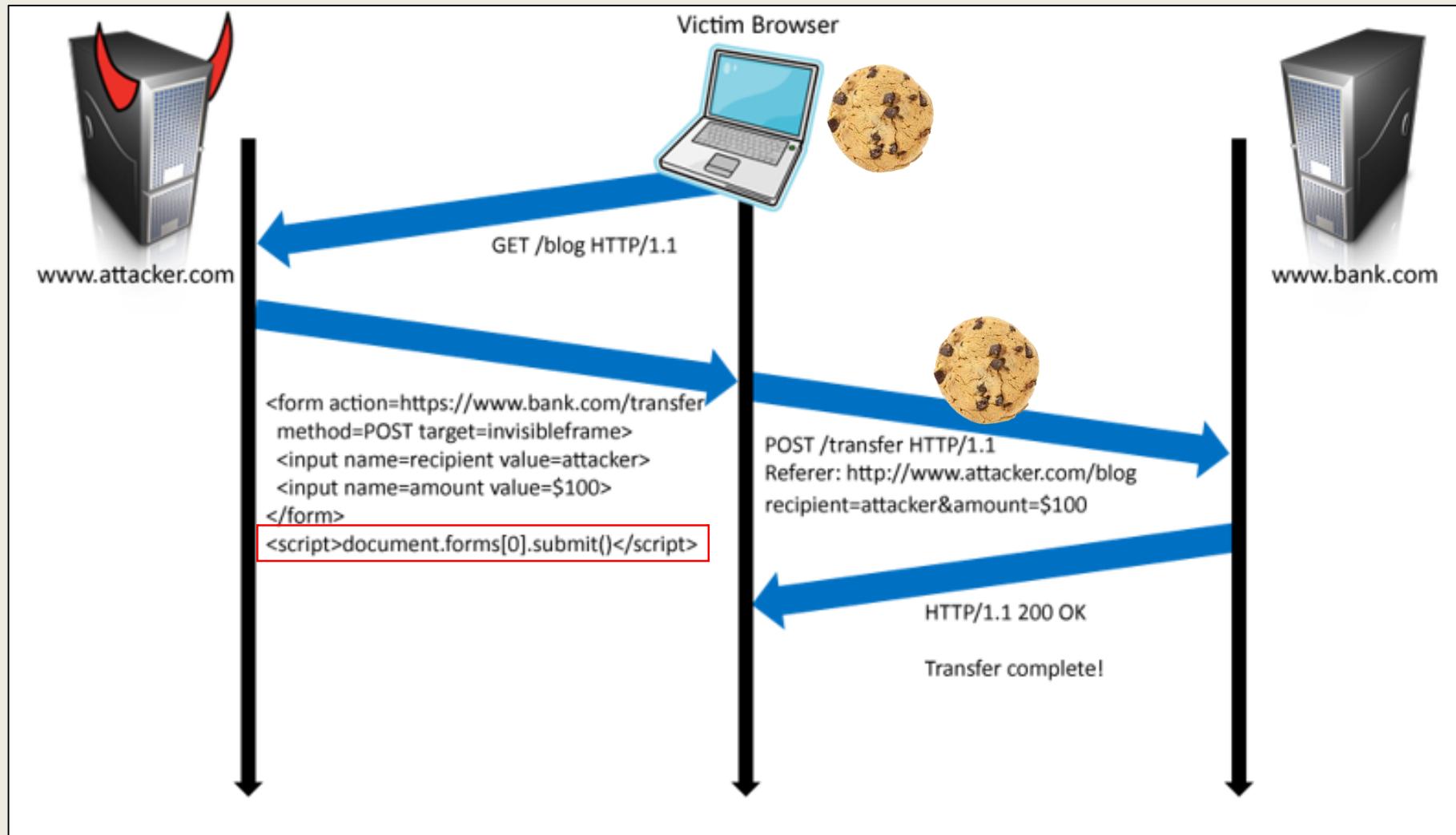
Cookies in Cross-Site Request



- 當 A.com 的頁面向 A.com 發送請求時，瀏覽器會將 A.com 的 cookies 一併送出
- 當 A.com 的頁面向 B.com 發送請求時，瀏覽器也會將 B.com 的 cookies 送出 (如果cookie沒有 SameSite屬性)

`SameSite=<samesite-value>`

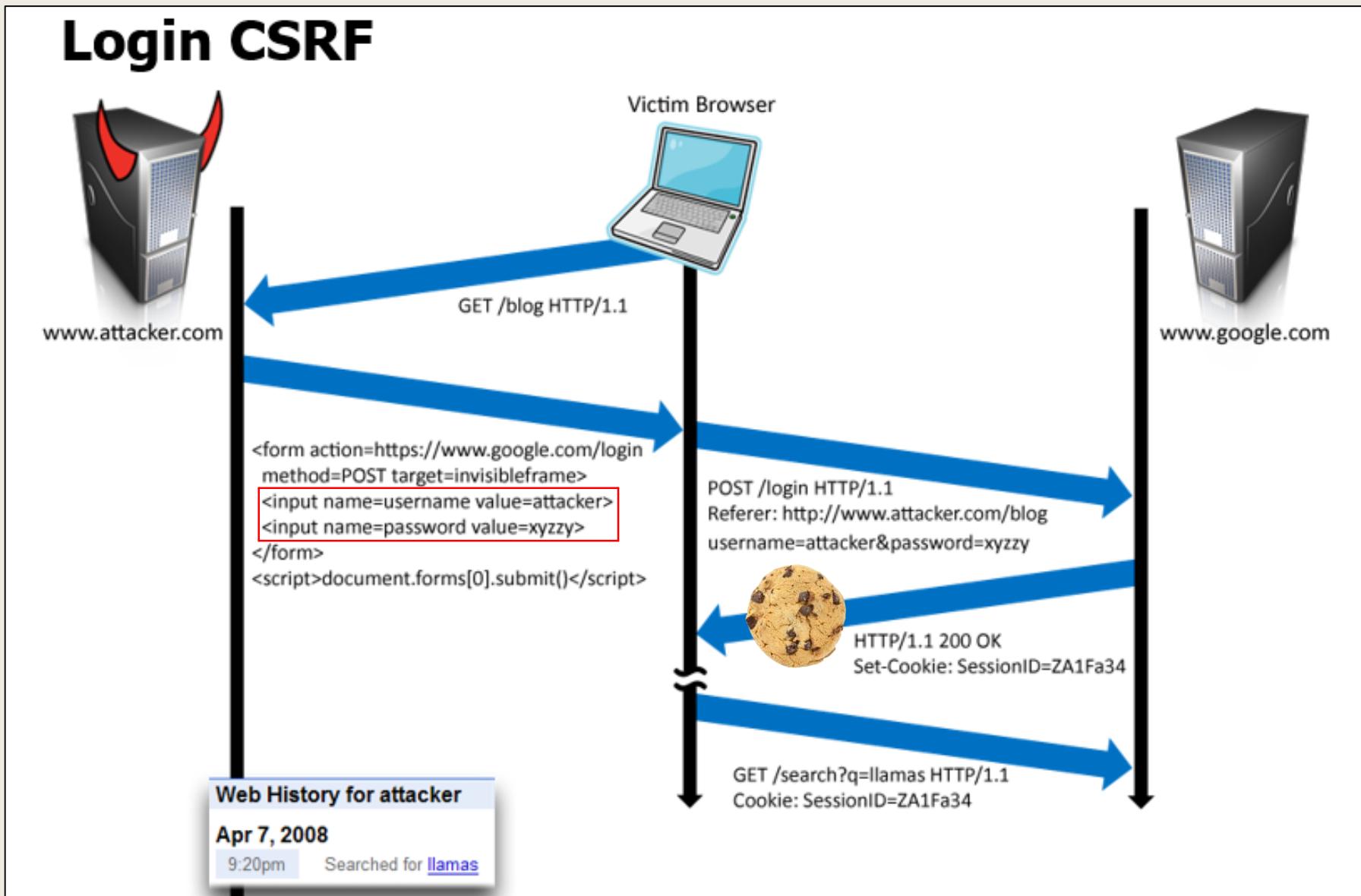
CSRF Attack Flow



CSRF Login Attack

- 攻擊者偽造一個登入請求，使用的是攻擊者自己的帳號密碼
- 目的：攻擊者之後可以登入自己的帳號，查看歷史紀錄，得知受害者做了那些操作或留下哪些資訊

CSRF Login Attack



Trigger request in CSRF

- JavaScript：
攻擊者可以用 JavaScript 送出請求

```
document.getElementsByName("myForm")[0].submit();
```

- 和 <iframe> tag：
HTML 的 和 <iframe> tag 會對 src attribute 所指定的 URL 送出 GET request

```
  
  
<iframe  
    src="http://www.bank32.com/transfer.php?to=3220&amount=500">  
</iframe>
```

CSRF by GET request

- When you click “Add Friend”

```
http://www.csrflabelgg.com/action/friends/add?friend=40&__elgg_ts=1532750193&__elgg_token=..
```

- The tag will trigger an HTTP GET request.

```
<html>
<head>
<title>
Malicious Web of CSRF Attack
</title>
</head>
<body>
<H1>Congratulation! You have won a grand prize!</H1>



<H2>WebMaster: James Yu</H2>
</body>
</html>
```

CSRF Countermeasures

- 幫助 server 端分辨 same-site request / cross-site request
 - 1. Referer header (browser's help)
 - 2. Same-site cookie (browser's help)
 - 3. Secret token (the server helps itself)

1. HTTP Referer Header

- HTTP header field identifies the address of the web page from which the request is generated.

```
▼ Hypertext Transfer Protocol
▶ GET /action/friends/add?friend=40 HTTP/1.1\r\n
  Host: www.csrflabelgg.com\r\n
  User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:
  Accept: image/png,image/*;q=0.8,*/*;q=0.5\r\n
  Accept-Language: en-US,en;q=0.5\r\n
  Accept-Encoding: gzip, deflate\r\n
  Referer: http://www.csrflabattacker.com/\r\n
  Cookie: Elgg=tjc0go05bq3cc7npsua3asttu0\r\n
  Connection: keep-alive\r\n
  \r\n
```

1. HTTP Referer Header

- Server 可以確認這個 request 是否源自一個信任的頁面
- 這種方式只能用在暫時性防禦，或者是已知使用者環境的情況下。
 - 有些 proxy 為了隱私，會省略 Referer Header
 - 檢查規則容易遺漏或規避，而有些軟體或瀏覽器可以控制傳送或停用 Referer
 - 一個 request 若缺少 Referer header，Server 必須拒絕這個請求
- 可能存在其他漏洞可以達成 Referer spoofing
 - E.g., CRLF injection
 - 瀏覽器本身的漏洞

2. Same-Site Cookies

- 透過 “Set-Cookie” headers
 - ***Set-Cookie: NAME=VALUE; expires=DATE; path=PATH; domain=DOMAIN_NAME; Secure; HttpOnly; SameSite=<samesite-value>***

SameSite=<samesite-value> | Optional

- **Strict**: The browser sends the cookie **only for same-site requests** (that is, requests originating from the same site that set the cookie). If the request originated from a different URL than the current one, no cookies with the **SameSite=Strict** attribute are sent.
- **Lax**: The cookie is **withheld on cross-site subrequests** such as calls to load images or frames, but is sent when a user navigates to the URL from an external site, such as by following a link.
- **None**: The browser sends the cookie with both cross-site and same-site requests.

3. Secret Token

- a) Server 端生成的 CSRF token
- b) Server 端生成的 Double Submit Cookie
- c) Client 端生成的 Double Submit Cookie

3. Secret Token

a) Server 端生成的 CSRF token

1. 在 form 裡加一個欄位: name='csrftoken' value='<亂碼>'，並將<亂碼>存在 server
 2. 收到 request 時比對 form 裡的<亂碼>是否等於 server 端的 <亂碼>
-
- 產生：Server
 - 儲存：Server
 - 漏洞：若 csrftoken 不是亂碼，攻擊者可以先發一個 request 取得 csrftoken

3. Secret Token

b) Server 端生成的 Double Submit Cookie

1. 在 form 裡加 CSRF token，同時 set 一個 cookie 的值等於 CSRF token
 2. 收到 request 時比對 form 裡的 CSRF token 是否等於 cookie 的值
- 這方法利用的是攻擊者無法取得 cookie 的值，因此偽造的 form 中的 CSRF token 不會和 cookie 裡的值一樣
 - 產生：Server
 - 儲存：Client
 - 漏洞：攻擊者如果掌握了你底下任何一個 subdomain，就可以幫你來寫 cookie

Weak Integrity of Cookie

- Cookies 對 subdomain 並不具有完整性
- 舉例來說，foo.example.com 可以對 example.com 設置 cookie，而這個有可能把 bar.example.com 對 example.com 設置的 cookie 紿蓋掉
- 最糟的情況下，當 bar.example.com 收到這個 cookie 時，區分不出是自己設置的還是別人設置的。foo.example.com 就可以利用這個特性來攻擊 bar.example.com。

3. Secret Token

c) Client 端生成的 Double Submit Cookie

- 由前端在 form 裡加 CSRF token，同時 set 一個 cookie 的值等於 CSRF token

```
document.cookie = "username=John Doe; expires=Thu, 18 Dec 2013 12:00:00 UTC";
```

- 收到 request 時比對 form 裡的 CSRF token 是否等於 cookie 的值

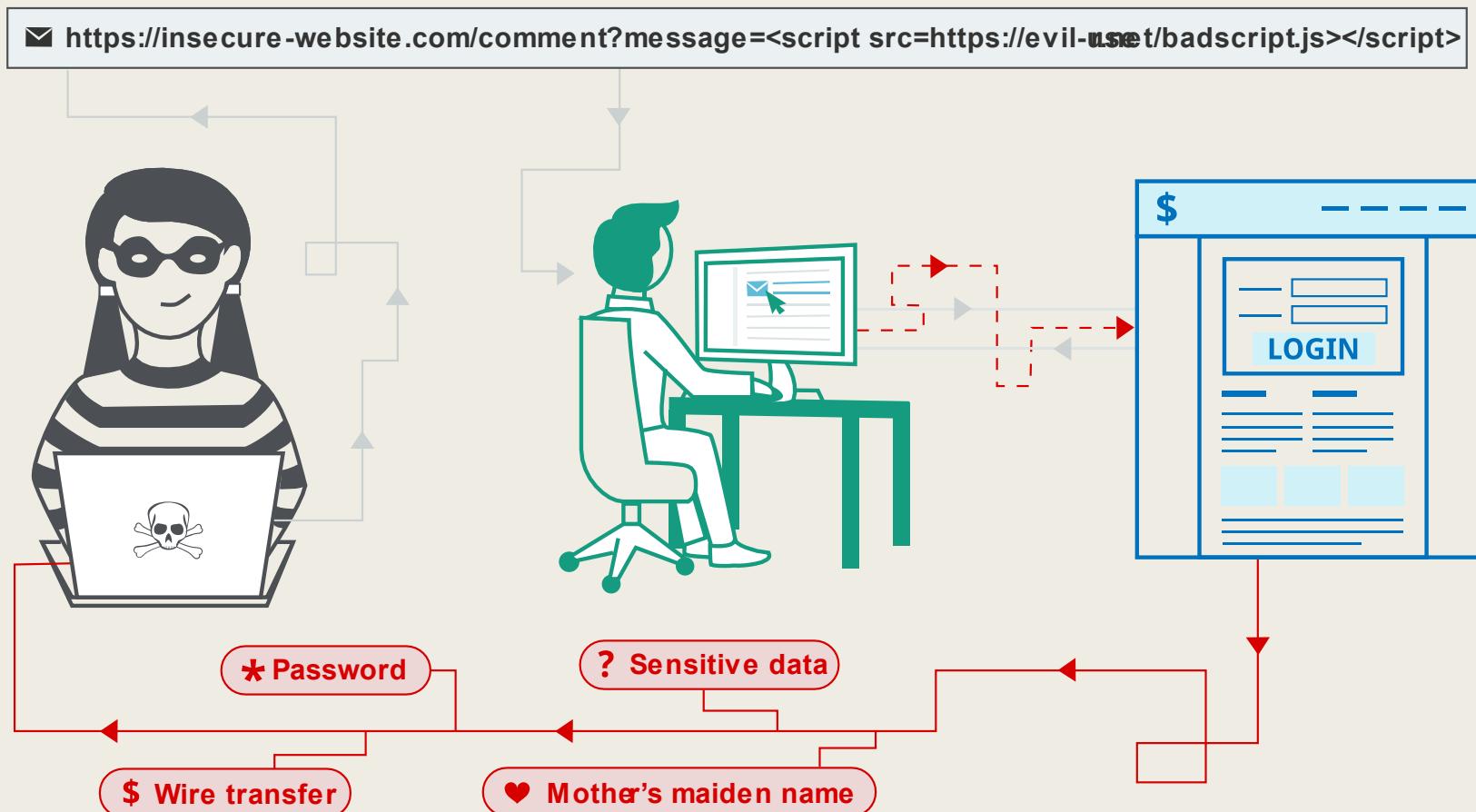
- 單頁應用 (single-page application) 在拿取 CSRF token 會有困難，可以改成 Client 端生成
- 產生：Client
- 儲存：Client

Summary of CSRF

- Cross-site requests vs. Same-site requests
- How does a hacker launch CSRF attack?
- The fundamental cause of the CSRF vulnerability
- Countermeasures against CSRF attack

XSS (Cross-Site Scripting)

- 跨網站指令碼
 - 攻擊者將惡意程式碼注入到受害網頁上，使其他用戶在觀看網頁時執行。



What hackers can do with XSS attacks?

- Forge user requests (even same-site requests)
- Steal cookies ([HttpOnly](#))
- Man-in-the-browser
 - Get form values / HTML contents
 - Tabnabbing (一種釣魚攻擊的手法)

Types of XSS Attacks

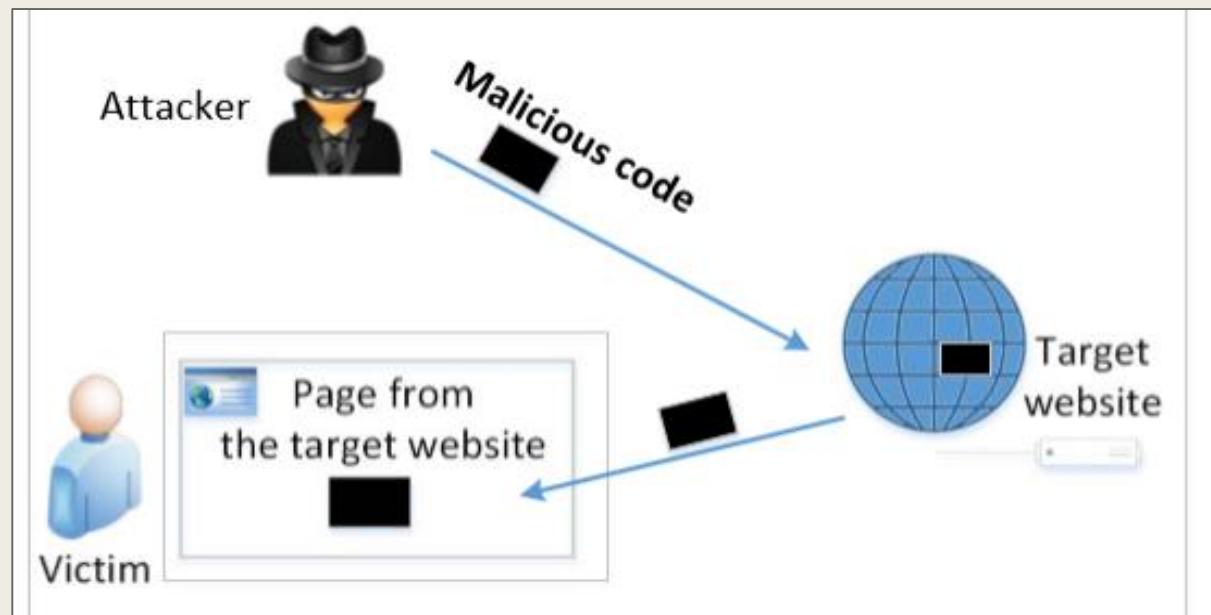
- Stored XSS Attack
- Reflected XSS Attack

Stored XSS Attack

1. 攻擊者將惡意代碼提交到目標網站的資料庫中。
2. 用戶瀏覽受害網站時，Server 將惡意代碼從資料庫取出，拼接在 HTML 中返回給用戶瀏覽器。

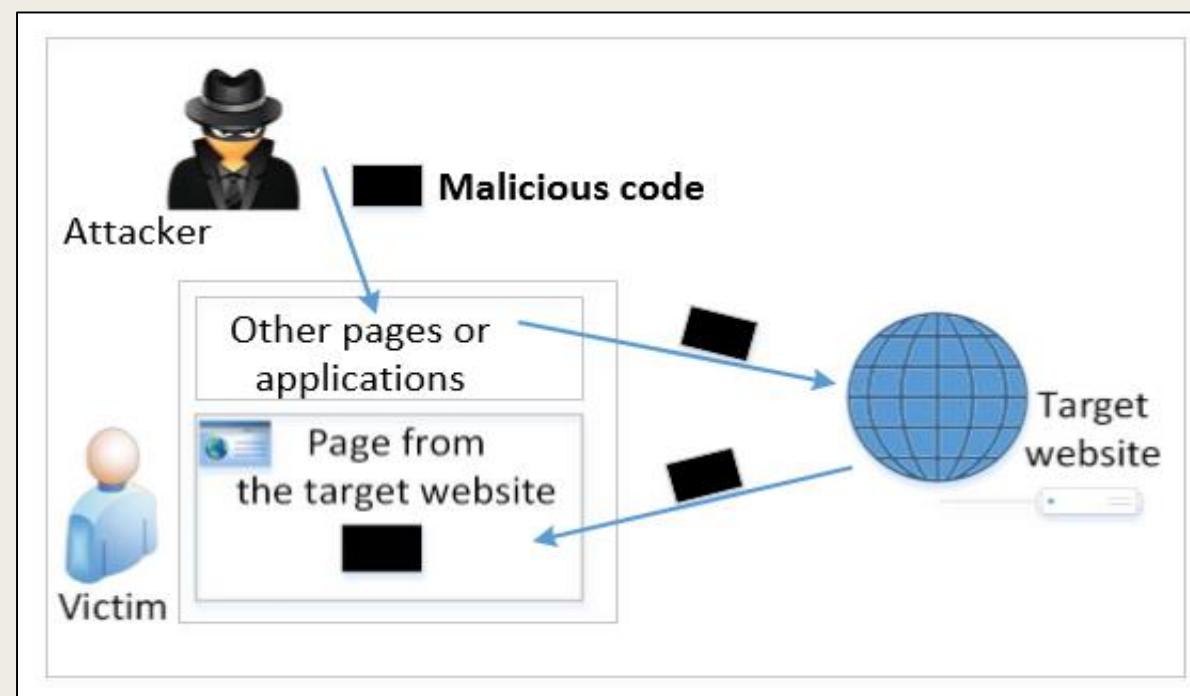
- Example :

- 社群網站個人頁面
- 留言



Reflected XSS Attack

1. 攻擊者構造出特殊的 URL，其中包含惡意代碼。
2. 用戶打開帶有惡意代碼的 URL 時，Server 將惡意代碼從 URL 中取出，拼接在 HTML 中返回給瀏覽器。



Example - Reflected XSS Attack

- 有個網站的搜尋使用 GET request
 - <http://www.example.com/search?input=word>
- 網站回應包含搜尋目標字串

你搜尋了 **XXX** !
- 攻擊者構造惡意連結
 - [http://www.example.com/search?input=<script>alert\(1\);</script>](http://www.example.com/search?input=<script>alert(1);</script>)

Example - Reflected XSS Attack CRLF injection (HTTP Response Splitting)

- The header of a HTTP response and its body are separated by two CRLF(\r\n\r\n) characters.

```
> Content-Length: 43\r\n
X-Cache: MISS from 172.19.134.2\r\n
X-Cache-Lookup: MISS from 172.19.134.2:3128\r\n
Via: 1.0 172.19.134.2:3128 (squid/2.6.STABLE14)\r\n
Connection: keep-alive\r\n
\r\n 回車換行 (Carriage-Return Line-Feed )
```

140	61	6c	69	76	65	0d	0a	0d	0a	47	49	46	38	39	61	01	alive	...	GIF89a
150	00	01	00	80	00	00	ff	ff	ff	00	00	00	21	f9	04	01	!
160	00	00	00	00	2c	00	00	00	00	01	00	01	00	00	02	02,
170	44	01	00	3b													D ..;		



登入您的帳戶

電子郵件地址

輸入您的電子郵件

下一步

重設密碼

或



使用 Google 登入

Example - Reflected XSS Attack CRLF injection (HTTP Response Splitting)

- Redirect users after successful login:
 - `http://localhost/login?page=http%3A%2F%2Flocalhost%2Findex`

- HTTP response:

HTTP/1.1 302 Moved Temporarily

Date: Tue, 17 Aug 2010 20:00:29 GMT

Server: Apache mod_fcgid/2.3.5 mod_auth_passthrough/2.1
mod_bwlimited/1.4 FrontPage/5.0.2.2635

Location: `http://localhost/index`

Example - Reflected XSS Attack CRLF injection (HTTP Response Splitting)

- Redirect users after successful login:
 - `http://localhost/login?page=http%3A%2F%2Flocalhost%2Fcheckout%0D%0A%0D%0A%3Cs cript%3Ealert%28%27hello%27%29%3C%2Fscript%3E`

- HTTP response:

```
HTTP/1.1 302 Moved Temporarily
Date: Tue, 17 Aug 2010 20:00:29 GMT
Server: Apache mod_fcgid/2.3.5 mod_auth_passthrough/2.1
mod_bwlimited/1.4 FrontPage/5.0.2.2635
Location: http://localhost/checkout<CRLF>
<CRLF>
<script>alert('hello')</script>
```

Fundamental Causes of XSS

- 使用者提交的 data 包含 **HTML tags** 和 **JavaScript** 代碼
- 如果這些 data 沒有經過過濾就傳到 User 的瀏覽器 → XSS attack / CRLF injection
- 瀏覽器無法分辨這些 Script 的來源，認為這就是網站的功能

Countermeasures

1. Filtering approach
2. Encoding approach
3. WAF (Web Application Firewall)

1. Filtering Approach

- Input Filtering Approach
 - 移除所有 user input 的 script/code/HTML tags
- Output Filtering Approach
 - 資料庫中保存原始的 user input，輸出到網頁時才做過濾
- 過濾方法可能存在漏洞，除了用 <script> tag 還有別種方法注入代碼

```

```
- 開發者要處理所有 user input 來源
 - e.g., query parameters, body parameters of POST request, HTTP headers
- 可用 open-source libraries 過濾 JavaScript code
 - e.g., jsoup

2. Encoding Approach

- 將 HTML Markups 換成 HTML Entities

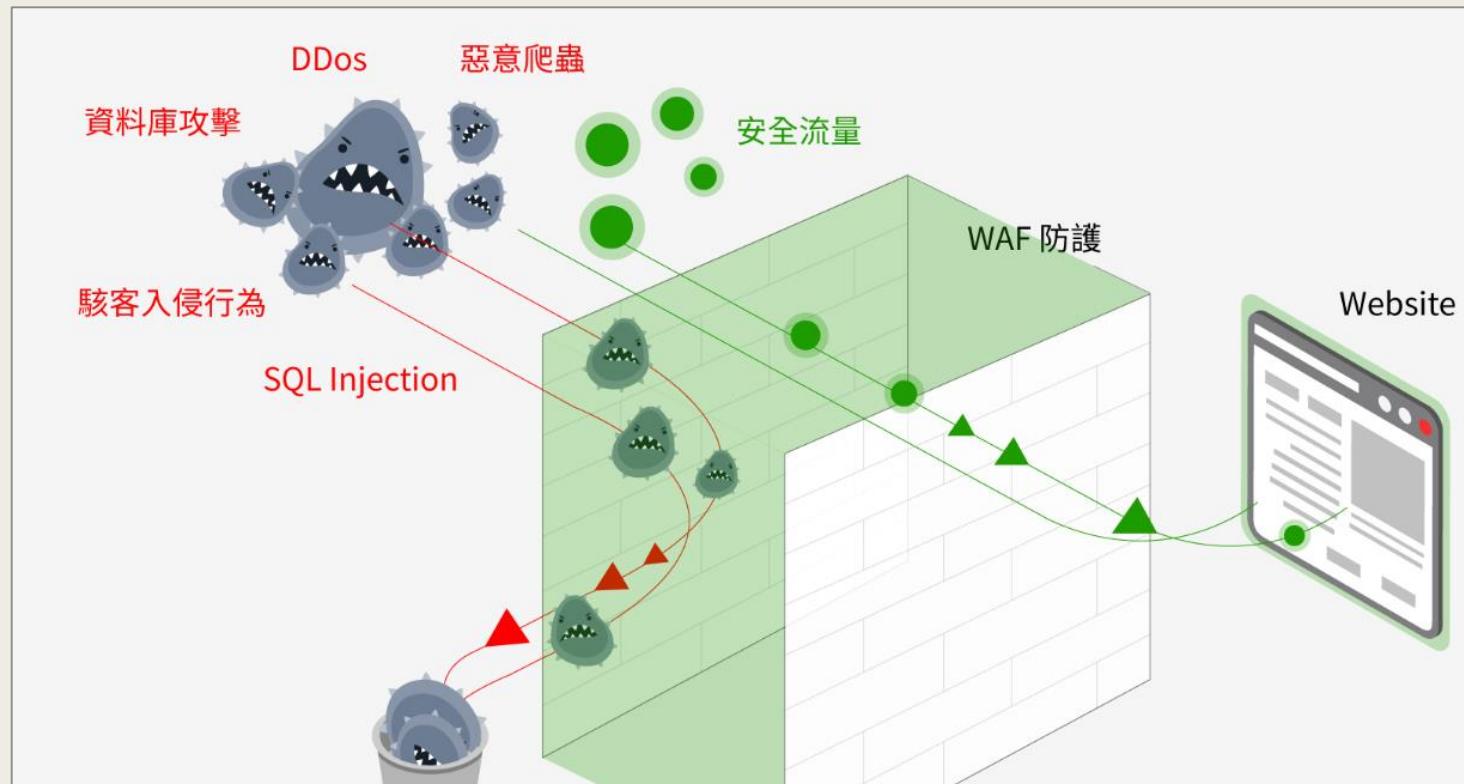
```
<span>a span text</span>
```

```
<span>&lt;span&gt;a span text&lt;/span&gt;</span>
```

- <script> alert('XSS') </script> → <script>alert('XSS')
- 這些 JavaScript 被 encode 過後，就會被瀏覽器顯示而不是執行
- 開發者要處理所有 user input 來源
 - e.g., query parameters, body parameters of POST request, HTTP headers

3. WAF (Web Application Firewall)

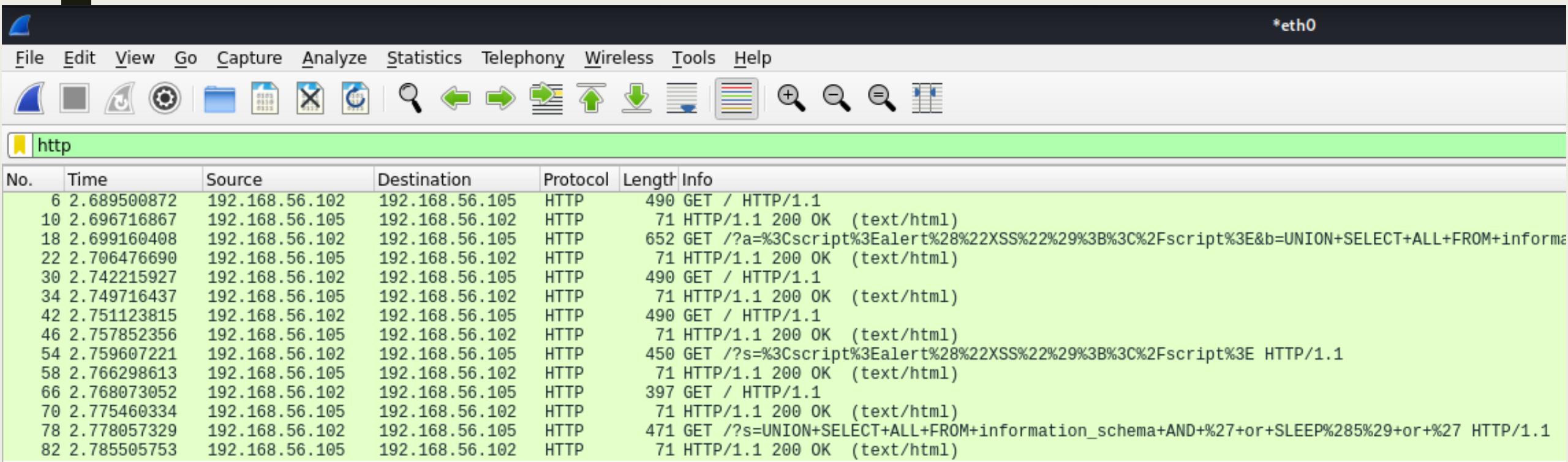
- 網站應用防火牆 (WAF) 會自動過濾惡意的 input，包含惡意的
 - Path, HTTP headers, HTML tag patterns 和 Javascript, SQL patterns



wafw00f

wafw00f

wafw00f



The screenshot shows the Wireshark interface with the following details:

- Network Interface:** *eth0
- File Menu:** File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, Help
- Toolbar:** Includes icons for opening files, saving files, capturing, stopping, zooming, and various analysis tools.
- Selected Filter:** http
- Table Headers:** No., Time, Source, Destination, Protocol, Length, Info
- Table Data:** A list of 15 captured HTTP requests. Key entries include:
 - Row 6: GET / HTTP/1.1
 - Row 10: 71 HTTP/1.1 200 OK (text/html)
 - Row 18: 652 GET /?a=%3Cscript%3Ealert%28%22XSS%22%29%3B%3C%2Fscript%3E&b=UNION+SELECT+ALL+FROM+information_schema+AND+%
 - Row 22: 71 HTTP/1.1 200 OK (text/html)
 - Row 30: 490 GET / HTTP/1.1
 - Row 34: 71 HTTP/1.1 200 OK (text/html)
 - Row 42: 490 GET / HTTP/1.1
 - Row 46: 71 HTTP/1.1 200 OK (text/html)
 - Row 54: 450 GET /?s=%3Cscript%3Ealert%28%22XSS%22%29%3B%3C%2Fscript%3E HTTP/1.1
 - Row 58: 71 HTTP/1.1 200 OK (text/html)
 - Row 66: 397 GET / HTTP/1.1
 - Row 70: 71 HTTP/1.1 200 OK (text/html)
 - Row 78: 471 GET /?s=UNION+SELECT+ALL+FROM+information_schema+AND+%27+or+SLEEP%285%29+or+%27 HTTP/1.1
 - Row 82: 71 HTTP/1.1 200 OK (text/html)

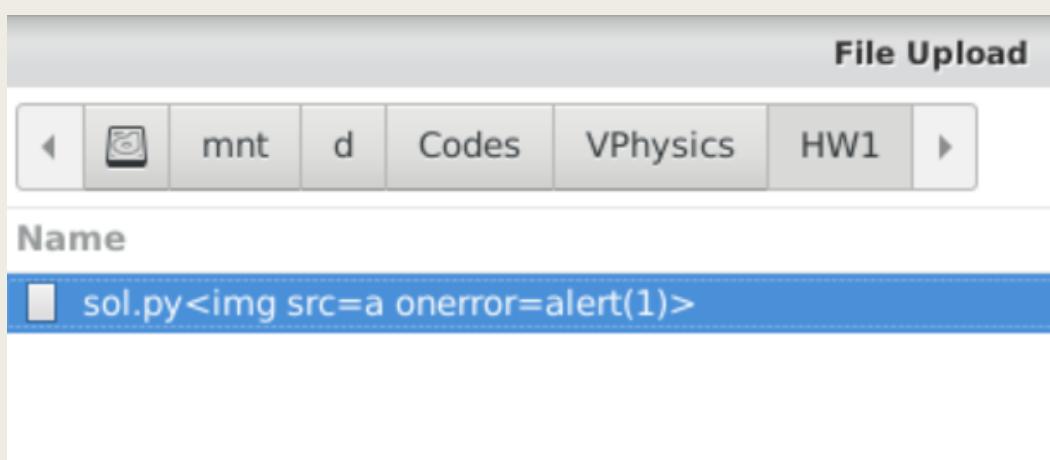
wafw00f

```
(kali㉿kali)-[~]
$ wafw00f -l
[+] Can test for these WAFs:
```

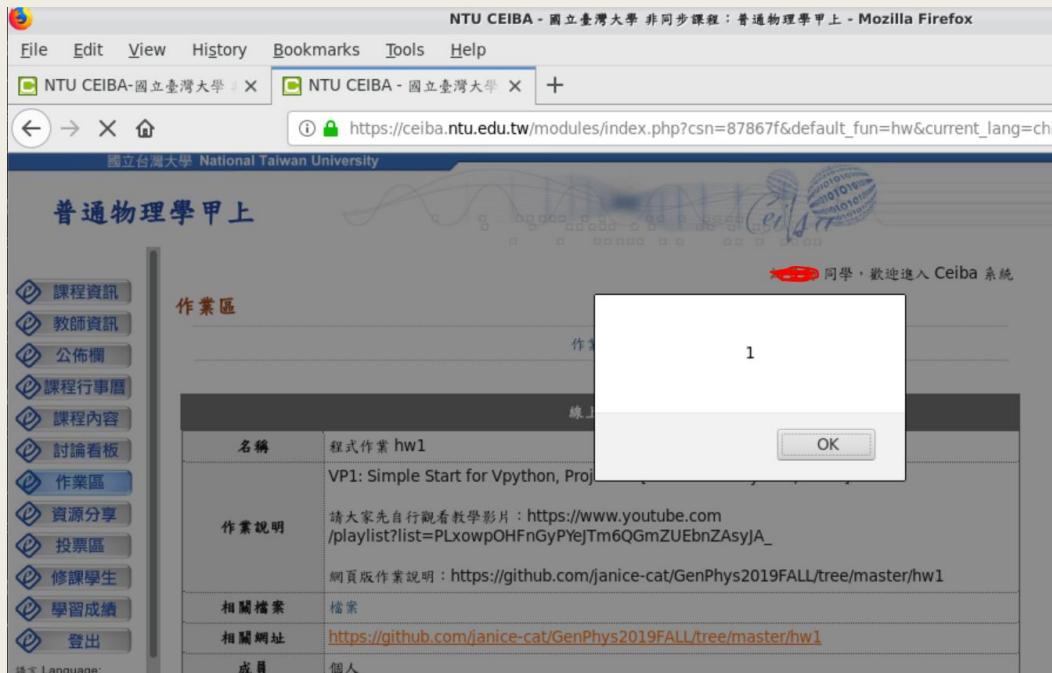
WAF Name	Manufacturer
ACE XML Gateway	Cisco
aeSecure	aeSecure
AireeCDN	Airee
Airlock	Phion/Ergon
Alert Logic	Alert Logic
AliYunDun	Alibaba Cloud Computing
Anquanbao	Anquanbao
AnYu	AnYu Technologies
Approach	Approach
AppWall	Radware
Armor Defense	Armor
ArvanCloud	ArvanCloud
ASP.NET Generic	Microsoft
ASPA Firewall	ASPA Engineering Co.
Astra	Czar Securities
AWS Elastic Load Balancer	Amazon
AzionCDN	AzionCDN
Azure Front Door	Microsoft
Barikode	Ethic Ninja
Barracuda	Barracuda Networks
Bekchy	Faydata Technologies Inc.
Beluga CDN	Beluga
BIG-IP Local Traffic Manager	F5 Networks
BinarySec	BinarySec
BitNinja	BitNinja
BlockDoS	BlockDoS
Bluedon	Bluedon IST
BulletProof Security Pro	AITpro Security
CacheWall	Varnish
CacheFly CDN	CacheFly

CEIBA 網站 XSS 漏洞

- 可透過上傳檔案之副檔名達成 XSS 攻擊
- 名稱的部分會被重新命名，但是副檔名並不會被過濾



The screenshot shows the 'File Upload' interface on the NTU CEIBA platform. A file named 'sol.py' is selected for upload. The interface includes navigation buttons for 'mnt', 'd', 'Codes', 'VPhysics', and 'HW1'.



The screenshot shows a confirmation dialog box from Mozilla Firefox. It displays the URL https://ceiba.ntu.edu.tw/modules/index.php?csn=87867f&default_fun=hw¤t_lang=ch. The dialog box contains the message: "同學，歡迎進入 Ceiba 系統" (Welcome to the Ceiba system) and "1" (indicating one item). The "OK" button is visible at the bottom right.

Summary of XSS

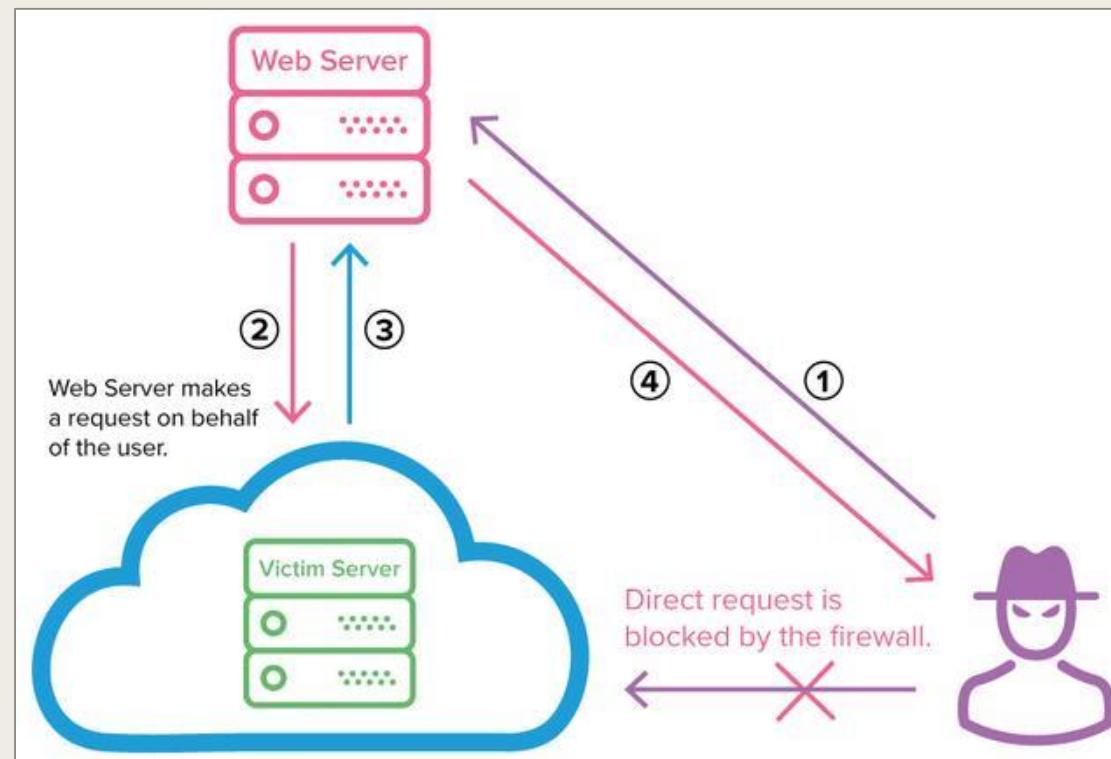
- Two types of XSS attacks
- How to launch XSS attacks
- Countermeasures against XSS attacks

The best defense of XSS?

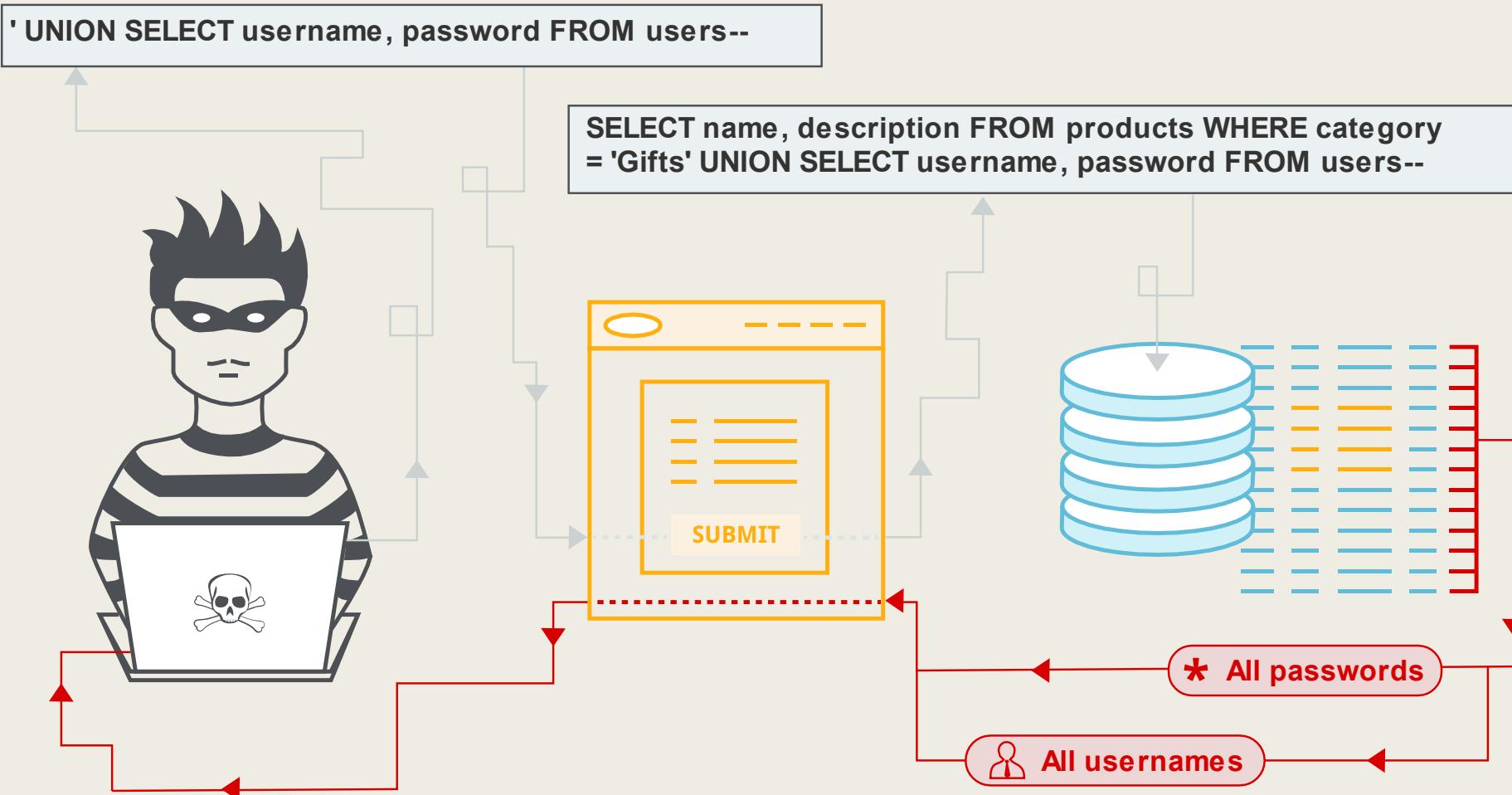
- Define trust boundaries
- Validate and sanitize untrusted data

SSRF (Server-Side Request Forgery)

- 服務器端請求偽造，以伺服器的身份發送一條構造好的請求
- 很多 web 應用都提供了從其他的伺服器上獲取數據的功能
- SSRF 可讓攻擊者的請求重定向到防火牆後的內部網路或本地主機



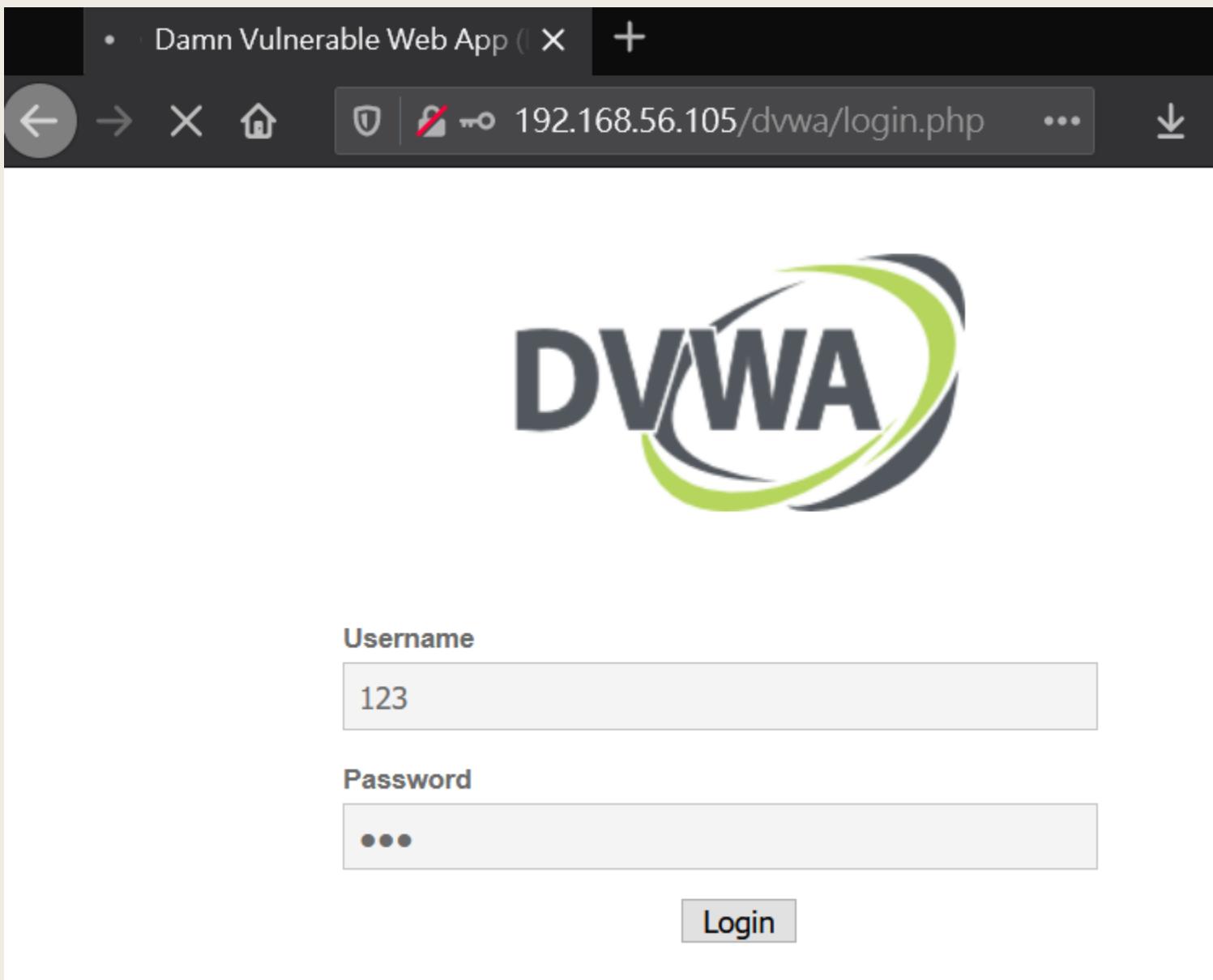
SSRF (Server-Side Request Forgery)



TOOL

Burp
DirBuster

Lab1: Brute Force



Send the login POST request to Intruder

The screenshot shows the Burp Suite interface for a 'Temporary Project'. The 'Proxy' tab is selected, and the 'Intercept' button is active. A POST request to `http://192.168.56.105/dvwa/login.php` is listed in the message list. The 'Actions' context menu is open over the request, with the 'Send to Intruder' option highlighted in orange. Other options in the menu include: Send to Repeater (Ctrl-R), Send to Sequencer, Send to Comparer, Send to Decoder, Request in browser, Engagement tools [Pro version only], Change request method, Change body encoding, Copy URL, Copy as curl command, Copy to file, Paste from file, Save item, Don't intercept requests, Do intercept, Convert selection, and URL-encode as you type.

Damn Vulnerable Web App (x) +

192.168.56.105/dvwa/login.php

Burp Suite Community Edition v2021.3.1 - Temporary Project

Burp Project Intruder Repeater Window Help

Dashboard Target **Proxy** Intruder Repeater Sequencer Decoder Comparer Extender Pro

Intercept HTTP history WebSockets history Options

Request to http://192.168.56.105:80

Forward Drop Intercept is on Actions

Pretty Raw \n Actions ▾

```
1 POST /dvwa/login.php HTTP/1.1
2 Host: 192.168.56.105
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:86.0) Gecko/20100101 Firefox/86.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: zh-TW,zh;q=0.8,en-US;q=0.5,en;q=0.3
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 37
9 Origin: http://192.168.56.105
10 Connection: close
11 Referer: http://192.168.56.105/dvwa/login.php
12 Cookie: security=high; PHPSESSID=8a012a8537a0e970a
13 Upgrade-Insecure-Requests: 1
14
15 username=123&password=qwe&Login=Login
```

Send to Intruder Ctrl-I

Send to Repeater Ctrl-R

Send to Sequencer

Send to Comparer

Send to Decoder

Request in browser >

Engagement tools [Pro version only] >

Change request method

Change body encoding

Copy URL

Copy as curl command

Copy to file

Paste from file

Save item

Don't intercept requests >

Do intercept >

Convert selection >

URL-encode as you type

1 x 2 x ...

Payload Positions**Start attack**

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.

Attack type: Cluster bomb <https://ithelp.ithome.com.tw/articles/10246457>

```
1 POST /dvwa/login.php HTTP/1.1
2 Host: 192.168.56.105
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:86.0) Gecko/20100101 Firefox/86.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: zh-TW,zh;q=0.8,en-US;q=0.5,en;q=0.3
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 37
9 Origin: http://192.168.56.105
10 Connection: close
11 Referer: http://192.168.56.105/dvwa/login.php
12 Cookie: security=high; PHPSESSID=8a012a8537a0e970a145d7746dd6e945
13 Upgrade-Insecure-Requests: 1
14
15 username=$123$&password=$qwe$&Login=Login
```

Add §**Clear §****Auto §****Refresh**

Target Positions **Payloads** Options

② Payload Sets

You can define one or more payload sets. The number of payload sets depends on the target, and each payload set has a defined payload count and request count.

Payload set: **1** Payload count: 5

Payload type: **Simple list** Request count: 0

③ Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

Paste Load ... Remove Clear

admin
root
password
administrator

Add Enter a new item f08921a01

Payload Sets

You can define one or more payload sets. The number of payload sets can be defined in the Target tab. Each payload set contains one or more payload types. The number of payload types available for each payload set, and each payload type, depends on the target configuration.

Payload set: **2** Payload count: 1,244
 Payload type: Simple list Request count: 6,220

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used for various services.

Paste	admin
Load ...	password
Remove	1234
Clear	epicrouter
Add	sysadm
	access
	root
	tech
Add	Enter a new item

Look In: metasploit

- adobe_top100_pass.txt
- av_hips_executables.txt
- av-update-urls.txt
- burnett_top_500.txt
- burnett_top_1024.txt
- can_flood_frames.txt
- cms400net_default_userpass.txt
- common_roots.txt
- dangerzone_a.txt
- dangerzone_b.txt
- db2_default_pass.txt
- db2_default_user.txt
- db2_default_userpass.txt
- default_pass_for_services_unhash.txt**
- default_userpass_for_services_unhash.txt
- default_users_for_services_unhash.txt
- dlink_telnet_backdoor_userpass.txt
- hci_oracle_passwords.csv
- http_default_pass.txt
- http_default_userpass.txt
- http_default_users.txt
- http_owa_common.txt
- idrac_default_pass.txt
- idrac_default_user.txt
- ipmi_passwords.txt
- ipmi_users.txt
- joomla.txt
- keyboard-patterns.txt
- lync_subdomains.txt
- malicious_urls.txt
- mirai_pass.txt
- mirai_user.txt
- mirai_user_pass.txt
- multi_vendor_cctv_dvr_pass.txt
- multi_vendor_cctv_dvr_users.txt
- named_pipes.txt
- namelist.txt
- oracle_default_hashes.txt
- oracle_default_passwords.csv
- oracle_default_userpass.txt
- password.lst
- piata_ssh_userpass.txt

File Name: **default_pass_for_services_unhash.txt**

```
(kali㉿kali)-[~]
$ sudo cp -Lr /usr/share/wordlists /media/sf_vm_share
```

? Payload Sets**Start attack**

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: ▼ Payload count: 1,244

Payload type: ▼ Request count: 6,220

? Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

<input type="button" value="Paste"/>	admin
<input type="button" value="Load ..."/>	password
<input type="button" value="Remove"/>	1234
<input type="button" value="Clear"/>	epicrouter
<input type="button" value="Add"/>	sysadm
	access
	root
	tech

Enter a new item

Fail/Success Responses

Intruder attack 20

Attack Save Columns

Results Target Positions Payloads Options

Filter: Showing all items

Request ^	Payload1	Payload2	Status	Error	Timeout	Length	Comment
0			302	<input type="checkbox"/>	<input type="checkbox"/>	354	
1	admin	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	354	
2	root	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	354	
3	password	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	354	
4	administrator	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	354	
5	admin		302	<input type="checkbox"/>	<input type="checkbox"/>	354	

Request Response

Pretty Raw Render \n Actions ▾

```
1 HTTP/1.1 302 Found
2 Date: Sun, 21 Mar 2021 17:36:56 GMT
3 Server: Apache/2.2.8 (Ubuntu) DAV/2
4 X-Powered-By: PHP/5.2.4-2ubuntu5.10
5 Expires: Thu, 19 Nov 1981 08:52:00 GMT
6 Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
7 Pragma: no-cache
8 Location: login.php
9 Content-Length: 0
10 Connection: close
11 Content-Type: text/html
12
```



Screenshot-01: Payload1 放上學號

Intruder attack 19

Attack Save Columns

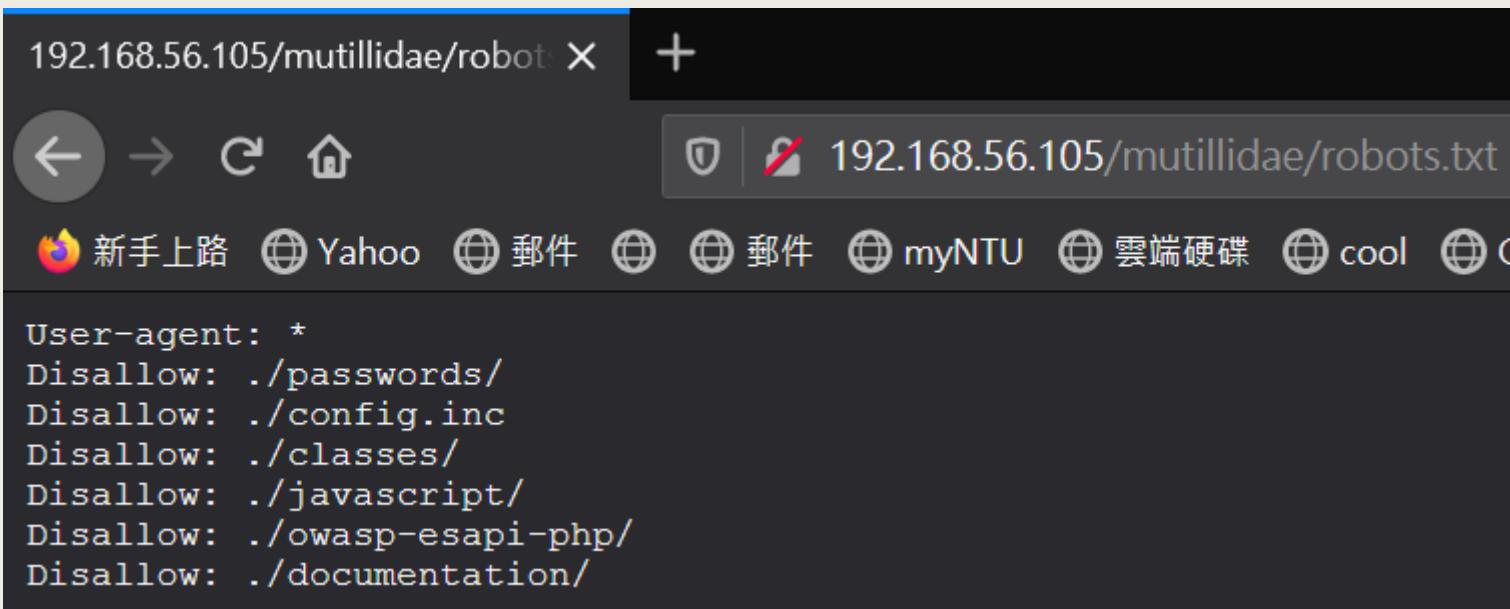
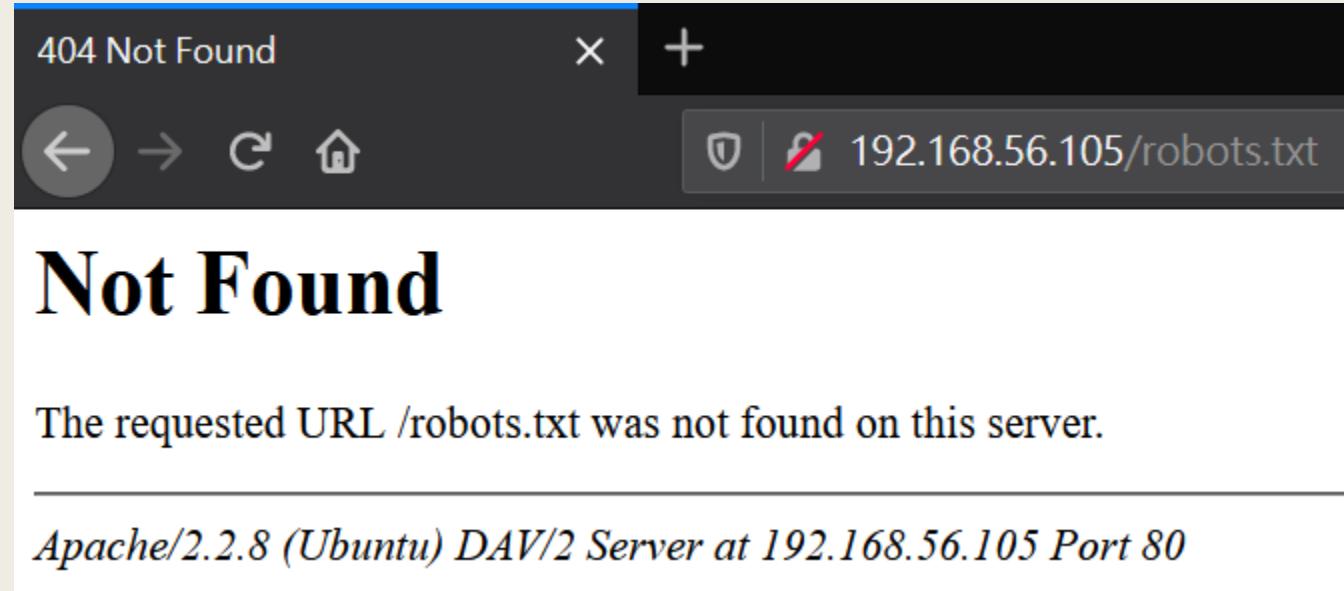
Results Target Positions Payloads Options

Filter: Showing all items ?

Request ^	Payload1	Payload2	Status	Error	Timeout	Length	\nLocation:
0	f08921a01		302			354	login.php
1	admin	admin	302			354	login.php
2	root	admin	302			354	login.php
3	password	admin	302			354	login.php
4	administrator	admin	302			354	login.php
5	admin		302			354	login.php
6	root		302			354	login.php
7	password		302			354	login.php
8	administrator		302			354	login.php
9	admin	password	302			354	index.php
10	root	password	302			354	login.php
11	password	password	302			354	login.php
12	administrator	password	302			354	login.php
13	admin	1234	302			354	login.php
14	root	1234	200			254	login.php

Request Response

robots.txt

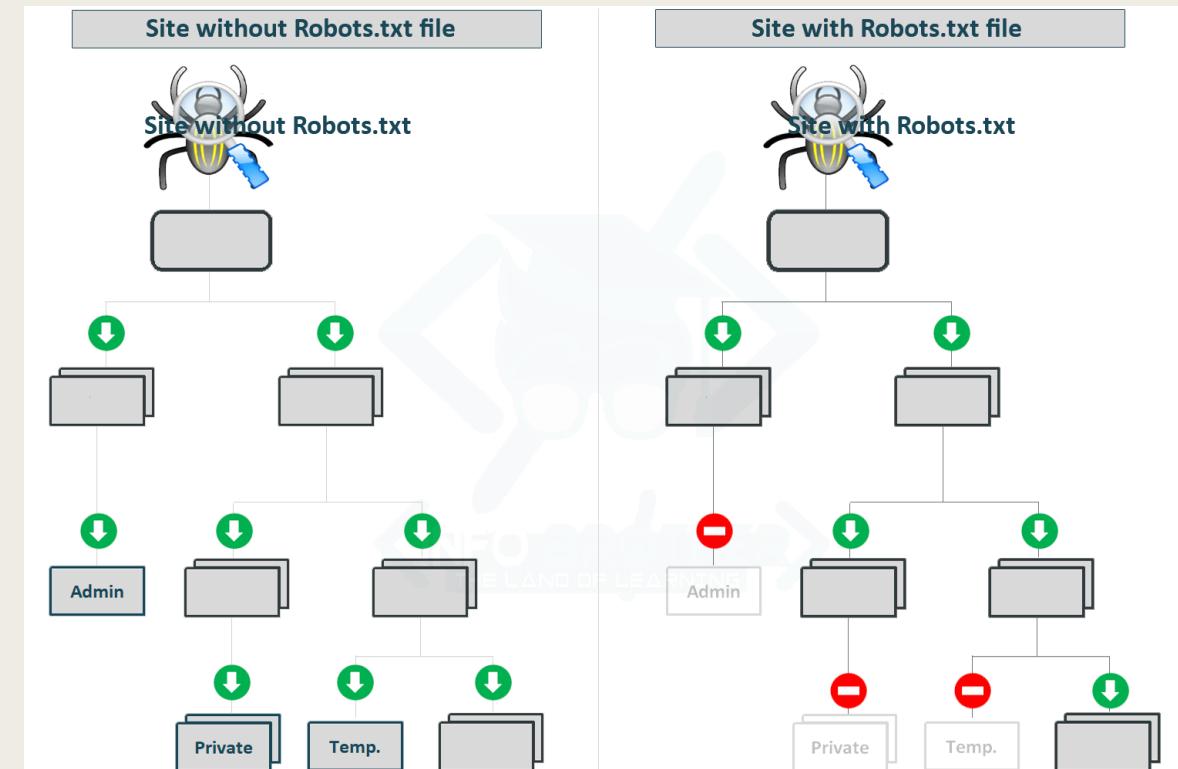


robots.txt

- robots.txt 主要用於管理搜尋引擎檢索器（又稱網路蜘蛛）對網站造成的流量負擔
- 告訴搜尋引擎檢索器，可以或不可以對網站上的哪些網頁或檔案提出要求
- robots.txt 協定
 - 並不是一個規範，而是約定俗成
 - 並不能保證網站的隱私

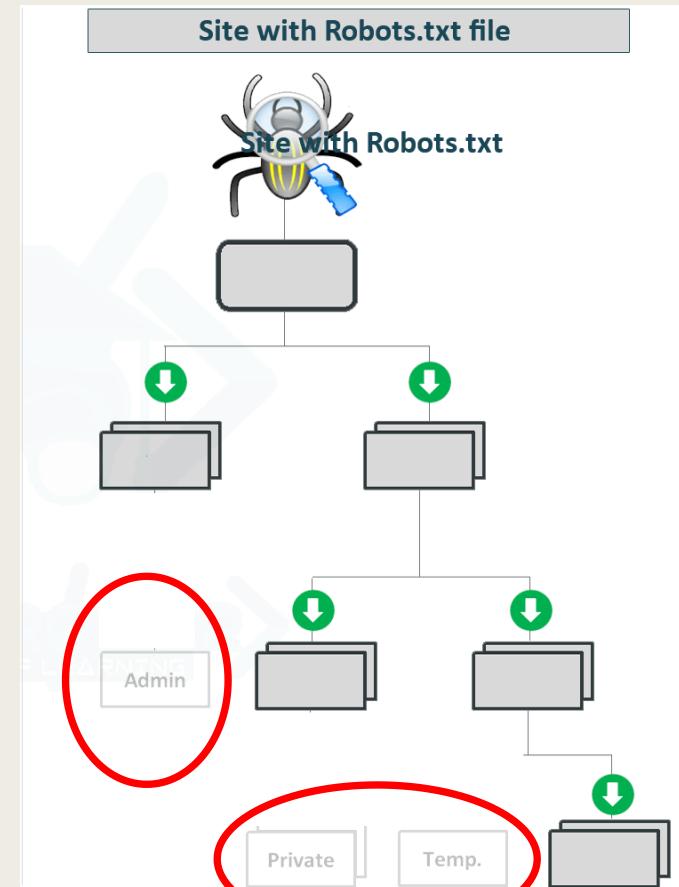
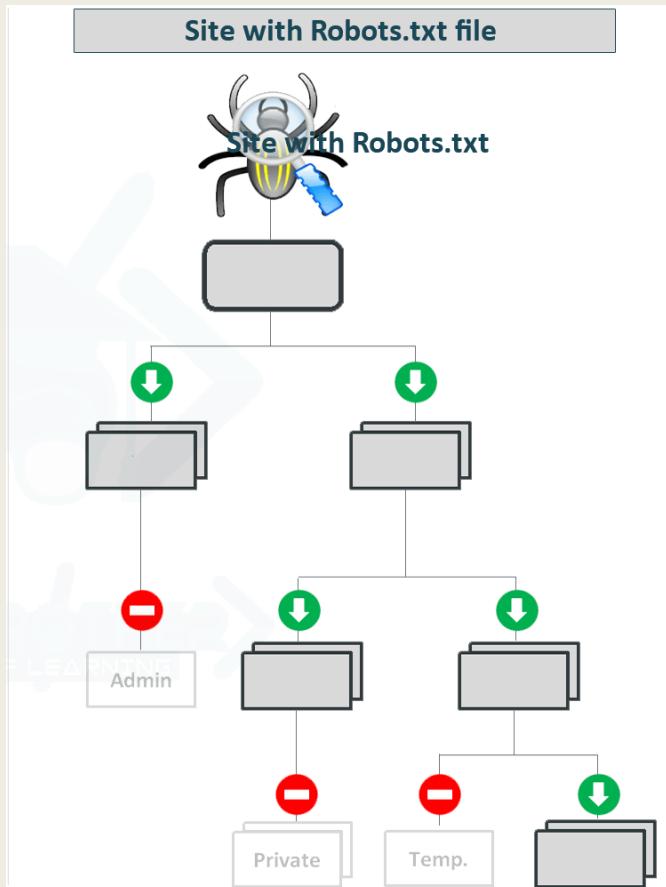
Google:

<https://developers.google.com/search/docs/advanced/robots/intro>



Forceful Browsing

- Forceful browsing (forced browsing) is a brute force attack that aims to enumerate files and gain access to resources that the application **does not reference, but can still retrieve**.



DirBuster



```
(kali㉿kali)-[~]
$ dirbuster
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Starting OWASP DirBuster 1.0-RC1
[]
```

OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

File Options About Help

Target URL (eg http://example.com:80/)
http://192.168.56.106/mutillidae/

Work Method Use GET requests only Auto Switch (HEAD and GET)

Number Of Threads 10 Threads Go Faster

Select scanning type: List based brute force Pure Brute Force

File with list of dirs/files
/usr/share/wordlists/dirbuster/directory-list-2.3-small.txt

Char set Min length Max Length

Select starting options: Standard start point URL Fuzz
 Brute Force Dirs Be Recursive Dir to start with /mutillidae/
 Brute Force Files Use Blank Extension File extension

URL to fuzz - /test.html?url={dir}.asp
/mutillidae/

Please complete the test details

/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt

/usr/share/wordlists/dirbuster/directory-list-2.3-small.txt

OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

File Options About Help

http://192.168.56.106:80/mutillidae/

Scan Information | Results - List View: Dirs: 0 Files: 955 | Results - Tree View | Errors: 8

Type	Found	Response	Size
File	/mutillidae/styles/ddsmoothmenu/ddsmoothmenu.css	200	2602
Dir	/mutillidae/styles/ddsmoothmenu/	200	1517
Dir	/mutillidae/styles/	200	1293
File	/mutillidae/styles	200	1293
File	/mutillidae/set-up-database.php	200	183
File	/mutillidae/robots	200	482
Dir	/mutillidae/register/	200	2000
File	/mutillidae/register	200	2062
File	/mutillidae/passwords/accounts.txt	200	418
Dir	/mutillidae/passwords/	200	1095
File	/mutillidae/passwords	200	1095
Dir	/mutillidae/notes/	200	1898
File	/mutillidae/notes	200	1957
Dir	/mutillidae/login/		
File	/mutillidae/login		
File	/mutillidae/javascript/html5-secrets.js		
File	/mutillidae/javascript/follow-mouse.js		
File	/mutillidae/javascript/ddsmoothmenu/readme.txt		
File	/mutillidae/javascript/ddsmoothmenu/jquery.min.js	200	57558
File	/mutillidae/javascript/ddsmoothmenu/ddsmoothme...	200	9074
Dir	/mutillidae/javascript/ddsmoothmenu/	200	1523
File	/mutillidae/javascript/bookmark-site.js	200	1320
Dir	/mutillidae/javascript/	200	1700
File	/mutillidae/javascript	200	1700
Dir	/mutillidae/installation/	200	8317
File	/mutillidae/installation	200	8383
File	/mutillidae/index.php	200	326
Dir			

Current speed: 405 requests/sec
Average speed: (T) 336, (C) 386 requests/sec
Parse Queue Size: 1349
Total Requests: 85842/178261
Time To Finish: 00:03:59

(Select and right click for more options)

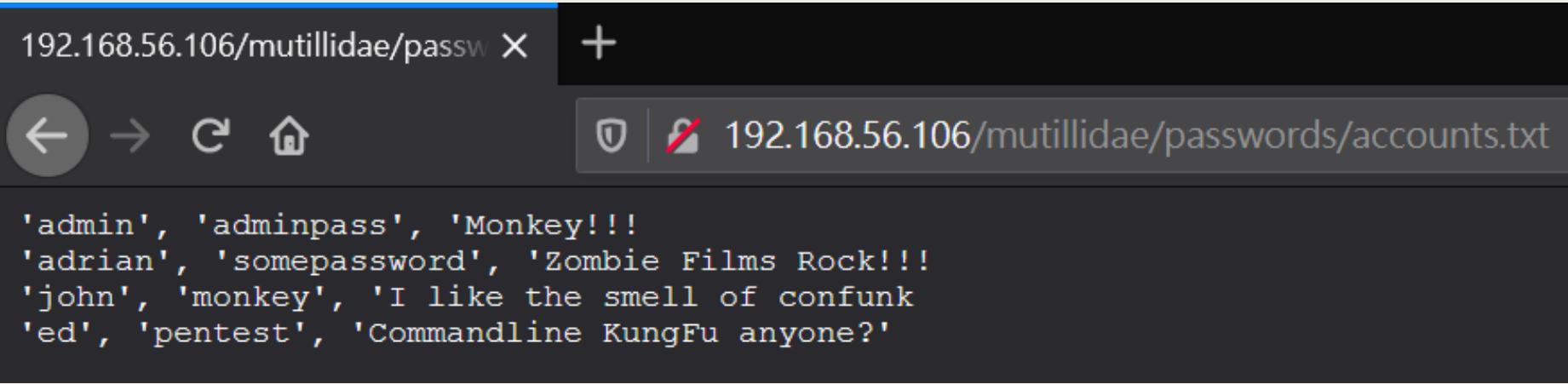
Current number of running threads: 10

Back Pause Stop Report

Starting dir/file list based brute forcing /mutillidae/242794/

Screenshot-02: Change 欄位放上學號

Hidden resource



A screenshot of a web browser window. The address bar shows the URL `192.168.56.106/mutillidae/passw`. Below the address bar are standard navigation buttons: back, forward, refresh, and home. To the right of the address bar is a status bar showing the URL `192.168.56.106/mutillidae/passwords/accounts.txt`. The main content area of the browser displays a text file with the following content:

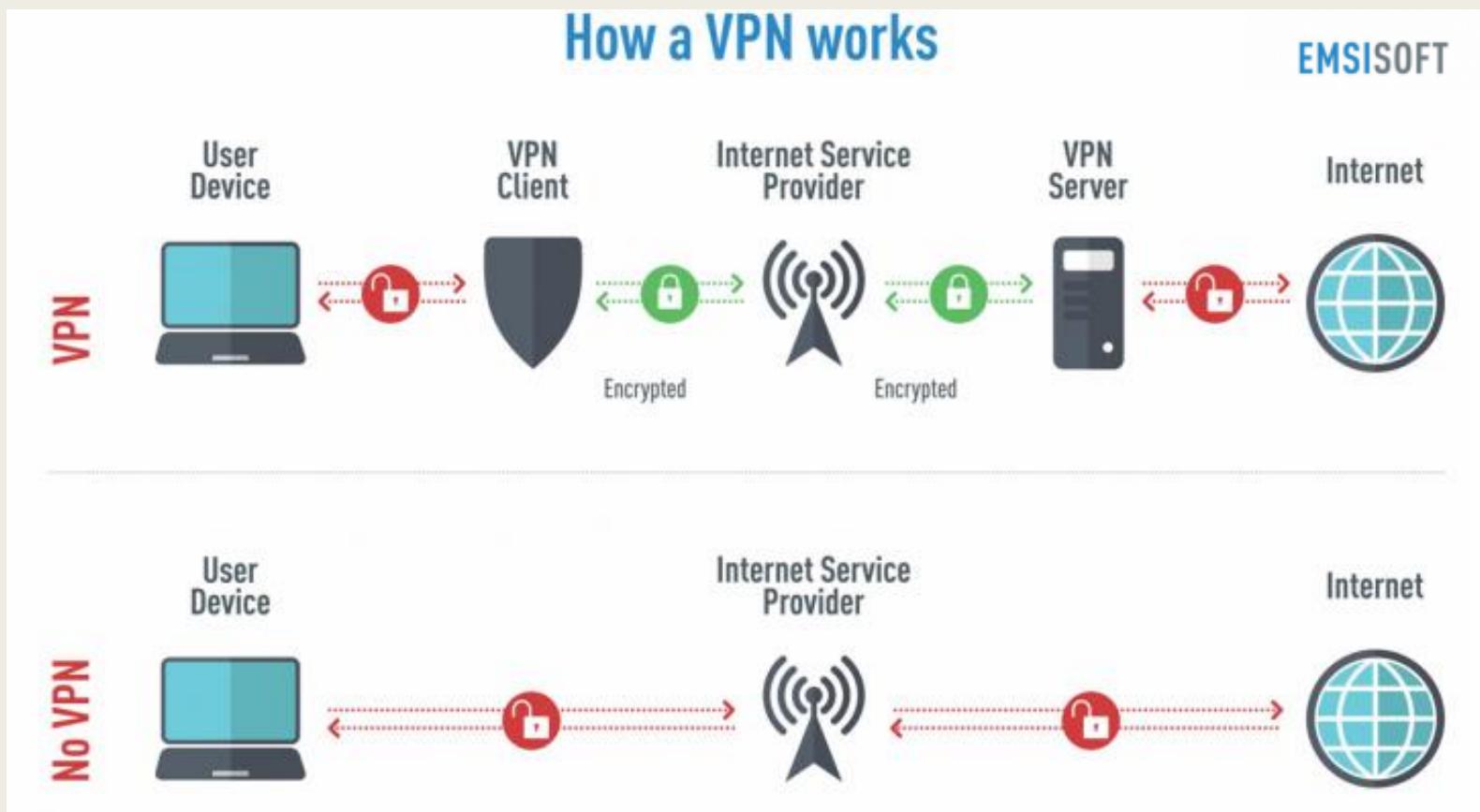
```
'admin', 'adminpass', 'Monkey!!!  
'adrian', 'somepassword', 'Zombie Films Rock!!!  
'john', 'monkey', 'I like the smell of confunk  
'ed', 'pentest', 'Commandline KungFu anyone?'
```

PRIVACY & ANONYMITY

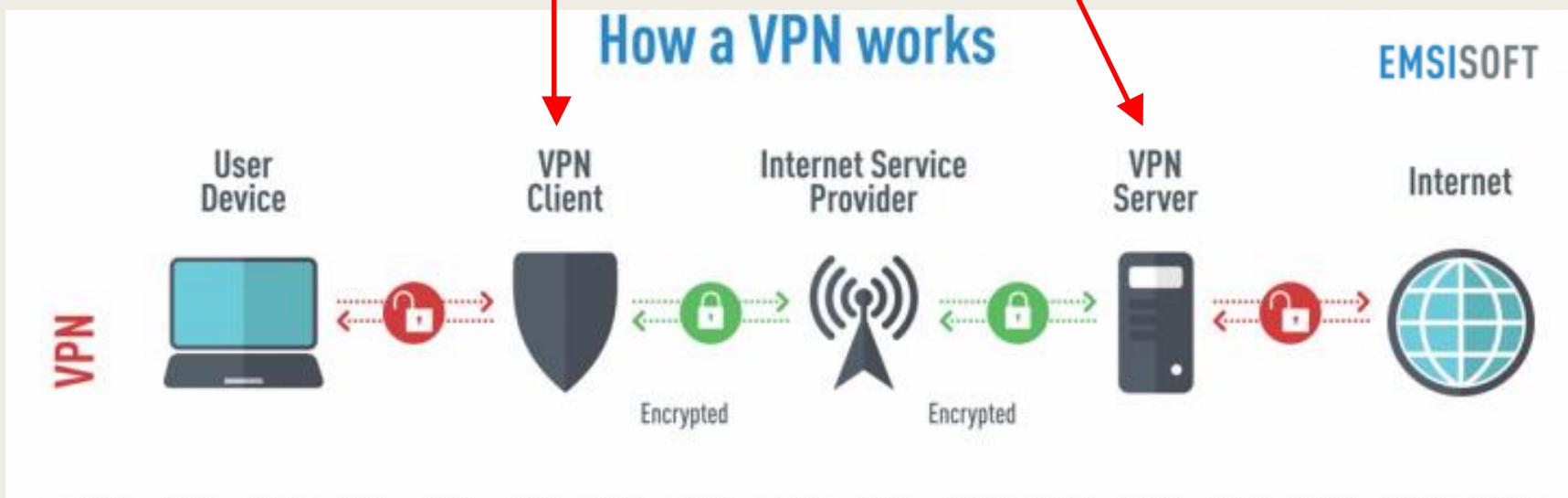
VPN
DNS Over HTTPS
TOR

VPN (Virtual Private Network)

- A type of software that allows you to connect to the internet via an encrypted tunnel.
- It hides your IP address and location using proxy servers.

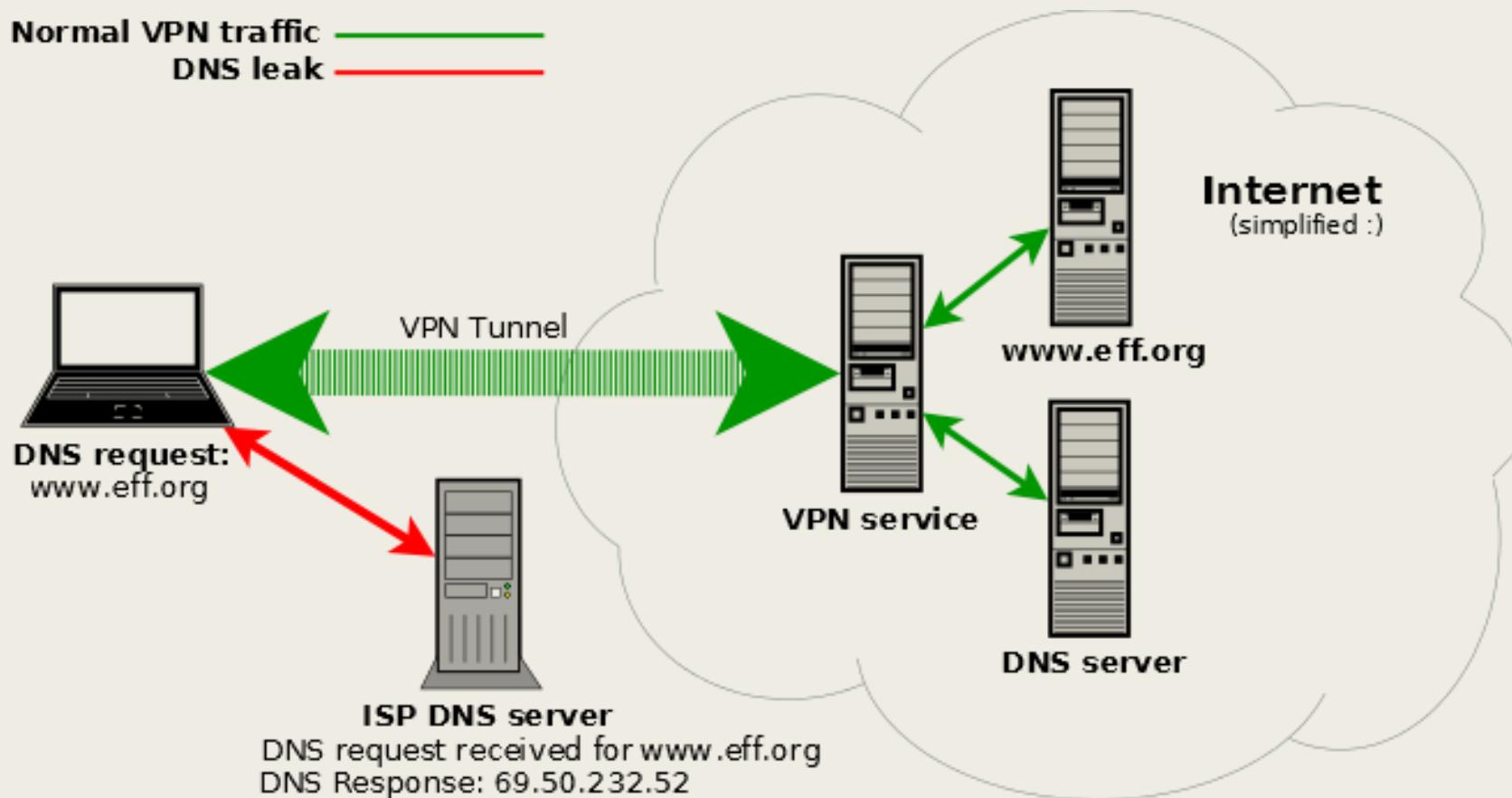


VPN



DNS leak in VPN

<https://www.dnsleaktest.com/>



DNS leak Demo

```
wifi0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
      inet 172.20.10.8 netmask 255.255.255.240 broadcast 172.20.10.15
```



```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1350  
      inet 140.112.150.27 netmask 255.255.255.255 broadcast 140.112.150.27
```

DNS leak Demo

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1350 inet 140.112.150.27 netmask 255.255.255.255 broadcast 140.112.150.27								
No.	Time	Source	Destination	Protocol	Length	Source Port	Info	
1	0.000000	2404:6800:4008:803::1	2001:288:1001:71:1::1	TCP	86	443	443	
2	1.122372	140.112.150.27	13.35.24.84	TCP	55	49232	492	
3	1.122434	140.112.150.27	13.35.24.84	TCP	55	49232	[TCP]	
4	1.288341	13.35.24.84	140.112.150.27	TCP	66	443	443	

wifi0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500 inet 172.20.10.8 netmask 255.255.255.240 broadcast 172.20.10.15								
No.	Time	Source	Destination	Protocol	Length	Source Port	Info	
1	0.000000	140.112.2.228	172.20.10.8	TLSv1.2	178	443	App	
2	0.041289	172.20.10.8	140.112.2.228	TCP	54	64205	642	
3	0.045642	140.112.2.228	172.20.10.8	TCP	54	443	443	
4	0.409225	140.112.2.228	172.20.10.8	TLSv1.2	185	443	App	

```
# nslookup sslvpn2.ntu.edu.tw
Server: 140.112.254.4
Address: 140.112.254.4#53

Name: sslvpn2.ntu.edu.tw
Address: 140.112.2.228
```

DNS leak Demo

```
wifi0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
inet 172.20.10.8 netmask 255.255.255.240 broadcast 172.20.10.15
```

No.	Time	Source	Destination	Protocol	Length	Source Port	Info
1227	4.466633	172.20.10.8	140.112.2.228	TCP	54	64205	64205 → 443 [ACK] Seq=74250 Ack=425820 Win=64755 Len=0
1228	4.466642	140.112.2.228	172.20.10.8	TLSv1.2	496	443	Application Data
1229	4.467371	172.20.10.8	140.112.2.228	TLSv1.2	158	64205	Application Data
1230	4.467466	172.20.10.8	172.20.10.1	DNS	73		Standard query 0x480c AAAA d.pub.network
1231	4.487771	172.20.10.8	140.112.2.228	TLSv1.2	169	64205	Application Data
1232	4.488003	172.20.10.8	172.20.10.1	DNS	84		Standard query 0xc31d A www.google-analytics.com
1233	4.488330	172.20.10.8	140.112.2.228	TLSv1.2	166	64205	Application Data
1234	4.488445	172.20.10.8	172.20.10.1	DNS	81		Standard query 0x5ffa A secure.quantserve.com
1235	4.490027	172.20.10.8	140.112.2.228	TLSv1.2	240	64205	Application Data
1236	4.491052	172.20.10.8	140.112.2.228	TLSv1.2	174	64205	Application Data
1237	4.495871	172.20.10.1	172.20.10.8	DNS	144		Standard query response 0xc31d A www.google-analytics.com CNAME
1238	4.505020	140.112.2.228	172.20.10.8	TLSv1.2	237	443	Application Data
1239	4.506226	172.20.10.1	172.20.10.8	DNS	138		Standard query response 0x480c AAAA d.pub.network SOA anirban.n
1240	4.533163	140.112.2.228	172.20.10.8	TLSv1.2	232	443	Application Data

```
預設閘道 . . . . . : fe80::4d2:5c5d:5820:9bac%17  
172.20.10.1  
DHCP 啟服器 . . . . . : 172.20.10.1  
DHCPv6 IAID . . . . . : 164160101  
DHCPv6 用戶端 DUID: . . . . . : 00-01-00-01-27-CB-E6-7F-C8-E2-65-FF-32-5F  
DNS 啟服器 . . . . . : fe80::4d2:5c5d:5820:9bac%17  
172.20.10.1
```

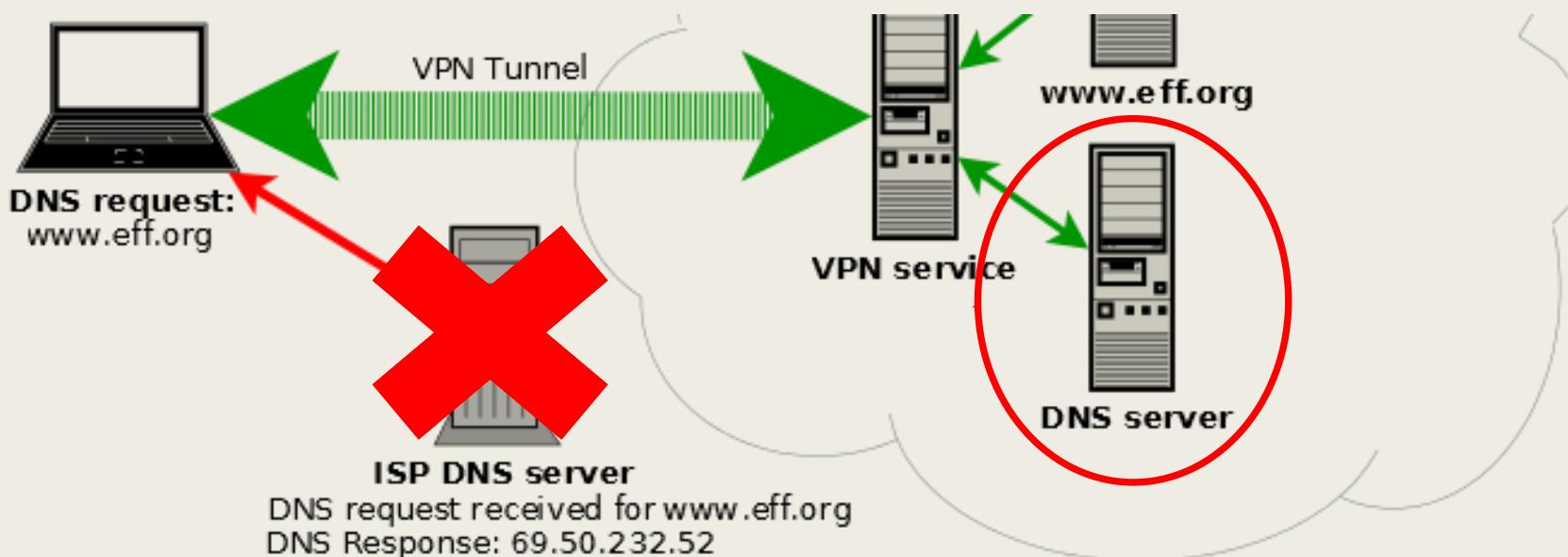
DNS leak Demo

- Leaked DNS queries are done by TWN Broadband

IP	Hostname	ISP	Country
140.112.254.65	ccdns1.cc.ntu.edu.tw.	Taiwan Academic Network	New Taipei, Taiwan 
140.112.254.66	ccdns2.cc.ntu.edu.tw.	Taiwan Academic Network	New Taipei, Taiwan 
140.112.254.69	ccdns3.cc.ntu.edu.tw.	Taiwan Academic Network	New Taipei, Taiwan 
140.112.254.70	ccdns6.cc.ntu.edu.tw.	Taiwan Academic Network	New Taipei, Taiwan 
175.96.61.21	dnsr23.tfn.net.tw.	TWN Broadband	Taiwan 
175.96.61.22	dnsr24.tfn.net.tw.	TWN Broadband	Taiwan 
60.199.22.62	dnsr03.tfn.net.tw.	TWN Broadband	Taiwan 

How to fix a DNS leak

- Depends on VPN service
 - Ensure that once connected to the VPN, you are using ONLY the DNS servers provided by the VPN service.



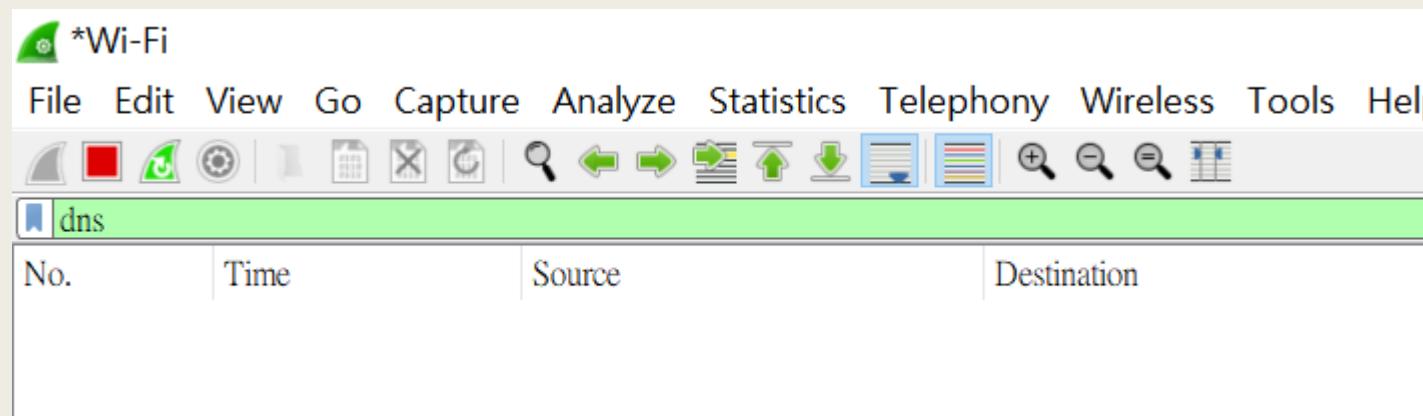
How to fix a DNS leak



- Assign DNS server

```
預設閘道 . . . . . : 172.20.10.1
DHCP 伺服器 . . . . . : 172.20.10.1
DNS 伺服器 . . . . . : 1.1.1.1
```

- No more DNS leak



DNS Over HTTPS (DoH)

■ 傳統 DNS

- 使用 53 port 向 DNS server 發送明文請求訊息

```
(root㉿kali)-[~/home/kali]
└─# strace ping google.com -c1 2>&1 | grep '(53)'
connect(5, {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet_addr("140.112.254.4")}, 0)
recvfrom(5, "\212\227\201\200\0\1\0\1\0\0\0\0\6google\3com\0\0\1\0\1\300\f\0\1"..., 2048
recvfrom(5, "\230\221\201\200\0\1\0\1\0\0\0\0\6google\3com\0\0\34\0\1\300\f\0\34"..., 65
connect(5, {sa_family=AF_INET, sin_port=htons(53), sin_addr=inet_addr("140.112.254.4")}, 0)
recvfrom(5, "\316\365\201\200\0\1\0\1\0\0\0\0\00214\00224\003217\003172\7in-ad"..., 1024

(root㉿kali)-[~/home/kali]
└─# cat /etc/resolv.conf
# Generated by NetworkManager
search ntu.edu.tw
nameserver 140.112.254.4
```

■ DNS Over HTTPS

- DNS requests will be encapsulated in encrypted HTTPS packets.

DoH deployment scenarios

- Using a DoH implementation within an application
 - Browsers
- Installing a DoH proxy
 - on the name server in the local network
 - on a local system
- Operating systems
 - In May 2020, Microsoft released Windows 10 Insider Preview Build 19628 that included initial support for DoH.
 - Apple's iOS 14 and macOS 11 released in late 2020 support both DoH and DoT protocols.

Enable DoH in Browser

在 Firefox 62 及以上版本中開啟 DNS over HTTPS

1. 在瀏覽器網址列輸入 `about:config` 然後打開，並同意警告資訊。
2. 搜尋 `network.trr`
3. 設定 `network.trr.mode` 值為 2
4. 在 `network.trr.uri` 中填入上表中任一伺服器，例如：<https://1.1.1.1/dns-query>

Publicly available DoH servers:

<https://github.com/curl/curl/wiki/DNS-over-HTTPS>

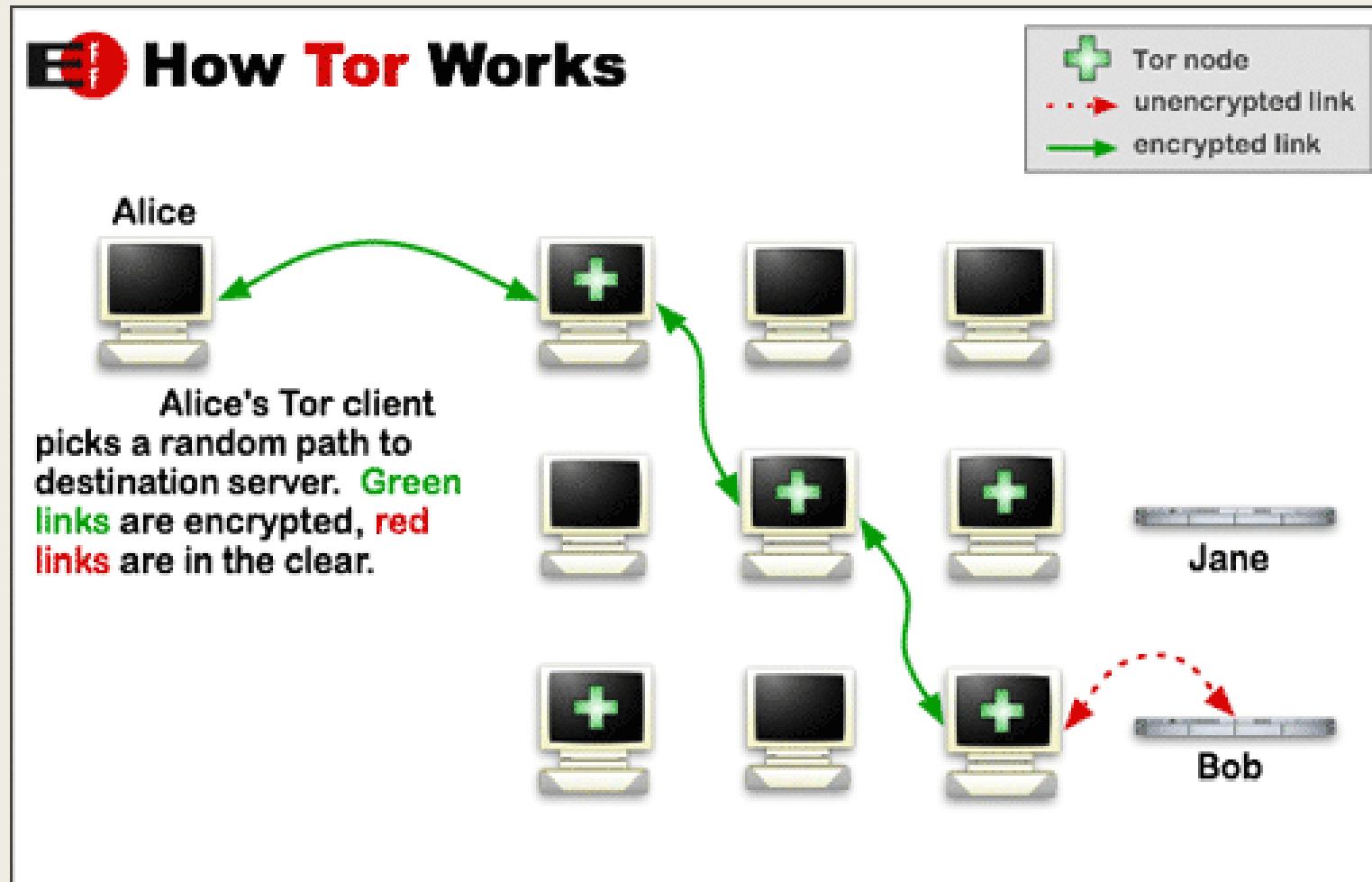
Test complete

Query round Progress... Servers found
1

Sponsored by
IVPN
Ultimate IP leak Protection

IP	Hostname	ISP	Country
162.158.241.8	None	Cloudflare	New Taipei, Taiwan 

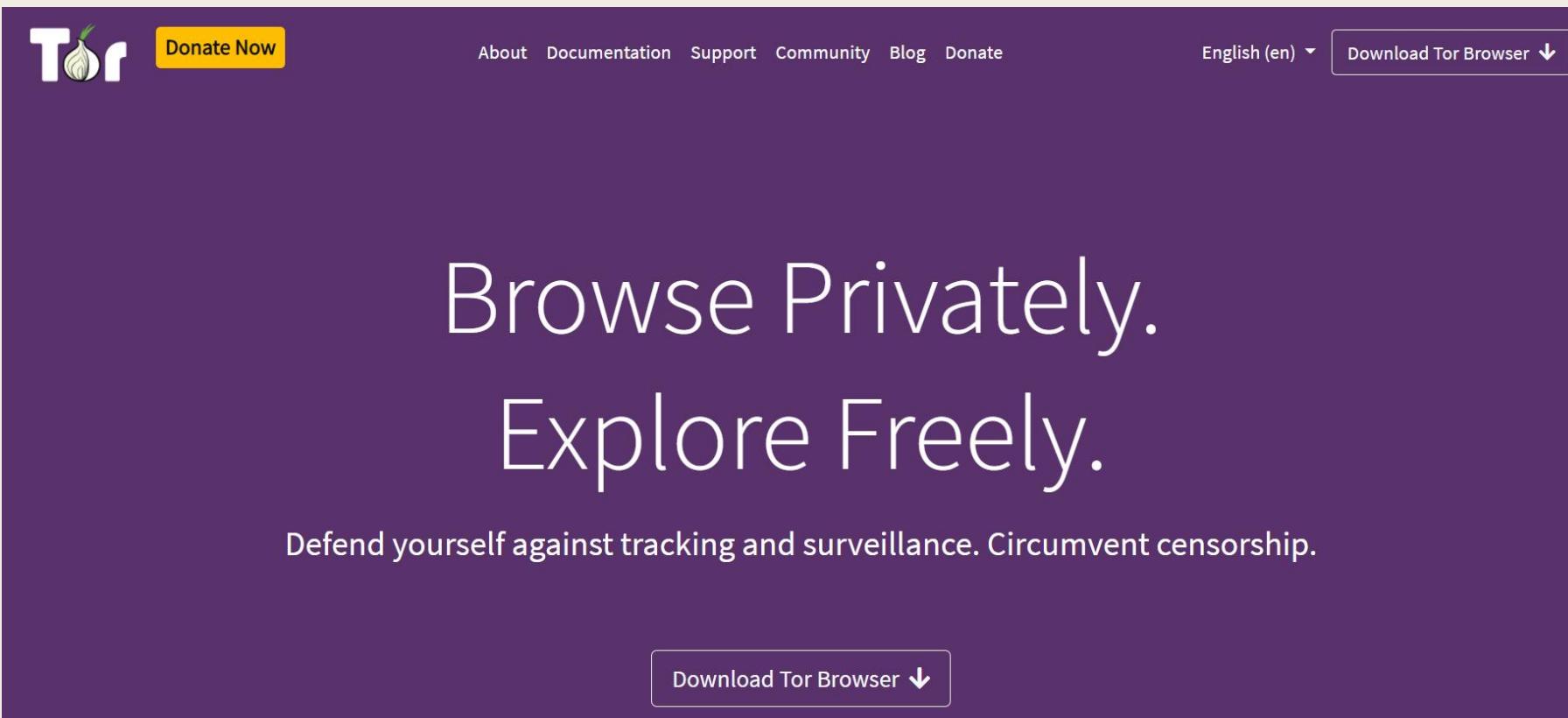
TOR (The Onion Router)



<https://www.eff.org/pages/tor-and-https>

Tor Browser

- <https://www.torproject.org/>



setup Tor & Proxchains in Kali

- apt-get install tor proxchains

- locate proxchains

```
(kali㉿kali)-[~] adding  
$ locate proxchains:  
/etc/proxchains4.conf:
```

- vim /etc/proxchains4.conf

setup Tor & Proxchains in Kali

- Uncomment
 - *dynamic_chain*

```
dynamic_chain info/proxchains4.postinst
#var/lib/dpkg/info/proxchains4.prem
# Dynamic - Each connection will be done via chained proxies
# all proxies chained in the order as they appear in the list
# at least one proxy must be online to play in chain
# (dead proxies are skipped)
# otherwise EINTR is returned to the app
#etc/alternatives/proxchains.1.gz
#strict_chain
#usr/bin/proxchains4
# Strict - Each connection will be done via chained proxies
# all proxies chained in the order as they appear in the list
# all proxies must be online to play in chain
# otherwise EINTR is returned to the app
"
```

- Comment
 - *strict_chain*

- Uncomment
 - *Proxy DNS requests – no leak for DNS data*

```
# Quiet mode (no output from library) lib-pr
#quiet_mode icons/hicolor/48x48/apps/kali-pr
/usr/share/icons/hicolor/scalable/apps/kali
Proxy DNS requests - no leak for DNS data ox
proxy_dns man/man1/proxchains.1.gz
/usr/share/man/man1/proxchains4.1.gz
# set the class A subnet number to use for 
```

- Put socks5 protocol statement on the last line
 - *socks5 127.0.0.1 9050*

```
[ProxyList]
# add proxy here ...
# meanwhile
# defaults set to "tor"
socks4 127.0.0.1 9050
socks5 127.0.0.1 9050
```

setup Tor & Proxchains in Kali

- service tor start

```
[kali㉿kali)-[~]ic chain ... 127.0.0.1:9050 ... content-signature-2.cdn.mozilla.net
└─$ service tor status
chain ... 127.0.0.1:9050 ... firefox.settings.services.mozilla.net
● tor.service - Anonymizing overlay network for TCP (multi-instance-master)
   Loaded: loaded (/lib/systemd/system/tor.service; disabled; vendor preset: disabled)
   Active: active (exited) since Mon 2021-03-29 11:05:10 EDT; 35min ago
     Process: 23738 ExecStart=/bin/true (code=exited, status=0/SUCCESS)
      Main PID: 23738 (code=exited, status=0/SUCCESS)
     CPU: 1ms
[proxy] dynamic chain ... 127.0.0.1:9050 ... dnsleaktest.com:443 ... OK
[proxy] dynamic chain ... 127.0.0.1:9050 ... dnsleaktest.com:443 ... OK
[proxy] dynamic chain ... 127.0.0.1:9050 ... dnsleaktest.com:443 ... OK
```

- proxchains [program] [argument]
 - *proxchains firefox www.dnsleaktest.com*

TECHNIQUES

Enterprise

Reconnaissance

[Home](#) > [Techniques](#) > [Enterprise](#) > [Proxy](#) > [Multi-hop Proxy](#)

Proxy: Multi-hop Proxy

■ <https://attack.mitre.org/techniques/T1090/003/>

Name	Description
APT29	A backdoor used by APT29 created a Tor hidden service to forward traffic from the Tor client to local ports 3389 (RDP), 1433 (SQL), and 1434 (MSSQL) on the victim's machine, enabling full remote access from outside the network. ^[2]
Attor	Attor has used Tor for C2 communication. ^[3]
Dok	Dok downloads and installs Tor via homebrew. ^[4]
FIN4	FIN4 has used Tor to log in to victims' email accounts. ^[5]
GreyEnergy	GreyEnergy has used Tor relays for Command and Control servers. ^[6]
Inception	Inception used chains of compromised routers to proxy C2 communications between them and cloud service providers. ^[7]
Keydnap	Keydnap uses a copy of tor2web proxy for HTTPS communications. ^[8]

Reference

- [HOW TO BECOME ANONYMOUS ONLINE](#)
- [Combination of VPN, Tor And ProxyChain For More Anonymity](#)

HW

- XSS - 30 labs
 - <https://portswigger.net/web-security/cross-site-scripting>
- CSRF - 8 labs
 - <https://portswigger.net/web-security/csrf>
- SSRF - 7 labs
 - <https://portswigger.net/web-security/ssrf>

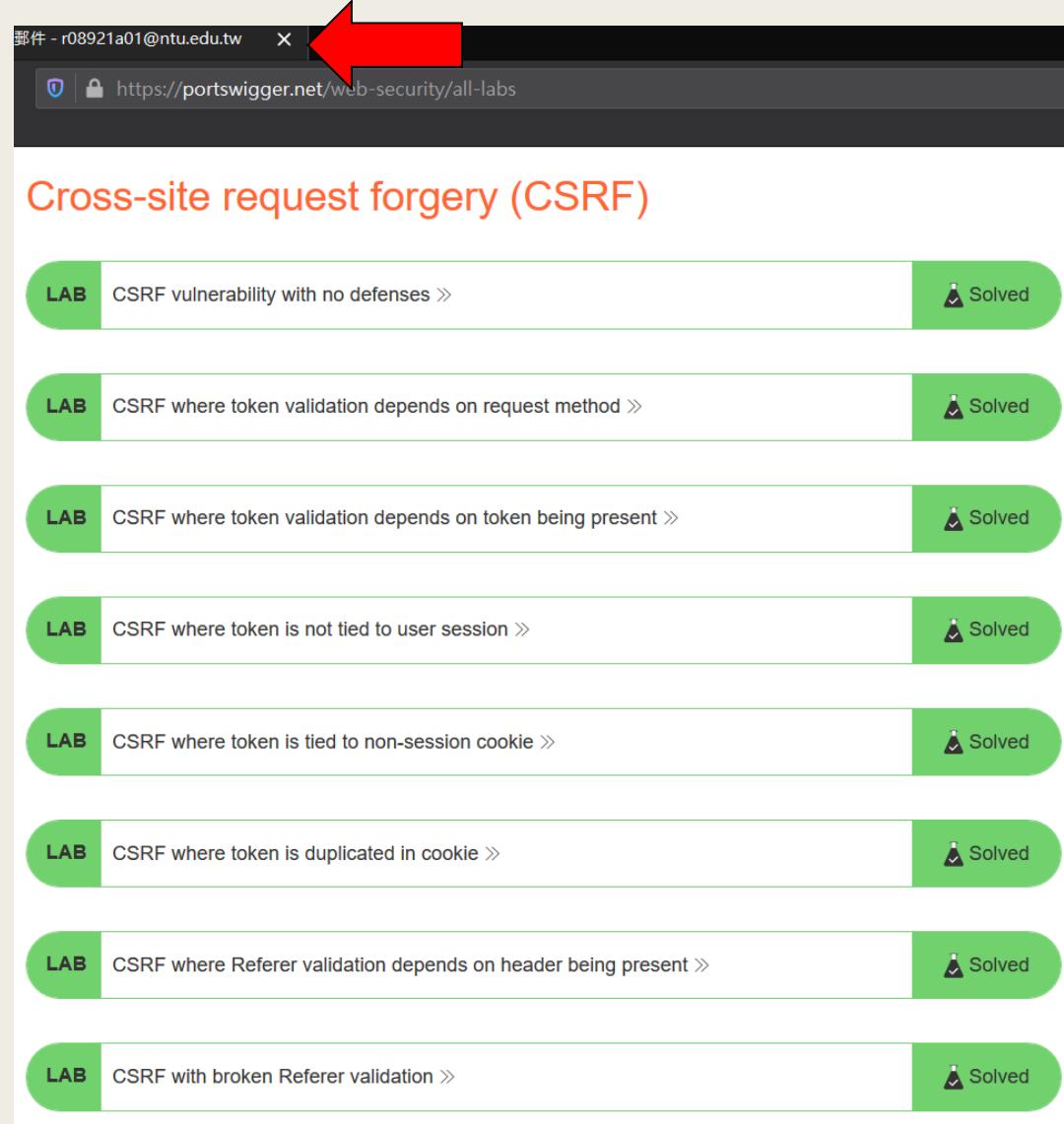
HW

- 上傳 pdf 檔，包含：

- 10pt: Screenshot-01: Payload1 放上學號
 - 10pt: Screenshot-02: Change 欄位放上學號
 - 80pt: Screenshots @ <https://portswigger.net/web-security/all-labs>

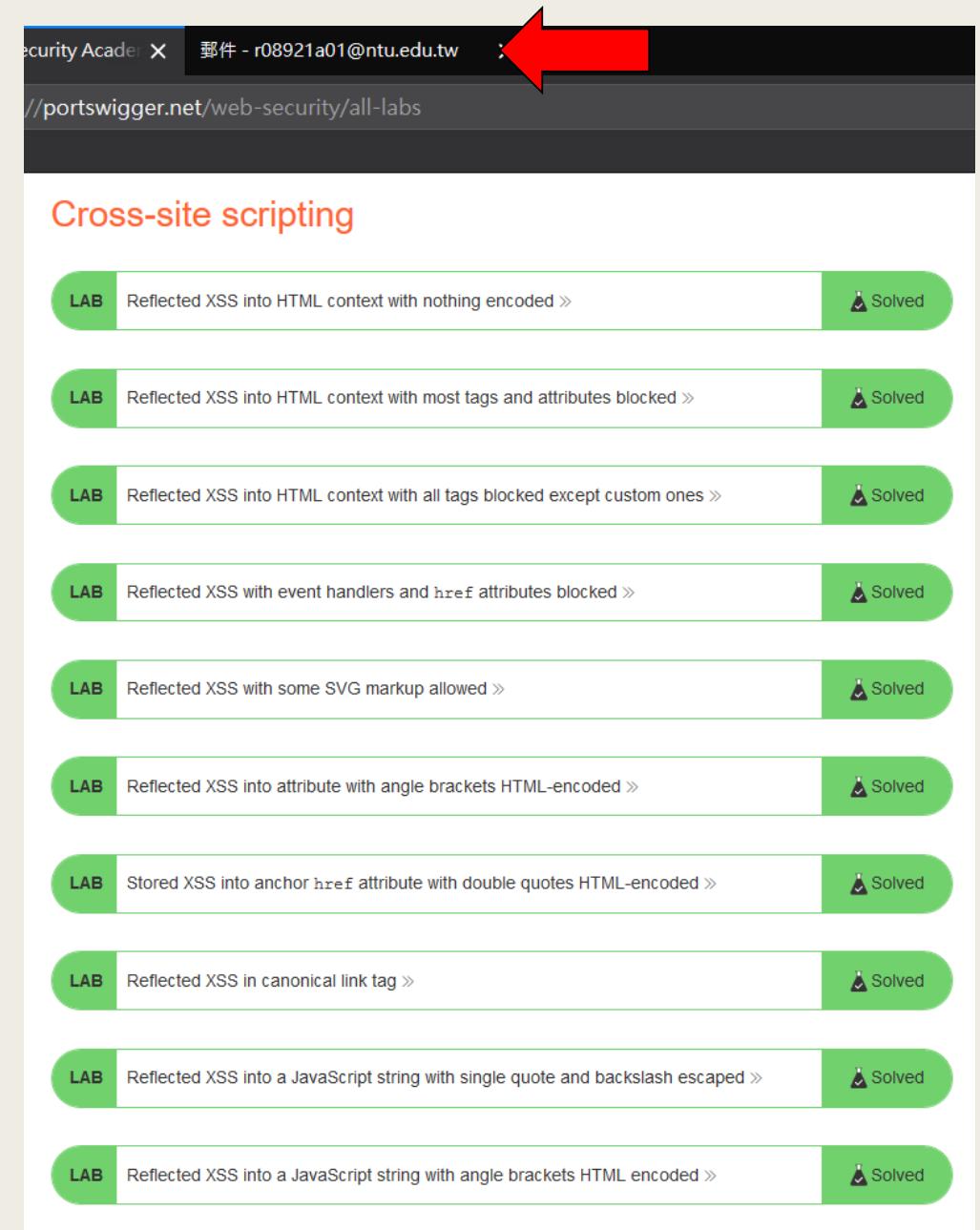
- Lab 總共 45 題，解 20 題得 60pt，之後每多一題得 1pt。

With your Email logged in



A screenshot of a browser window showing a solved lab from the PortSwigger Web Security Academy. The URL in the address bar is <https://portswigger.net/web-security/all-labs>. The page title is "郵件 - r08921a01@ntu.edu.tw". The main content is titled "Cross-site request forgery (CSRF)" and lists nine solved labs, each with a green "Solved" badge:

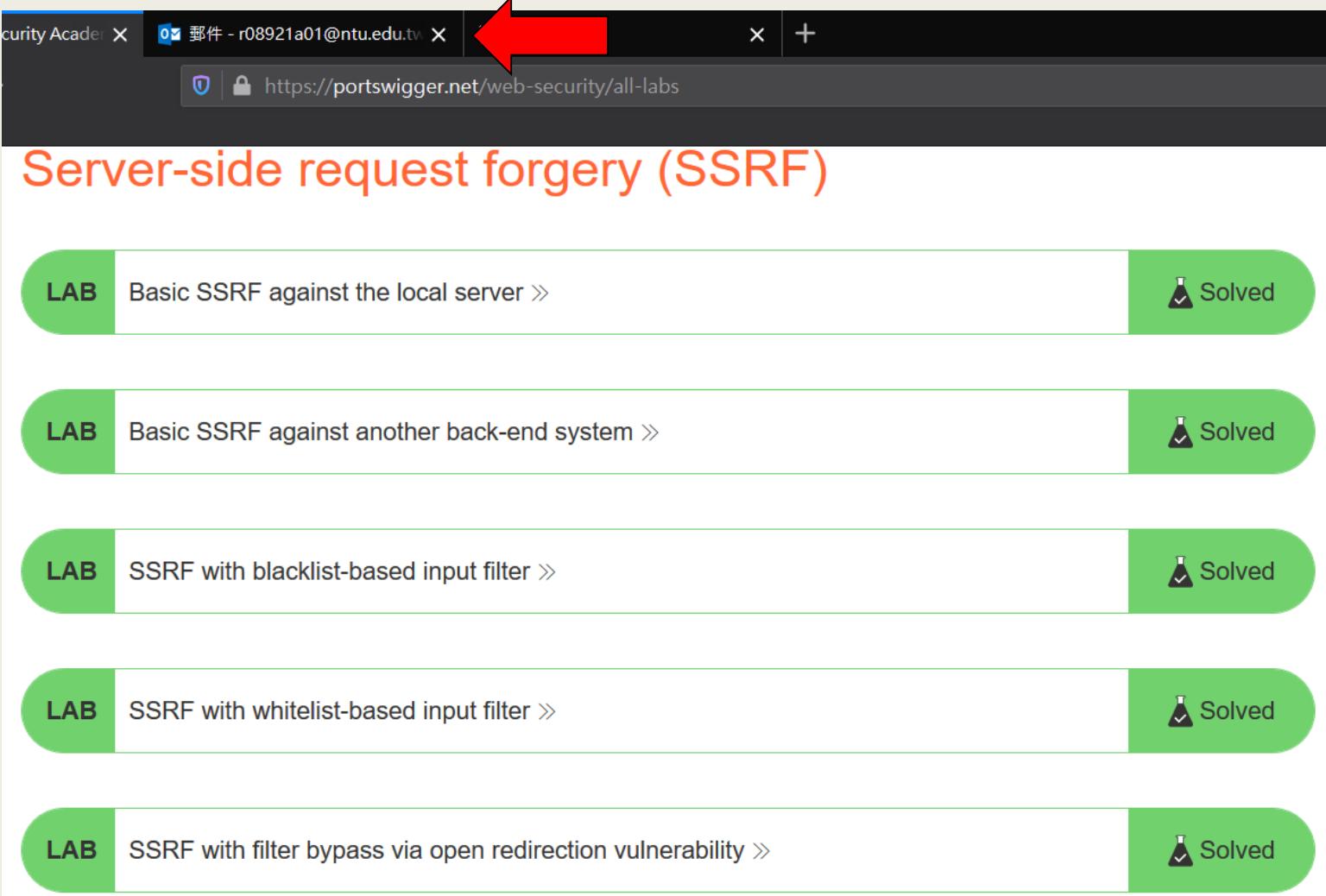
- LAB CSRF vulnerability with no defenses >>
- LAB CSRF where token validation depends on request method >>
- LAB CSRF where token validation depends on token being present >>
- LAB CSRF where token is not tied to user session >>
- LAB CSRF where token is tied to non-session cookie >>
- LAB CSRF where token is duplicated in cookie >>
- LAB CSRF where Referer validation depends on header being present >>
- LAB CSRF with broken Referer validation >>



A screenshot of a browser window showing a solved lab from the PortSwigger Web Security Academy. The URL in the address bar is <https://portswigger.net/web-security/all-labs>. The page title is "security Academy X 郵件 - r08921a01@ntu.edu.tw". The main content is titled "Cross-site scripting" and lists twelve solved labs, each with a green "Solved" badge:

- LAB Reflected XSS into HTML context with nothing encoded >>
- LAB Reflected XSS into HTML context with most tags and attributes blocked >>
- LAB Reflected XSS into HTML context with all tags blocked except custom ones >>
- LAB Reflected XSS with event handlers and `href` attributes blocked >>
- LAB Reflected XSS with some SVG markup allowed >>
- LAB Reflected XSS into attribute with angle brackets HTML-encoded >>
- LAB Stored XSS into anchor `href` attribute with double quotes HTML-encoded >>
- LAB Reflected XSS in canonical link tag >>
- LAB Reflected XSS into a JavaScript string with single quote and backslash escaped >>
- LAB Reflected XSS into a JavaScript string with angle brackets HTML encoded >>

With your Email logged in



A screenshot of a browser window showing a list of SSRF labs from portswigger.net. The browser tabs are "Security Academy" and "郵件 - r08921a01@ntu.edu.tw". A red arrow points to the "Security Academy" tab. The address bar shows "https://portswigger.net/web-security/all-labs". The main content area displays five lab items, each with a green "Solved" status icon.

- LAB Basic SSRF against the local server >> Solved
- LAB Basic SSRF against another back-end system >> Solved
- LAB SSRF with blacklist-based input filter >> Solved
- LAB SSRF with whitelist-based input filter >> Solved
- LAB SSRF with filter bypass via open redirection vulnerability >> Solved