

NETWORK & MULTIMEDIA LAB

WEB SECURITY

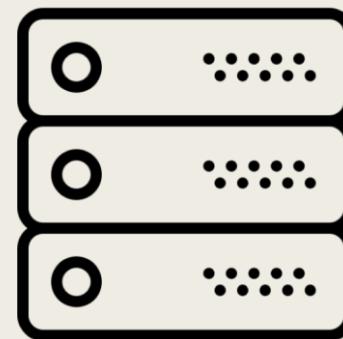
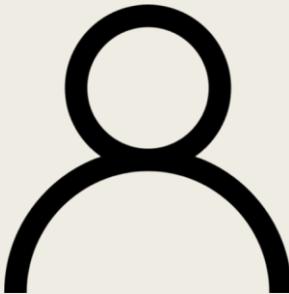
Spring 2021

Outline

- Web Basics
 - HTTP Protocols
 - Cookie and Session
- 3 Attacks
 - CSRF
 - XSS
 - SSRF
- 2 Tools
 - Burp
 - DirBuster

前端/後端

- 瀏覽器
- Chrome, Firefox, Safari
- UI 介面
- HTML, CSS, JS
- 伺服器
- Apache, Nginx
- 提供網頁、處理使用者請求
- JS, Python, PHP, Ruby, Go ...



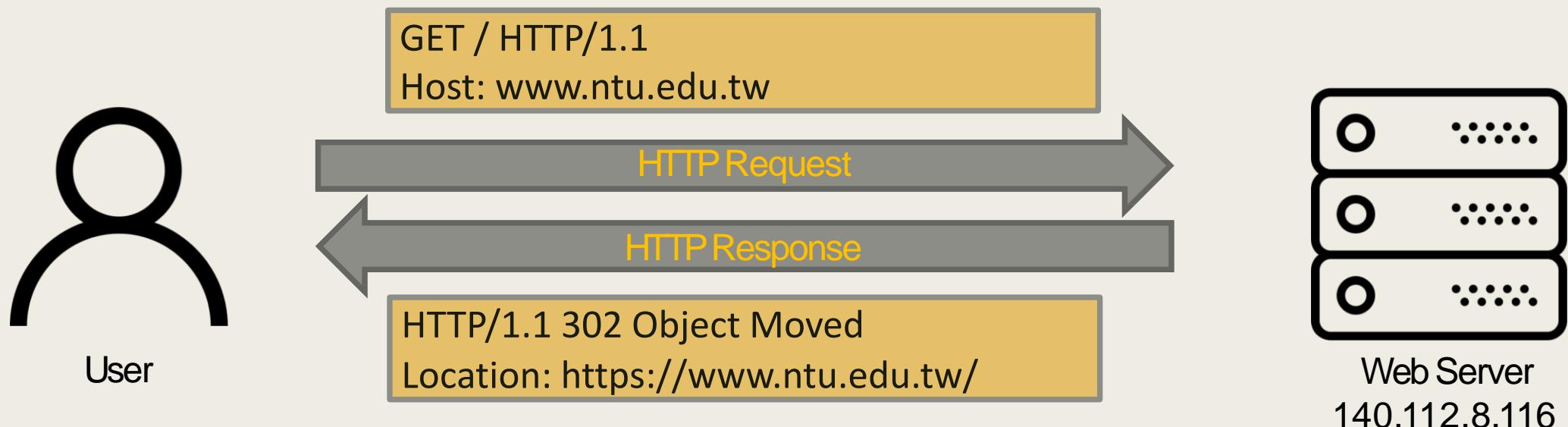
Protocols

當使用者在瀏覽網頁時，瀏覽器通常會使用以下幾種通訊協定跟網頁伺服器傳輸資料

- HTTP
 - 以明文的方式傳送與接收 HTML 等網頁資料，預設是以 TCP 80 Port 進行傳輸
- HTTPS
 - 原則上與 HTTP 相同，但是會以 TLS/SSL 協定將傳輸資料加密，預設是以 TCP 443 Port 進行傳輸
- WebSocket
 - 在使用者與伺服器之間提供一個全雙工(Full-Duplex)的通道，讓伺服器可以主動對使用者進行訊息推播

HTTP

- 瀏覽 <http://www.ntu.edu.tw> 時傳輸的 HTTP 內容



URL (Uniform Resource Locator)

- 統一資源定位符
- 格式：`scheme://authority]path[?query][#fragment]`
- `authority` = [userinfo@]host[:port]
- 例如：<https://www.youtube.com/watch?v=XpC04fMrhd8&t=5s#16w8>

`scheme` → https

`authority` → www.youtube.com

`path` → watch

`query` → v=XpC04fMrhd8&t=5s

`fragment` → 16w8 (不會被傳送至伺服器, 只提供瀏覽器來使用)

Request

Pretty

Raw

\n

Actions ▾

1 GET /watch?v=XpC04fMrhd8&t=5s HTTP/1.1

2 Host: www.youtube.com

HTTP Request

■ 架構

- Request Line
- Request Header
- Empty Line
- Request Body (Optional)

```
POST /cgi-bin/cbdic/gsweb.cgi HTTP/1.1
Host: dict.revised.moe.edu.tw
Connection: keep-alive
Content-Length: 81
Cache-Control: max-age=0
Origin: http://dict.revised.moe.edu.tw
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_3)
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,
Referer: http://dict.revised.moe.edu.tw/cgi-bin/cbdic/gswe
Accept-Encoding: gzip, deflate
Accept-Language: zh-TW,zh;q=0.9,en-US;q=0.8,en;q=0.7,zh-CN
Cookie: LB_new_revised=52588428.20480.0000; _ga=GA1.3.1889
o=e0&cccd=h1YUAg&sec=sec1&qs0=123&clscan=&selectmode=mode1&
```

HTTP Request

■ Request Line

- Request Method: GET, POST, ...
- <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>

■ Request Header

- Host: 網域名稱
- Content-Length: Body 的長度
- Content-Type: Body 的格式
- User-Agent: 使用者的 OS、瀏覽器資訊
- Cookie: 存在使用者瀏覽器中的資料

```
POST /cgi-bin/cbdic/gsweb.cgi HTTP/1.1
Host: dict.revised.moe.edu.tw
Connection: keep-alive
Content-Length: 81
Cache-Control: max-age=0
Origin: http://dict.revised.moe.edu.tw
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_3
Accept: text/html,application/xhtml+xml,application/xml;q=
Referer: http://dict.revised.moe.edu.tw/cgi-bin/cbdic/gswe
Accept-Encoding: gzip, deflate
Accept-Language: zh-TW,zh;q=0.9,en-US;q=0.8,en;q=0.7,zh-CN
Cookie: LB_new_revised=52588428.20480.0000; _ga=GA1.3.1889
o=e0&ccd=h1YUAg&sec=sec1&qs0=123&clscan=&selectmode=mode18
```

HTTP Response

■ 架構

- Status Line
 - Response Header
 - Empty Line
 - Message Body (Optional)

```
HTTP/1.1 200 OK
Date: Sat, 14 Mar 2020 07:47:01 GMT
Server: Apache
X-Frame-Options: http://140.111.34.3, h
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 247
Connection: close
Content-Type: text/html; charset=utf-8
```

```
<html>
<head><meta http-equiv="Pragma" content="no-cache">
<body>

    <table border="0" cellpadding="0" cellspacing="0" width="100%" style="width: 100%; border-collapse: collapse;">
        <tbody><tr>
            <td width="423" background="/cbg.jpg" style="background: url(/cbg.jpg) no-repeat center center; width: 423px; height: 200px;">
        </tr>
    </tbody></table>
    <br>

    no message
</body>
</html>
```

HTTP Response

■ Status Codes

- 1XX: Client 需要繼續處理 (較少見)
- 2XX: 成功
- 3XX: 重新導向
- 4XX: 客戶端錯誤
- 5XX: 伺服器錯誤

```
HTTP/1.1 200 OK
Date: Sat, 14 Mar 2020 07:47:01 GMT
Server: Apache
X-Frame-Options: http://140.111.34.3, h
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 247
Connection: close
Content-Type: text/html; charset=utf-8

<html>
<head><meta http-equiv="Pragma" content="no-cache">
<body>

    <table border="0" cellpadding="0" cellspacing="0" width="100%">
        <tbody><tr>
            <td width="423" background="/checkboxes/checkbox_01.gif" style="text-align: center; vertical-align: middle; padding-top: 10px;">
                </td>
        </tr>
    </tbody></table>
    <br>
    <br>
    no message
</body>
</html>
```

HTTP Response Splitting

CRLF injection (Carriage-Return Line-Feed)

- 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
```

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 ..;		

HTTP Response Splitting

CRLF injection (Carriage-Return Line-Feed)

- 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
```

HTTP Response Splitting

CRLF injection (Carriage-Return Line-Feed)

- Redirect users after successful login:

- `http://localhost/login?page=http%3A%2F%2Flocalhost%2Fcheckout%0D%0A%0D%0A%3Cscript%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>

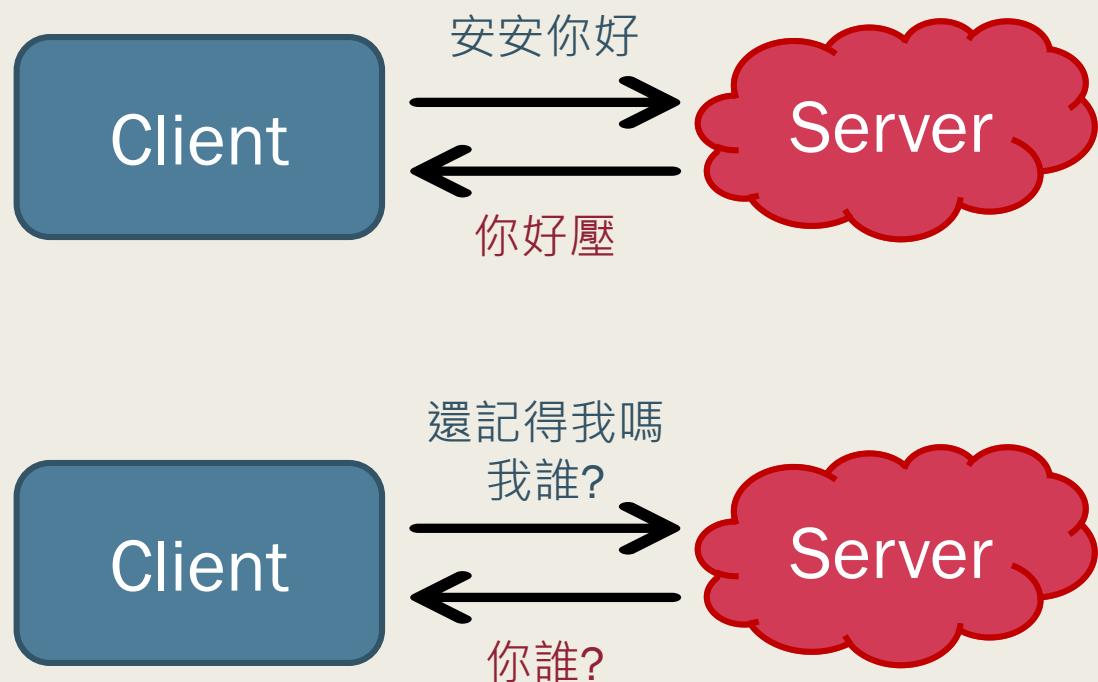
<CRLF>

<script>alert('hello')</script>
```

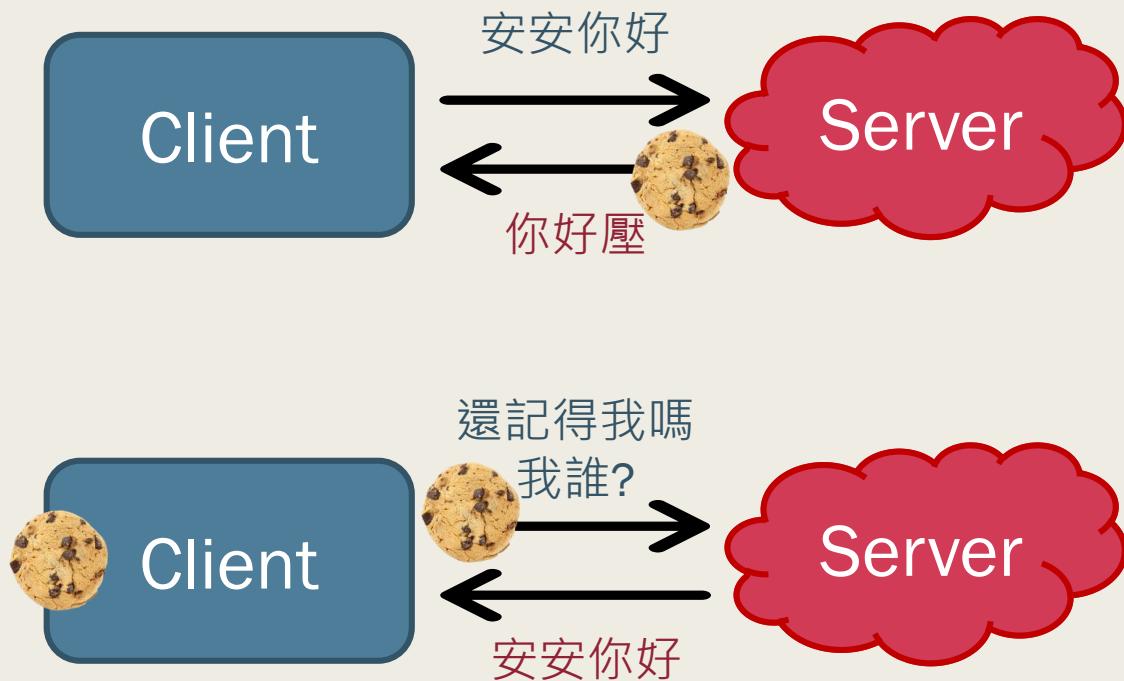
Cookie and Session

- HTTP 協定是無狀態的 (Stateless protocol)
 - 伺服器不會知道用戶上一次做了什麼

- Server
 - 如何判斷請求是來自哪個使用者？
 - 如何判斷使用者已經登入？
 - 如何避免使用者 A 假冒成使用者 B？



Cookie and Session



Cookie and Session

- Cookie 存放 ID
- Session 存放 ID 與對應的資料

Name	Value
SESSIONID	hGjrFF

Client (Cookie)

ID	Data
3ij4Ah	...
0g9XvX	...
hGjrFF	user=foo, login=True
o2AiHk	...

Server (Session)

Pretty Raw Render \n Actions ▾

```
4 X-XSS-Protection: 1;mode=block
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 Vary: Accept-Encoding
9 Set-Cookie: PHPSESSID=541e64be3f14c19d2174558cc1293fef; path=/; secure;
```

Cookie-based Session

- 把 Session 放在 Cookie 裡
 - Data 經過加密 (算法 & Key 未知，難以竄改)
- 優點:
 - 降低 Server 的負擔
- 缺點:
 - Cookie 會被附加在每個 HTTP 請求中，所以無形中增加了流量
 - Cookie 的大小限制在 4KB 左右

Cookie-based Session

```
7 Referer: https://github.com/login
8 Connection: close
9 Cookie: _gh_sess=
hLBIFh7T4fjMx0KIalAaKrsYU71gJ%2B1gbMGeuT%2BiBv3k4aIKjh12hciXQIFXM3UPWzk8Ctck0mGksf
zqo7B3p4ONCIzeoY%2FDzI%2Byom%2F31FnteVWp3QQHsjMLQwEpI0DX10UI%2B5IKXKw%2FLQtJIrTQC3P
gAKzX4b9TLt0Vsh1I95%2BNAgGP%2BGXo0DCIZGx9MRBE7nUbzzga%2BVo%2F8aU0cWW2TYSJMPGgBRymJ
%2B0xp0C4Fs2HZJG4hRLPxq3KRse5MTp0tWe5XqTkQr5zeJsDbx0ku54HhSNYH%2FZII2ZGkwnb0%2BJu
ms00P5i6DqrogHLGiYyx6a%2By6WUp7CjyloQLdEDmbeAf9a3upB1AC7uxL5yivtt2CA5v4CDc%2B0msyH
RlyA%2B3PkHw8xN4wNiS%2Brv%2FluxUlxEmgOPjpGuqXMQysYg4IRw2jqDXKWS8tbOPxA1AYe11Kca%2
FrfaWCLMAOVUqtWTAB4YjuYwFFGtQOSS1izWP5wiD0jWrUCCBcNFKKhgv0QDi%2FrzR8iKtTQwx%2FqW1y
JnA50plowUDEFz0ERkqR9A%2Bawtk%2Fc0xdjY29fgDs%2F611YBza6xia09qL10PP650ZFAD1lazho7J9
vvKNUMKCvssHhFT1112wibfskPJXxrYUboIGjPYM6KfCjHwpR9ZrYnH2tyTyHb1RxnlXM3hBXuGzEJLQ%2
FtHsghrDRxltGnP6RbJJzmw187SG0eVjBHJXbfn9N89FMt91CBqQvRMEbupWSe54Xan8yVoQyCEYSrxt6m
0Nz7IPARQkW04FUoEUbYBgC3VCmWKuQuG%2BhSF%2FcwdCqf%2F89gnJRI1VVyToLGmwrfnbnViri i6Bdi
M%2Fs%2BJYFYUzfQWmNYf4PtE1ZPrWtnK5F%2FP9w3JoS%2Bf4NT7qaI4VsR4bK9fMb1rEn0CPs%2FVqkA
2%2FpNMovPM%2F%2FXDh4A8RuwnCnBK2%2B2b%2FH5U1FJ7CC3RMh0WPotXg%3D%3D--H0RergUryjCdwa
77--VmMguPlwkaaPJFLUU%2B28dA%3D%3D; _octo=GH1.1.458315067.1616567046; logged_in=
```

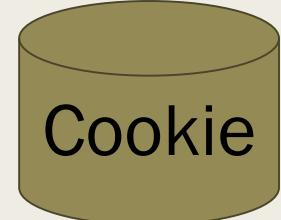
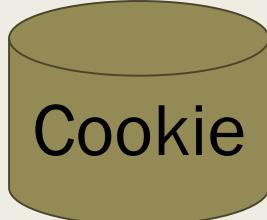
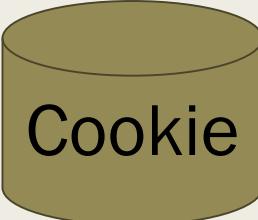
Cookie and Browser

- 每個瀏覽器存放 Cookie 的空間是獨立的

Chrome

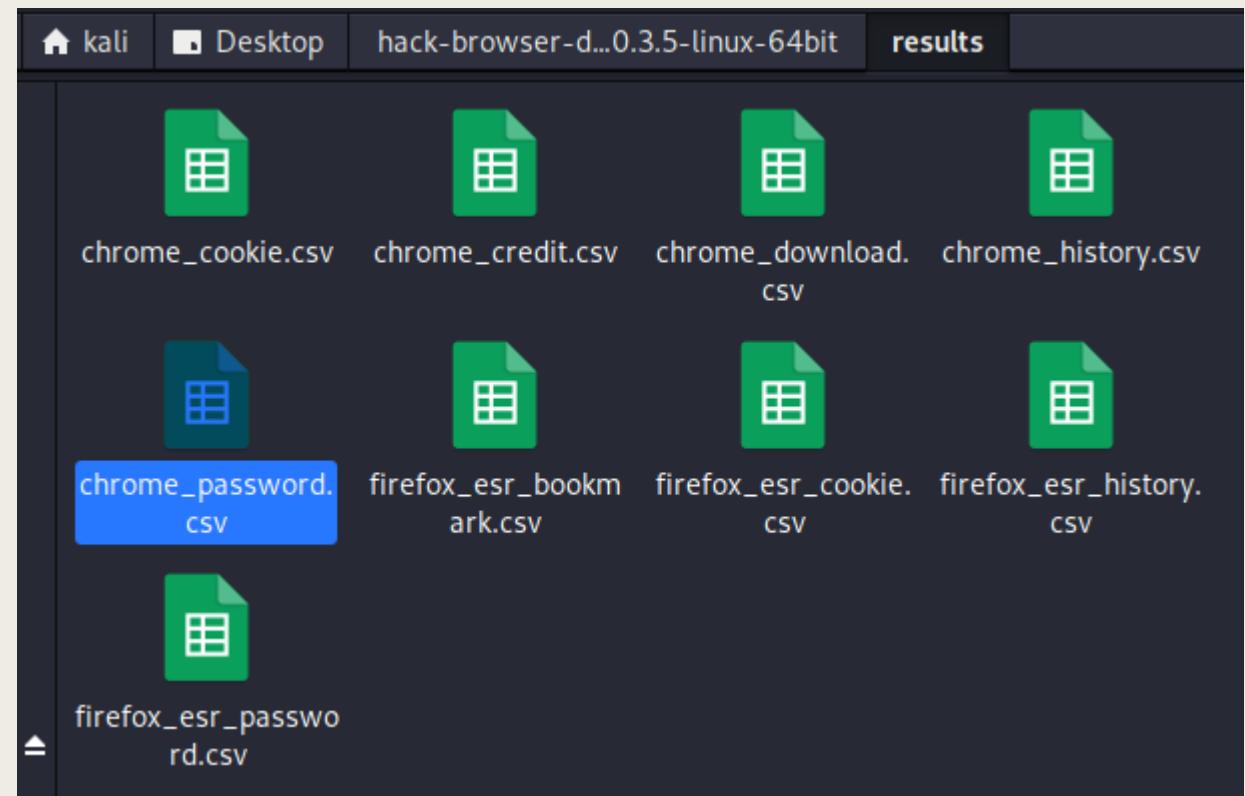
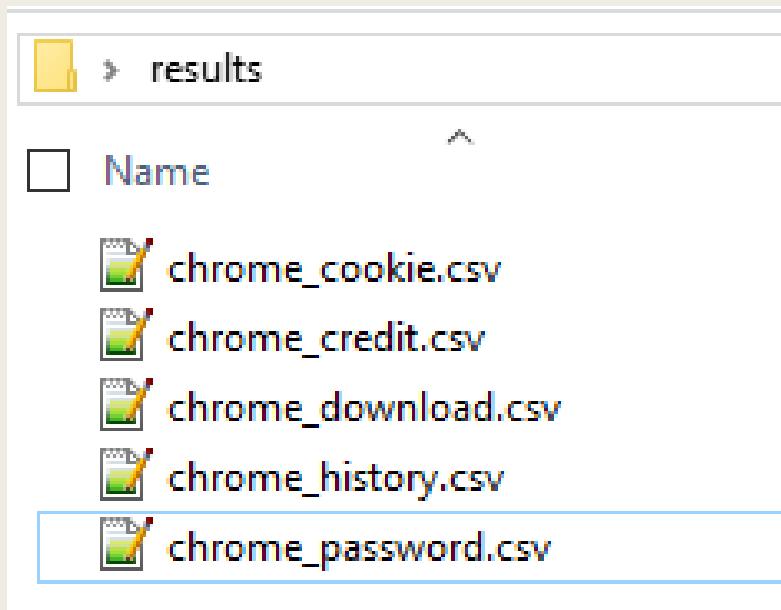
Edge

FireFox



Data and Browser

<https://github.com/moonD4rk/HackBrowserData>



CSRF (Cross-Site Request Forgery)

- 跨站請求偽造
- 一種讓用戶在當前已登入的網站上，執行非本意操作的攻擊手法

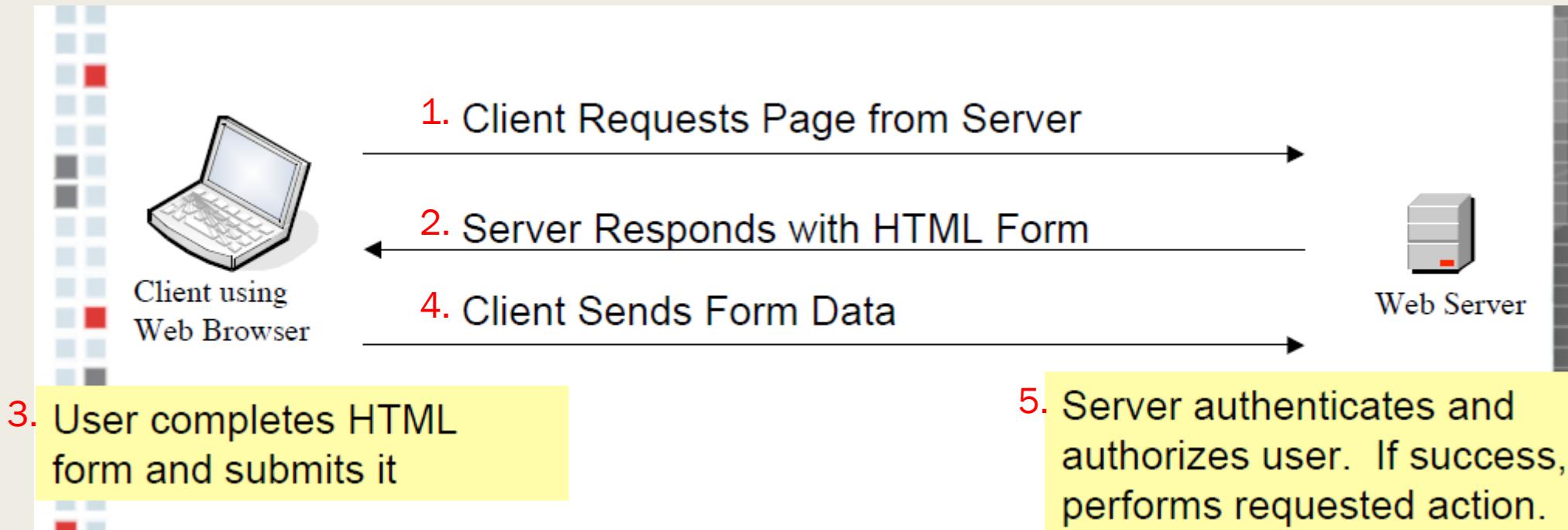
HTTP request

Change email

Email

Update email

HTTP request

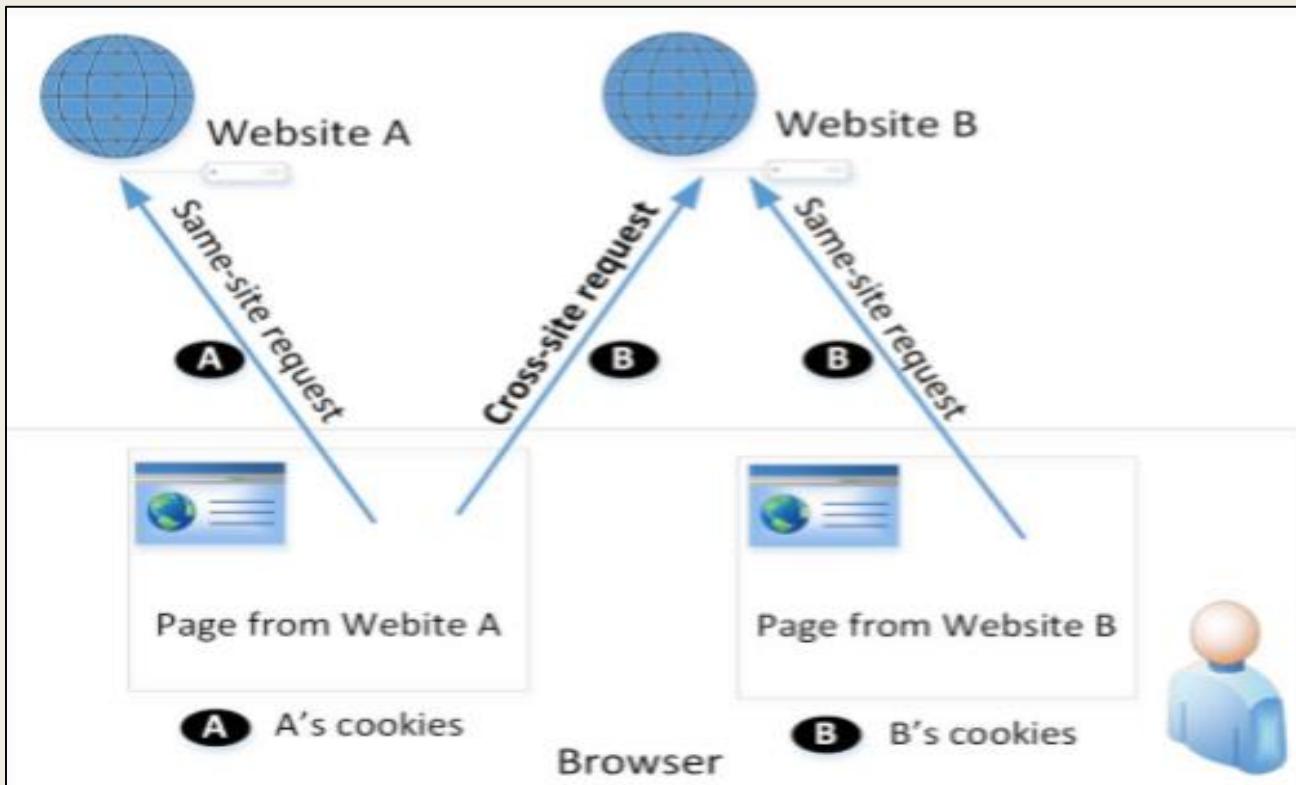


HTML form

```
<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



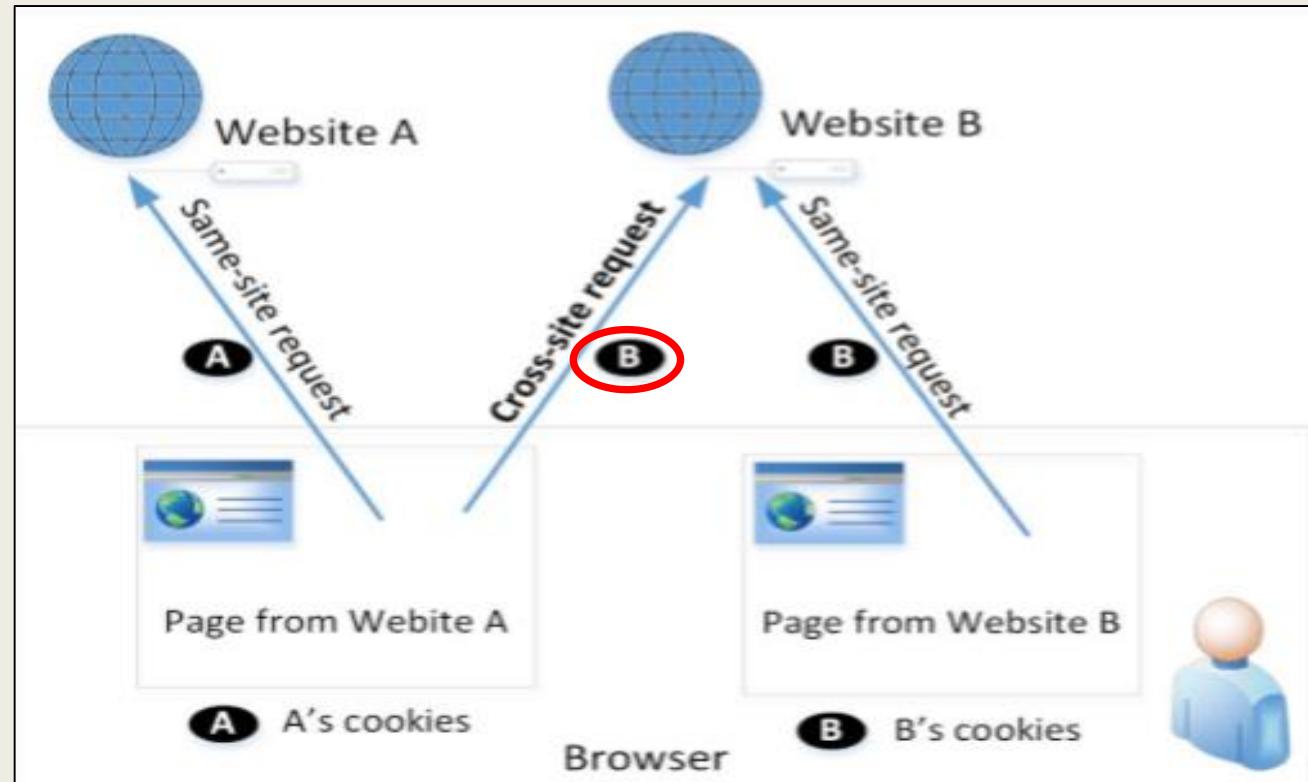
- same-site request:
一個網站的頁面向自己發送請求

```
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">
```

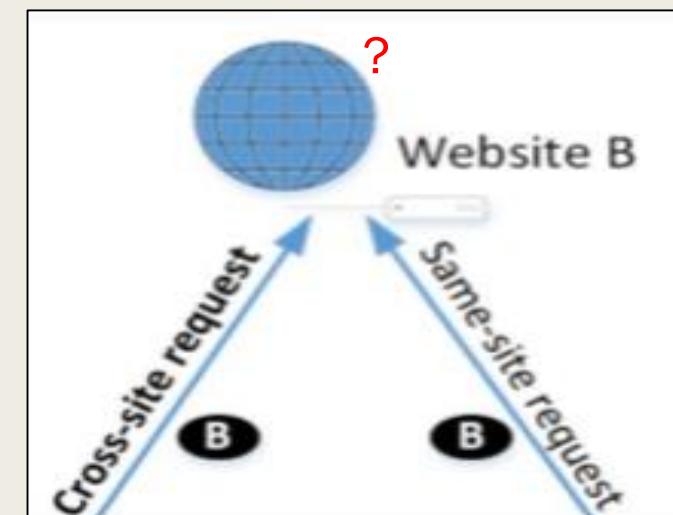
Cookies in Cross-Site Request



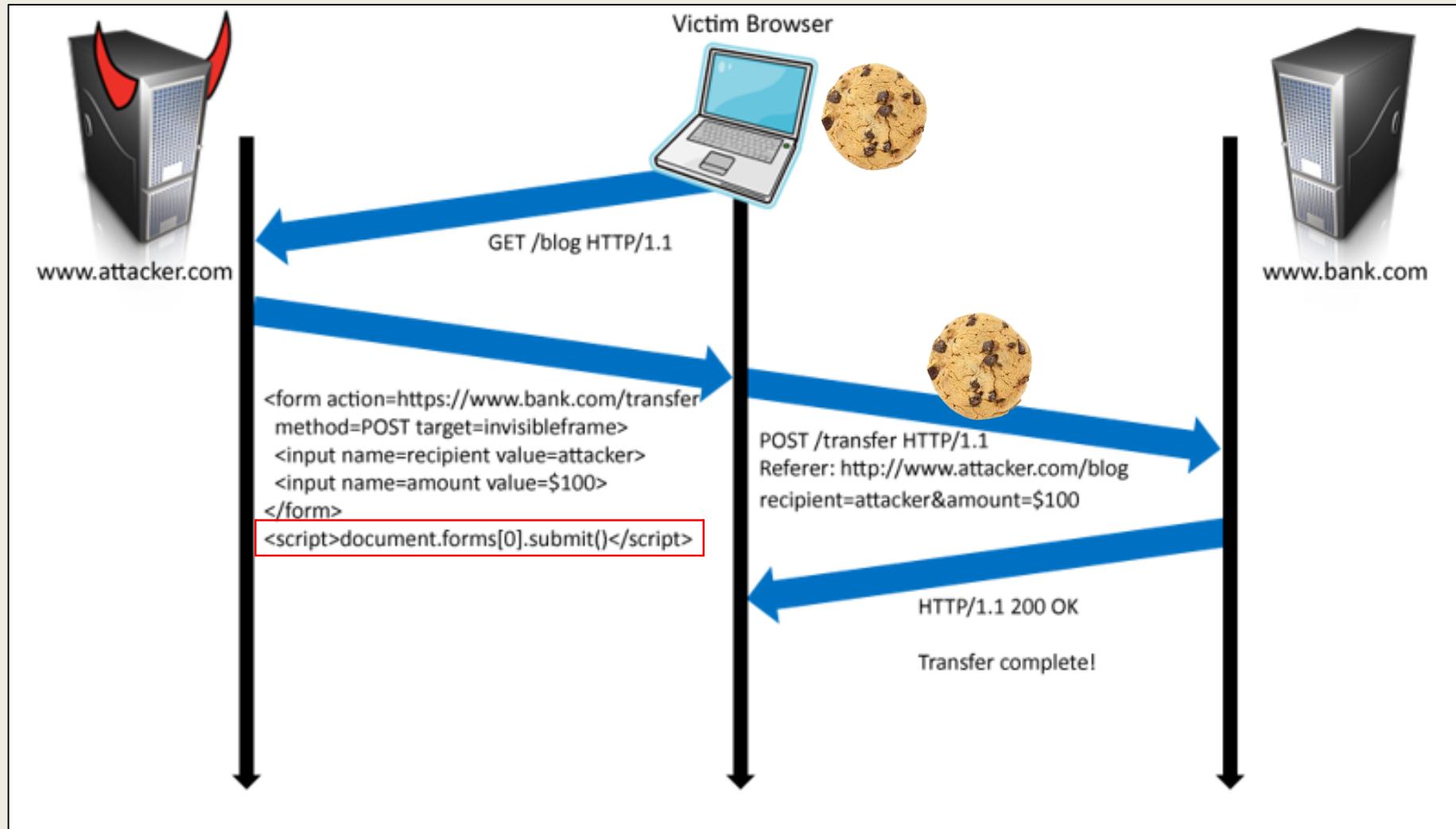
- 當 A.com 的頁面向 A.com 發送請求時，瀏覽器會將 A.com 的 cookies 一併送出
- 當 A.com 的頁面向 B.com 發送請求時，瀏覽器**也會**將 B.com 的 cookies 送出

Problems of Cross-Site Requests

- 因為 cross-site cookie 的特性，server 端無法分辨一個 request 是 same-site request 還是 cross-site request
- 第三方網站可以偽造一個跨站請求，若瀏覽器附上 cross-site cookie，受害網站便會認為這是使用者所發出的請求
- 因此這種攻擊手法稱為 CSRF



CSRF Attack Flow



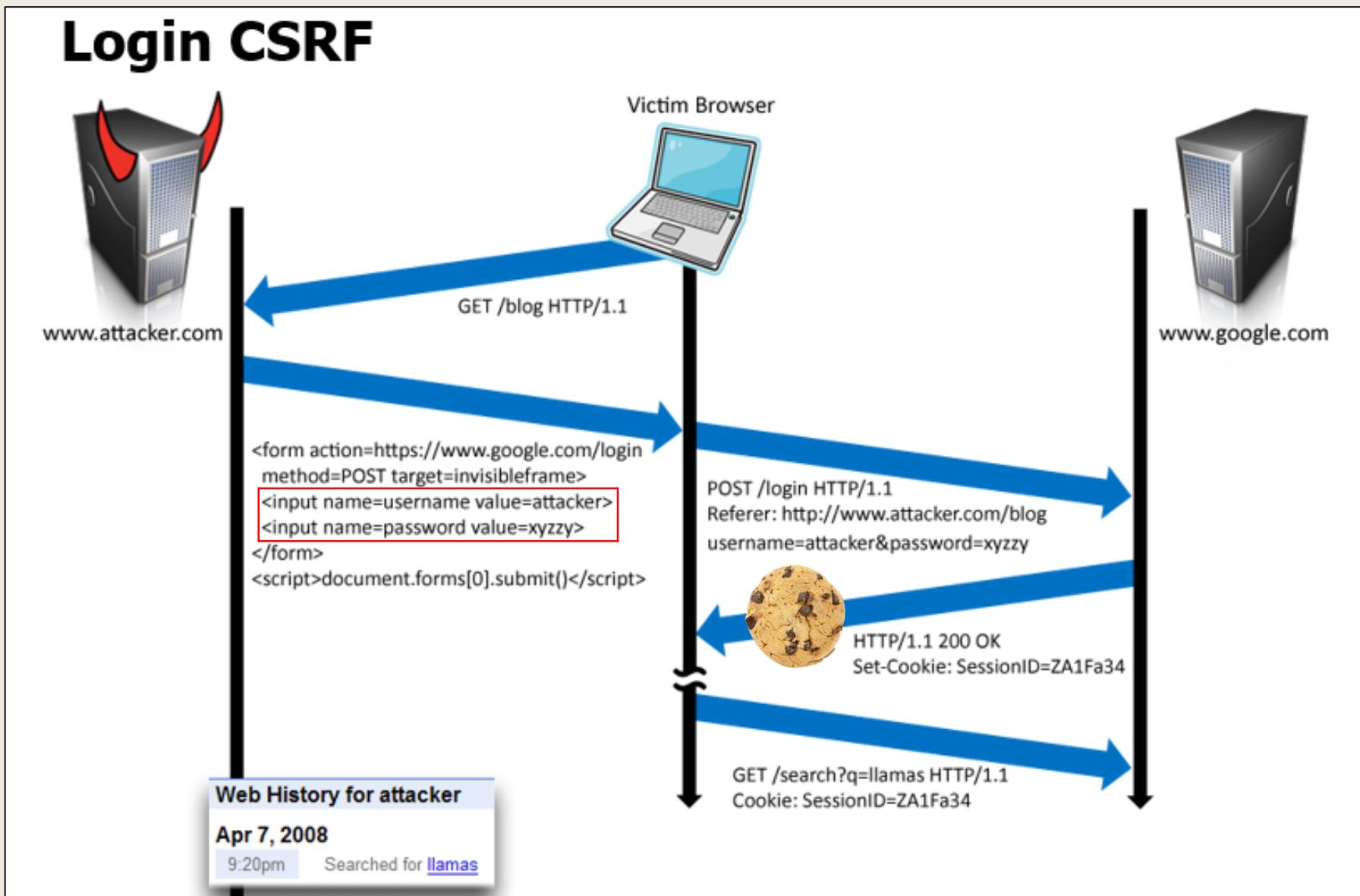
Attacking Scenario

- 受害者必須已經在受害網站上取得授權
(已登入並取得 cookie)
- 攻擊者要想辦法讓受害者造訪惡意網頁
 - a discussion forum
 - HTML email body or attachment
 - or any location that a victim is likely to click while logged into the target site
- 這個惡意網頁包含了偽造的請求，並自動發送給受害網站

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



CSRF Attack Methods (trigger request)

- 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 Attacks on HTTP GET Service

- When you click “Add Friend”

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

- The img 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>
```

Implementation of Cookies

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

Response

Pretty Raw Render \n Actions ▾

```
4 X-XSS-Protection: 1;mode=block
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 Vary: Accept-Encoding
9 Set-Cookie: PHPSESSID=541e64be3f14c19d2174558cc1293fef; path=/; secure;
HttpOnly;HttpOnly;Secure
10 Content-Length: 9199
11 Connection: close
12 Content-Type: text/html
13 Set-Cookie: TS01de58de=
010488152270503alc9226c41fbac46cf2196a4fe8eb2b7ale172al7e2ed84c7d3efc30795668d076c6cf8ac67
d38eba36a718f64029d217685d58a3884c6865c47435809c; Path=/; Domain=.ceiba.ntu.edu.tw
14 Set-Cookie: TS13740f90027=
08eeelde4aab2000d617f409115cb3c20c27f1c315c56e66dc0d8433fa77be5b608759a4b87871be080de340fb
11300050075491e94c4e357ec6fd10f89915ff6e28adeee7a5b8e34b9f2414af1544268d60129fb2241219faa3
a97275781eb2;Path=/
```

Implementation of Cookies

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

Secure Optional

A secure cookie is only sent to the server when a request is made with the `https:` scheme.
(However, confidential information should never be stored in HTTP Cookies, as the entire mechanism is inherently insecure and doesn't encrypt any information.)

HttpOnly Optional

`<script>alert(document.cookie)</script>`

Forbids JavaScript from accessing the cookie, for example, through the `Document.cookie` property. Note that a cookie that has been created with `HttpOnly` will still be sent with JavaScript-initiated requests, e.g. when calling `XMLHttpRequest.send()` or `fetch()`. This mitigates attacks against cross-site scripting (XSS).

Implementation of 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.

Implementation of Cookies

The screenshot shows a browser developer tools window displaying the cookies for the domain `ceiba.ntu.edu.tw`. The list includes:

- `.ceiba.ntu.edu.tw | TS01de58de`
- `.ntu.edu.tw | __utma`
- `.ntu.edu.tw | __utmc`
- `.ntu.edu.tw | __utmz`
- `.ntu.edu.tw | _ga`
- `.ntu.edu.tw | _gid`
- `.ntu.edu.tw | citrix_ns_id`
- `ceiba.ntu.edu.tw | PHPSESSID`

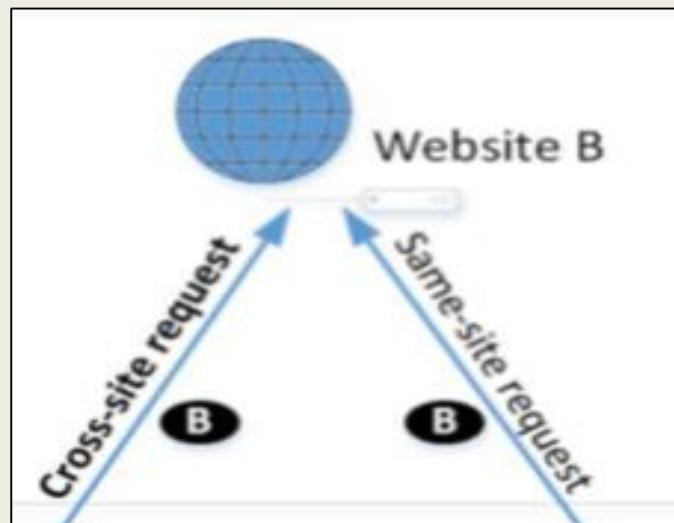
For the `PHPSESSID` cookie, the details are as follows:

- 值**: dc49e245eae4c34dfe0b5dfb9a8efc16
- 網域**: ceiba.ntu.edu.tw
- 路徑**: /
- 有效期**: Wed Mar 23 2022 02:34:22 GMT+0800 (台北標準時間)
- SameSite**: SameSite

At the bottom, there are checkboxes for **HostOnly**, **Session**, **Secure**, and **HTTP Only**.

Fundamental Causes of CSRF

- server 端無法分辨一個 request 是 same-site request (Trusted) 還是 cross-site request (Not Trusted)
- 然而，瀏覽器知道一個 request 是 same-site 還是 cross-site



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)*

Countermeasures

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
```

Countermeasures

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

Countermeasures

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.

Countermeasures

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

Countermeasures

3. Secret Token

a) Server 端生成的 CSRF token

- 產生：Server
- 儲存：Server
- 在 form 裡新增一個 name='csrftoken' value='<亂碼>'，比對 Server 端的 <亂碼>
- 漏洞：若 csrftoken 不是亂碼，攻擊者可以先發一個 request 取得 csrftoken

Countermeasures

3. Secret Token

b) Server 端生成的 Double Submit Cookie

- 產生：Server
- 儲存：Client
- 一樣在表單放 CSRF token，但這次參照值不是存在 Server 裡，而是存在 cookie 裡
- 這方法利用的是攻擊者無法取得 cookie 的值，因此偽造的表單中的 CSRF token 不會和 cookie 裡的值一樣
- 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。

Countermeasures

3. Secret Token

c) Client 端生成的 Double Submit Cookie

- 產生：Client
- 儲存：Client

- 前面提的 Double Submit Cookie，是由 Server 產生、存在 Client 的 Cookie
- 但由於單頁應用 (single-page application) 在拿取 CSRF token 會有困難，可以改成 Client 端生成
- 此 cookie 只是要確保攻擊者無法取得，沒有包含任何敏感資訊，所以不避擔心安全性考量

Summary of CSRF

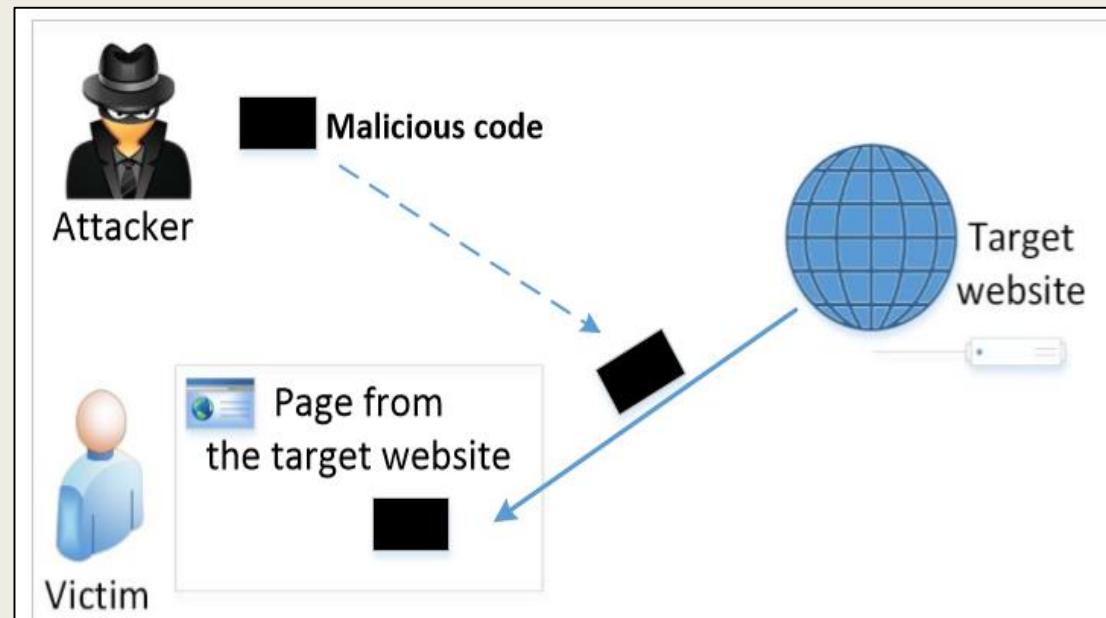
- Cross-site requests vs. Same-site requests
- How does a hacker launch CSRF attack?
- The fundamental cause of the CSRF vulnerability
- Countermeasures against CSRF attack

XSS (Cross-Site Scripting)

- 跨網站指令碼，屬於代碼注入的一種

Definition: XSS

- 它允許惡意使用者將程式碼注入到網頁上，使其他使用者在觀看網頁時受到影響
(執行程式碼)



What hackers can do with XSS attacks?

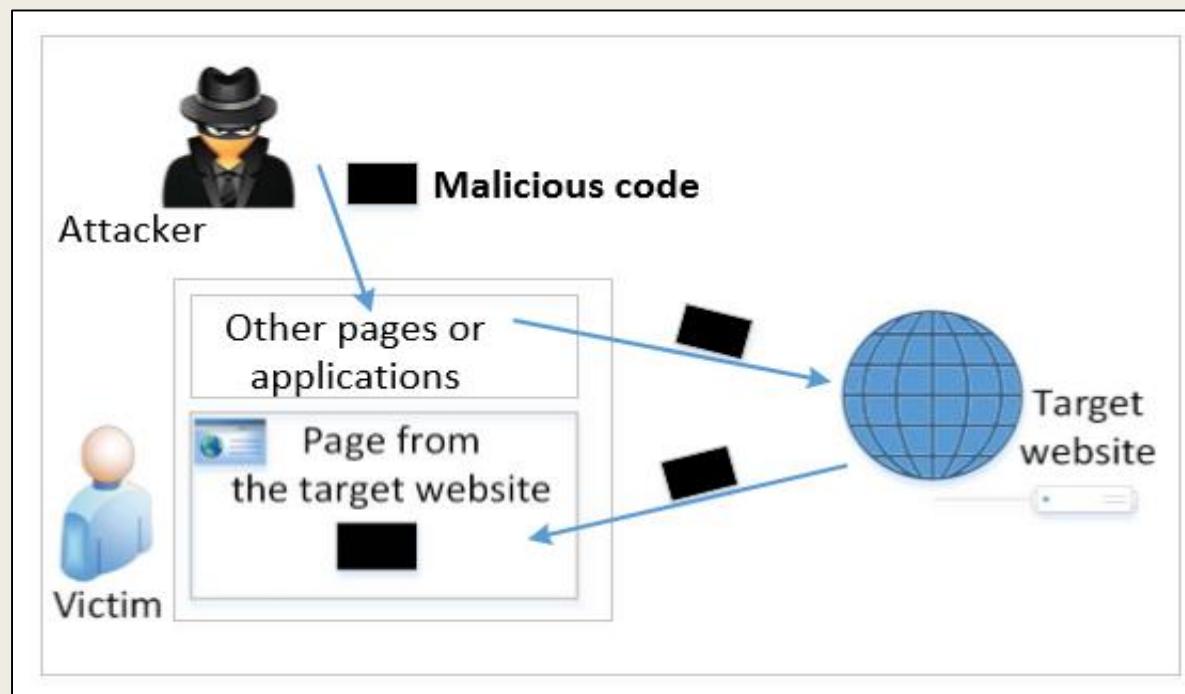
- Forge user requests
- Steal cookies
- Play a sound
- Get user-agent string
- Man-in-the-browser
- Get form values / HTML contents
- Fake notifications
- Tabnabbing (一種釣魚攻擊的手法)

Types of XSS Attacks

- Reflected XSS Attack
 - *Non-persistent*
- Stored XSS Attack
 - *Persistent*

Reflected XSS Attack

- 攻擊者構造出特殊的 URL，其中包含惡意代碼。
- 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。



Reflected XSS Attack

- Assume a vulnerable service on the website :
<http://www.example.com/search?input=word>, where word is provided by the users.

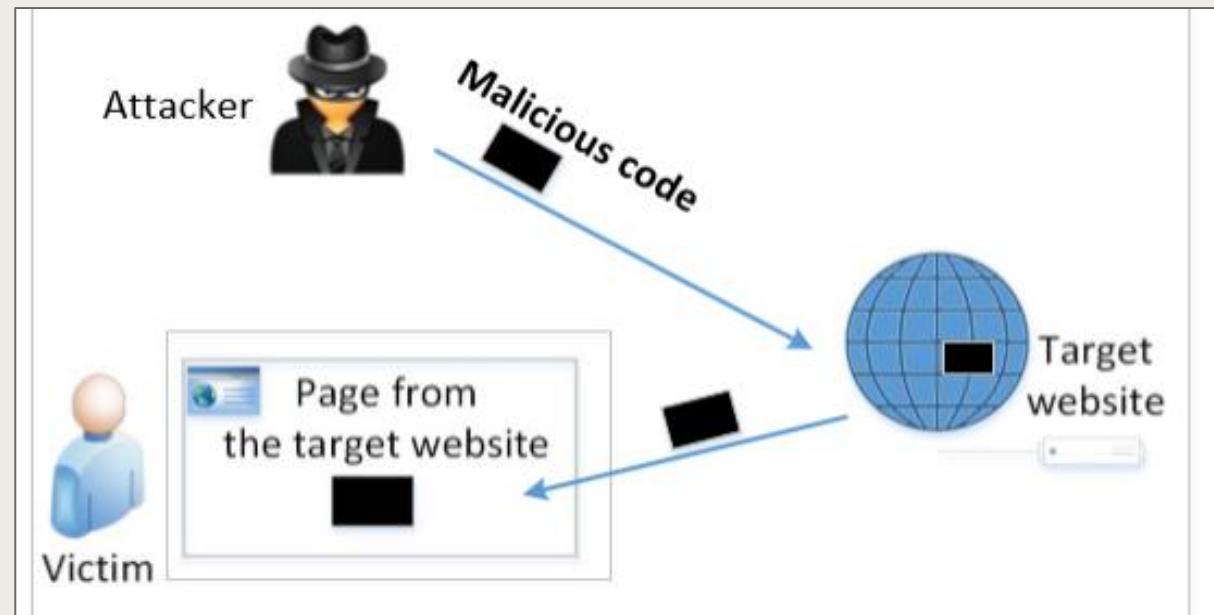


你搜尋了 **XXX** !

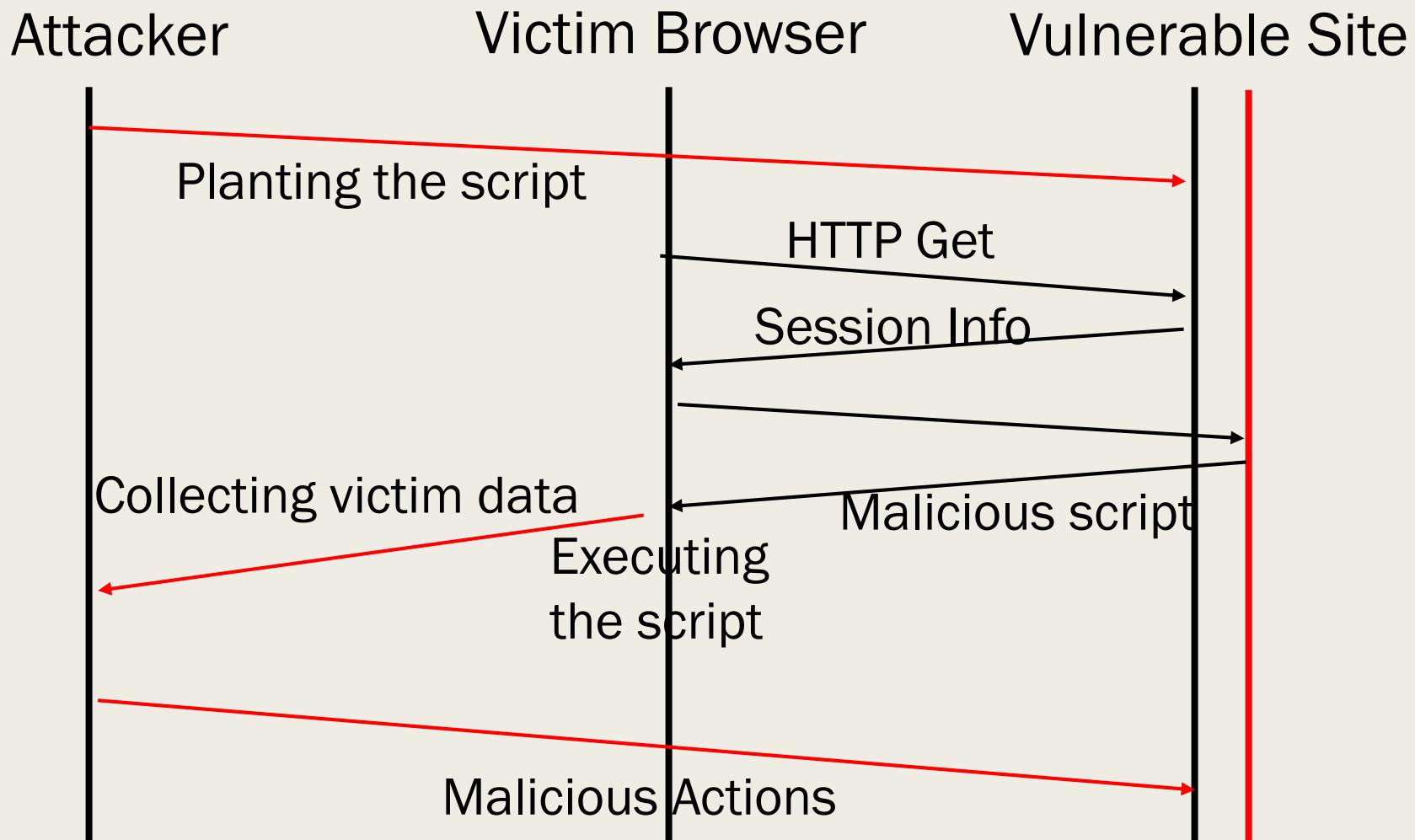
- Now the attacker sends the following URL to the victim and tricks him to click the link: [http://www.example.com/search?input=<script>alert\(1\);</script>](http://www.example.com/search?input=<script>alert(1);</script>)

Stored XSS Attack

- 攻擊用戶打開目標網站時，網站服務端將惡意代碼從資料庫取出，拼接在 HTML 中返回給瀏覽器。
- 著將惡意代碼提交到目標網站的資料庫中。
- Example :
 - 社群網站個人頁面
 - 留言



Stored XSS Attack Flow



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
- 瀏覽器無法分辨這些 Script 的來源，認為這就是原網站的功能

Countermeasures

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

1. Input Filtering Approach

- 移除所有 user input 的 script/code
- 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

```

```
- 用 open-source libraries 過濾 JavaScript code
 - Example: jsoup

2. Output Filtering Approach

- 移除所有 user input 的 script/code
- 資料庫中保存原始的 user input，輸出到網頁時才做過濾

Notes about Filtering Approach

- 這種方法要開發者處理所有 user input 來源
(query parameters, body parameters of POST request, HTTP headers)

3. 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

4. Web Application Firewall (WAF)

- 網站應用防火牆 (WAF) 會自動幫你過濾惡意的 input
(包含 path, HTTP headers, HTML tag patterns 和 Javascript, SQL patterns)



wafw00f

wafw00f

```
(kali㉿kali)-[~] Seq=1 Ack=406 Win=6912 Len=1138 TSval=27795 TSecr=3502769422
└─$ wafw00f http://192.168.56.105:64128 Len=0 TSval=3502769429 TSecr=27795
HTTP/1.1 200 OK (Text/HTML)
4424 -> 80 [ACK] Seq=406 Ack=1144 Win=64128 Len=0 TSval=3502769430 TSecr=27795
4424 -> 80 [FIN, ACK] Seq=406 Ack=1144 Win=64128 Len=0 TSval=3502769430 TSecr=27795
0 -> 4424 [FIN, ACK] Seq=1144 Ack=407 Win=6912 Len=0 TSval=27795 TSecr=3502769430
4424 -> 80 [ACK] Seq=407 Ack=1145 Win=64128 Len=0 TSval=3502769430 TSecr=27795
no has 192.168.56.105 Ts:1 192.168.56.102
92.168.56.105 is ,,
  08:00:27:b5:d2:bb 404 Hack Not Found
local Master Admin|User|Guest| /METASPLOITABLE, Workstation, Server, Print Queue Server, Domain/Workgroup Admin|User|Guest WORKGROUP, NT Workstation, Domain Enum
SEARCH * HTTP/1.1
SEARCH * *====*
SEARCH * HTTP/1.1
405 Not Allowed
403 Forbidden
502 Bad Gateway
500 Internal Error
S
~ WAFW00F : v2.1.0 ~
The Web Application Firewall Fingerprinting Toolkit

[*] Checking http://192.168.56.105
[+] Generic Detection results:
[-] No WAF detected by the generic detection
[~] Number of requests: 7

(kali㉿kali)-[~]
```

wafw00f

No.	Time	Source	Destination	Protocol	Length	Info
6	2.689500872	192.168.56.102	192.168.56.105	HTTP	490	GET / HTTP/1.1
10	2.696716867	192.168.56.105	192.168.56.102	HTTP	71	HTTP/1.1 200 OK (text/html)
18	2.699160408	192.168.56.102	192.168.56.105	HTTP	652	GET /?a=%3Cscript%3Ealert%28%22XSS%22%29%3B%3C%2Fscript%3E&b=UNION+SELECT+ALL+FROM+information_schema+AND+27+OR+SLEEP%285%29+OR+27 HTTP/1.1
22	2.706476690	192.168.56.105	192.168.56.102	HTTP	71	HTTP/1.1 200 OK (text/html)
30	2.742215927	192.168.56.102	192.168.56.105	HTTP	490	GET / HTTP/1.1
34	2.749716437	192.168.56.105	192.168.56.102	HTTP	71	HTTP/1.1 200 OK (text/html)
42	2.751123815	192.168.56.102	192.168.56.105	HTTP	490	GET / HTTP/1.1
46	2.757852356	192.168.56.105	192.168.56.102	HTTP	71	HTTP/1.1 200 OK (text/html)
54	2.759607221	192.168.56.102	192.168.56.105	HTTP	450	GET /?s=%3Cscript%3Ealert%28%22XSS%22%29%3B%3C%2Fscript%3E HTTP/1.1
58	2.766298613	192.168.56.105	192.168.56.102	HTTP	71	HTTP/1.1 200 OK (text/html)
66	2.768073052	192.168.56.102	192.168.56.105	HTTP	397	GET / HTTP/1.1
70	2.775460334	192.168.56.105	192.168.56.102	HTTP	71	HTTP/1.1 200 OK (text/html)
78	2.778057329	192.168.56.102	192.168.56.105	HTTP	471	GET /?s=UNION+SELECT+ALL+FROM+information_schema+AND+27+OR+SLEEP%285%29+OR+27 HTTP/1.1
82	2.785505753	192.168.56.105	192.168.56.102	HTTP	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

File Upload

mnt d Codes VPhysics HW1

Name

sol.py

NTU CEIBA - 國立臺灣大學 非同步課程：普通物理學甲上 - Mozilla Firefox

File Edit View History Bookmarks Tools Help

NTU CEIBA - 國立臺灣大學 NTU CEIBA - 國立臺灣大學

https://ceiba.ntu.edu.tw/modules/index.php?csn=87867f&default_fun=hw¤t_lang=chi

國立台灣大學 National Taiwan University

普通物理學甲上

同學，歡迎進入 Ceiba 系統

作業區

名稱 程式作業 hw1

作業說明 VP1: Simple Start for Vpython, Proj

請大家先自行觀看教學影片：https://www.youtube.com/playlist?list=PLxowpOHFnGyPYejTm6QGmZUEbnZAsyJA_

網頁版作業說明：<https://github.com/janice-cat/GenPhys2019FALL/tree/master/hw1>

相關檔案 檔案

相關網址 <https://github.com/janice-cat/GenPhys2019FALL/tree/master/hw1>

項目 個人

OK

Summary of XSS

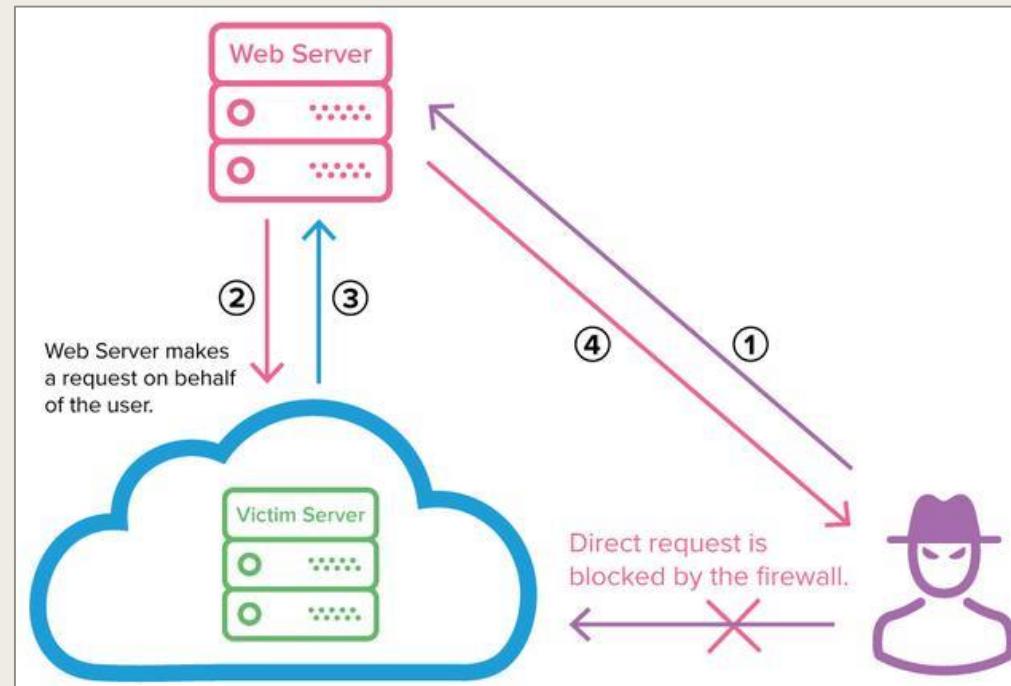
- Two types of XSS attacks
- How to launch XSS attacks
- Countermeasures against XSS attacks

The best defense of XSS?

- Define trust boundaries
- Validate and sanitize untrusted data

Server-Side Request Forgery (SSRF)

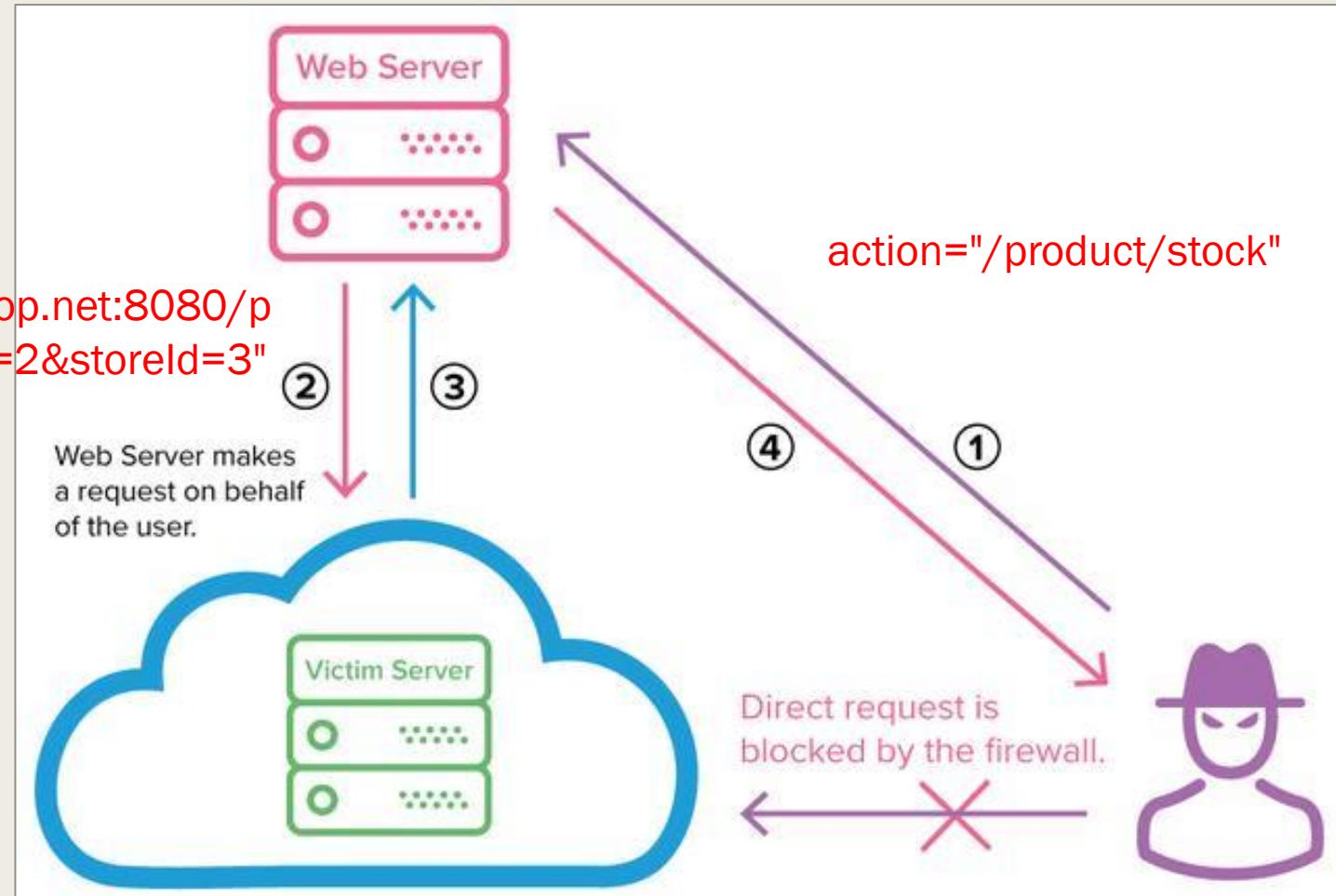
- 服務器端請求偽造，以伺服器的身份發送一條構造好的請求
- 很多 web 應用都提供了從其他的伺服器上獲取數據的功能
- SSRF 可讓攻擊者的請求重定向到防火牆後的內部網路或本地主機



<https://portswigger.net/web-security/ssrf/lab-basic-ssrf-against-localhost>

Server-Side Request Forgery (SSRF)

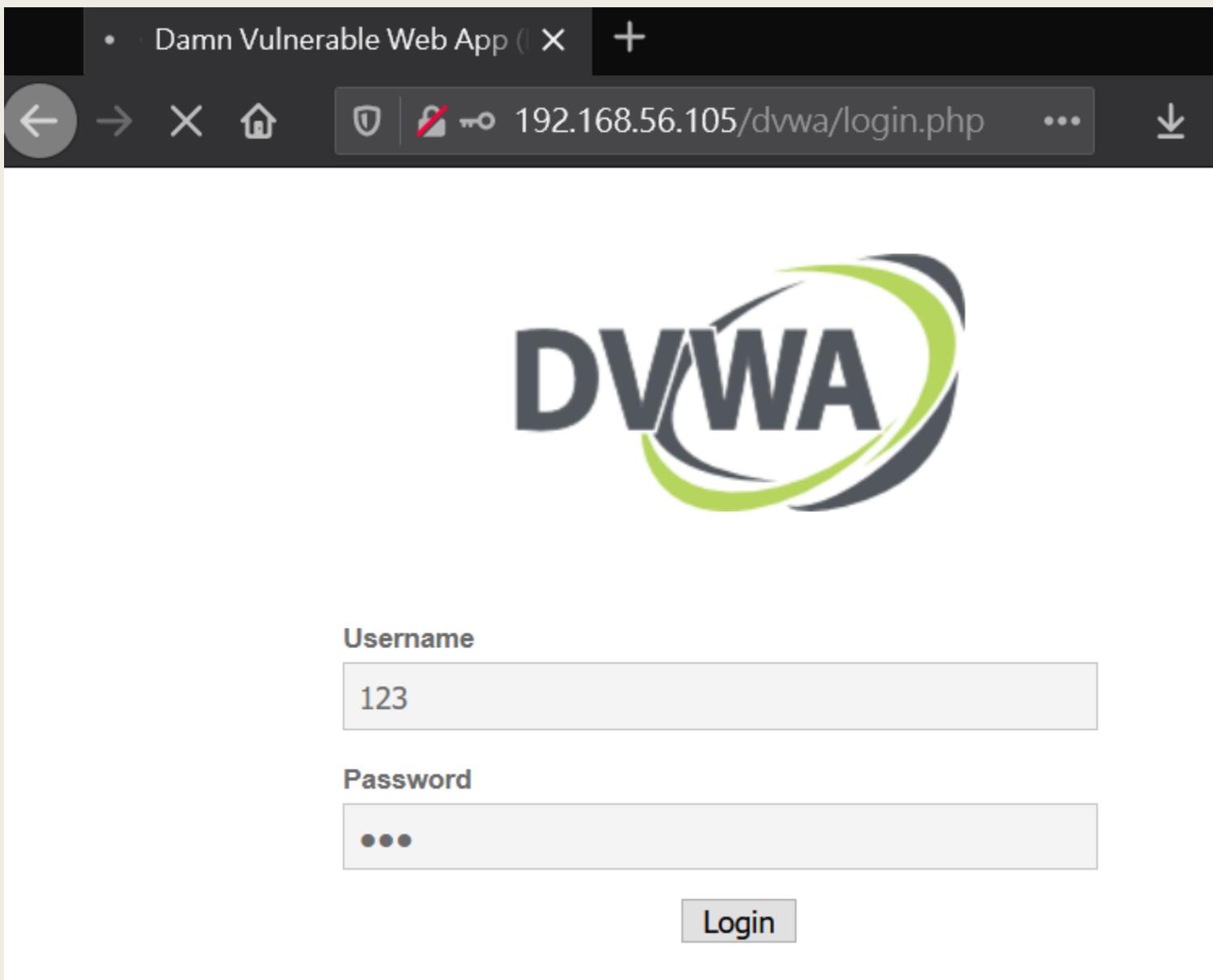
value="http://stock.weliketoshop.net:8080/p
roduct/stock/check?productId=2&storeId=3"



METASPLOITABLE 2

DVWA - Damn Vulnerable Web Application

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', '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'.

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

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

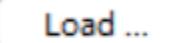
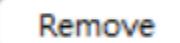
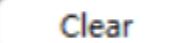
② Payload Sets

You can define one or more payload sets. The number of payload sets depends on the number of payload types available for each payload set, and each payload type can be customized in the following ways:

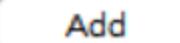
Payload set: **1**  Payload count: 5
Payload type: Simple list Request count: 0

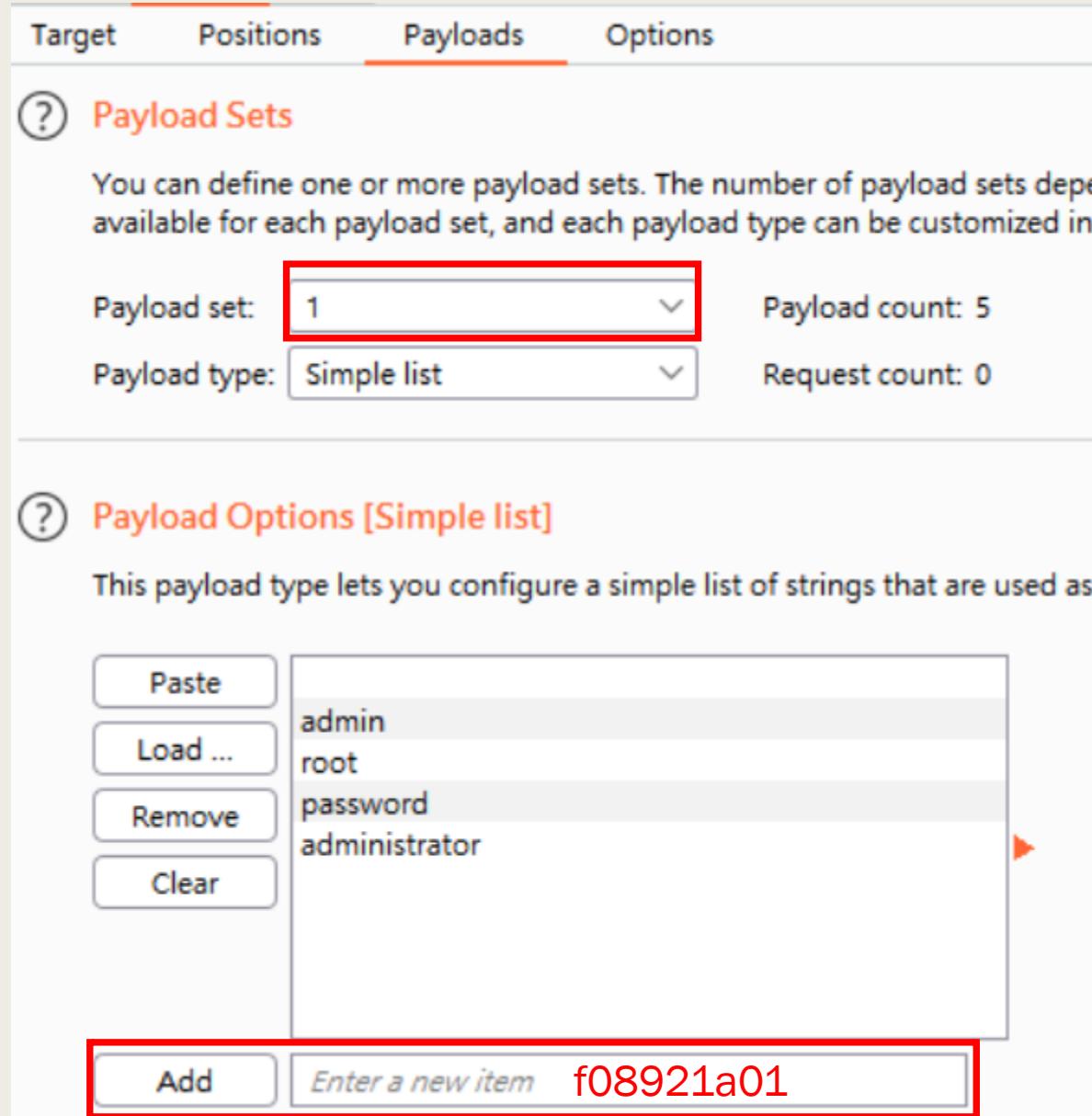
③ Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as input for the payload set.

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



Target Positions Payloads Options

Case sensitive match

Exclude HTTP headers

Grep - Extract

These settings can be used to extract useful information from responses.

Extract the following items from responses:

Add

Maximum capture length:

Grep - Payloads

These settings can be used to flag result items containing payload strings.

Search responses for payload strings
 Case sensitive match
 Exclude HTTP headers

Define extract grep item

Define the location of the item to be extracted. Selecting the item in the response panel will create a suitable configuration automatically. You can also modify the configuration manually to ensure it works effectively.

Define start and end

Start after expression: Extract from regex group

Start at offset:

End at delimiter: Case sensitive

End at fixed length:

Exclude HTTP headers Update config based on selection below

```
1 HTTP/1.1 302 Found
2 Date: Sun, 21 Mar 2021 17:21:34 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
13
```

Screenshot-01: Payload1 放上學號

Intruder attack 19

Attack Save Columns

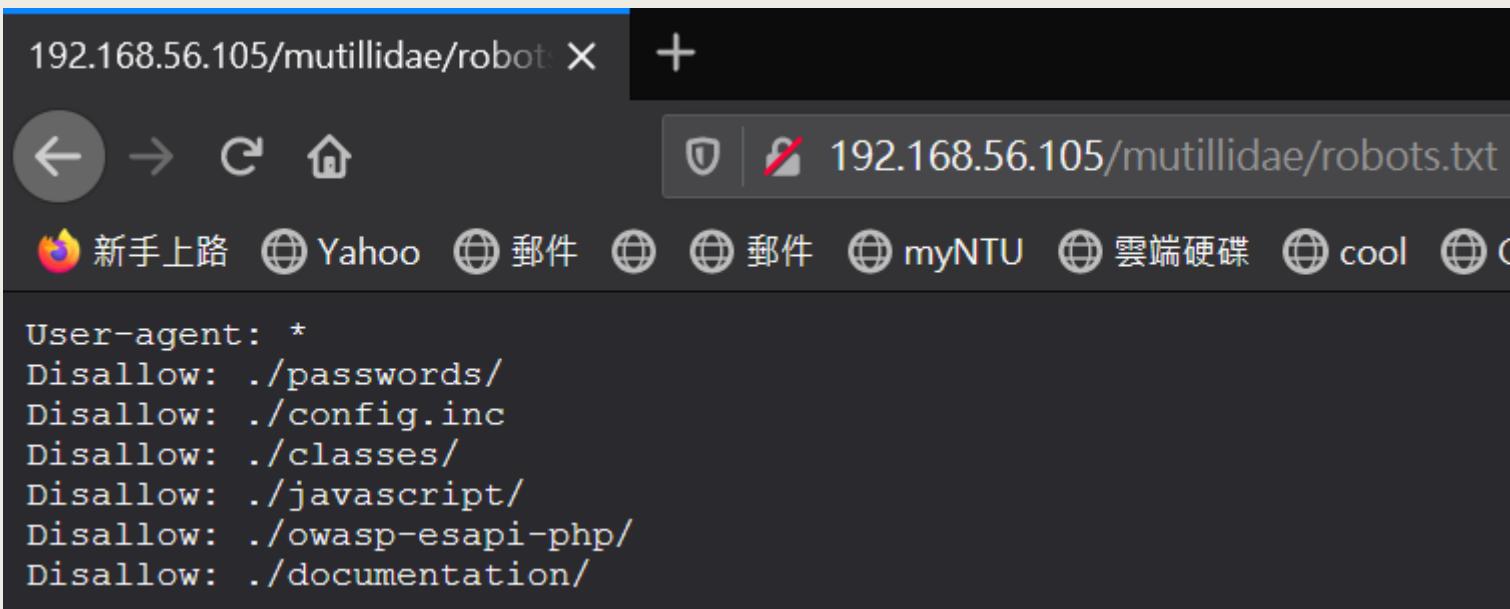
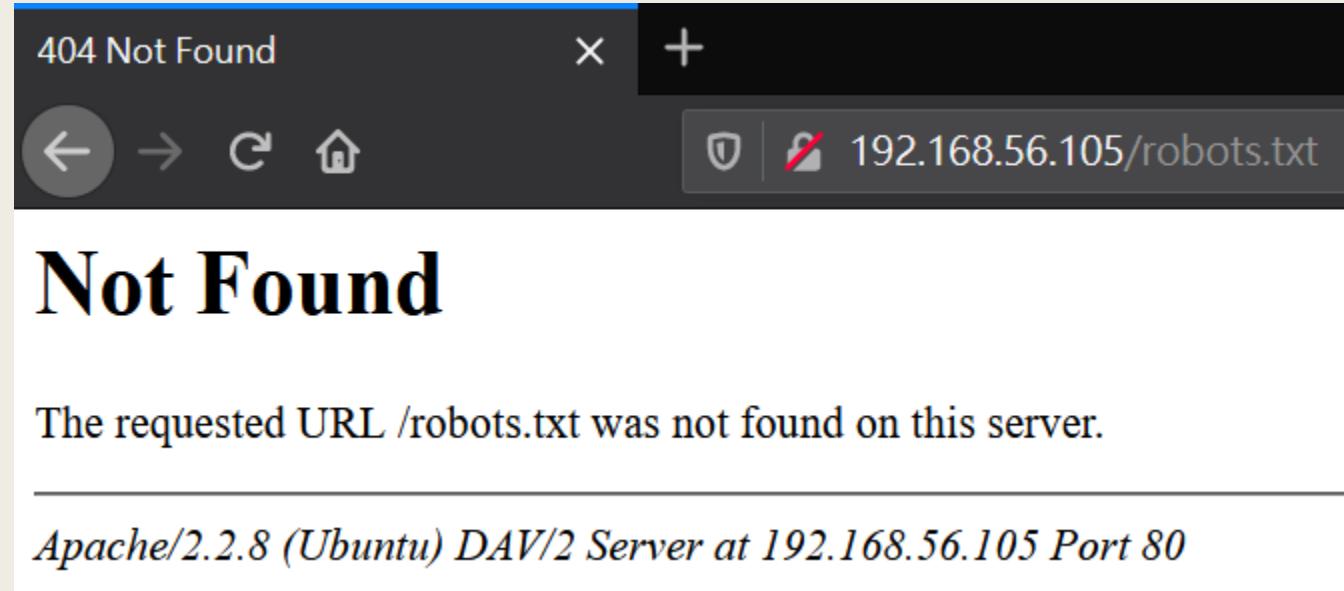
Results Target Positions Payloads Options

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

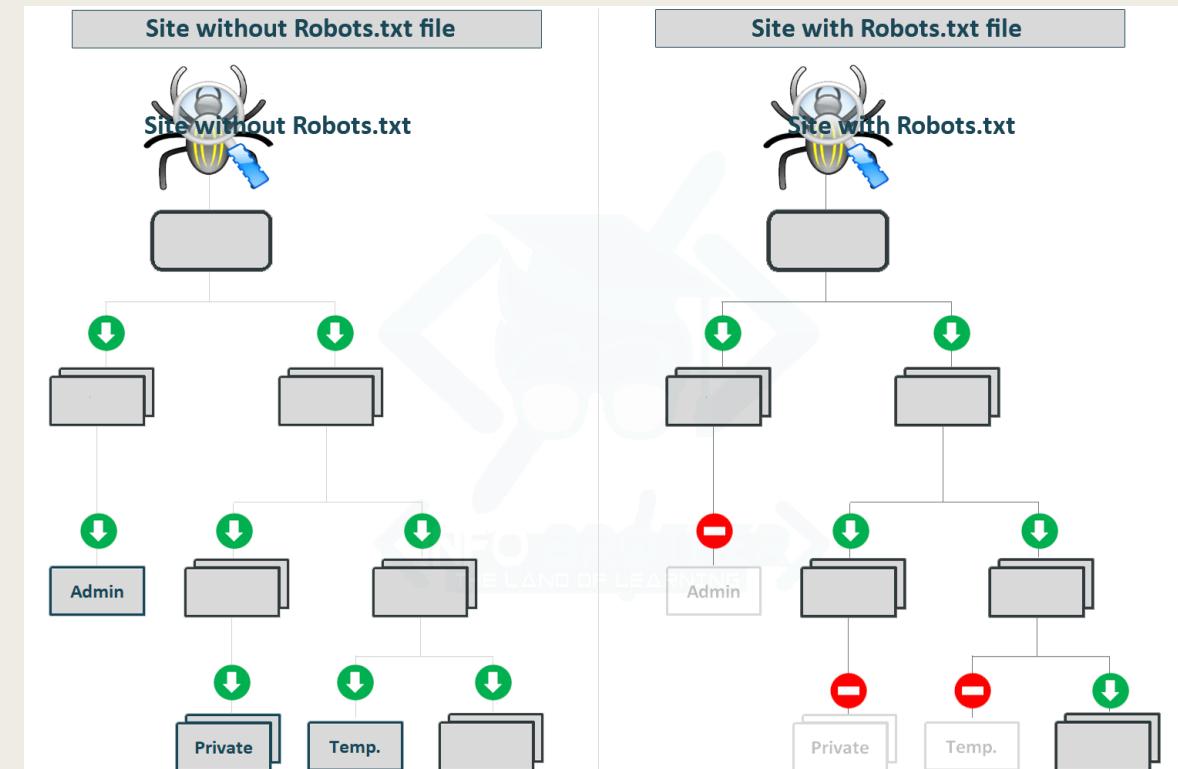


robots.txt

- 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

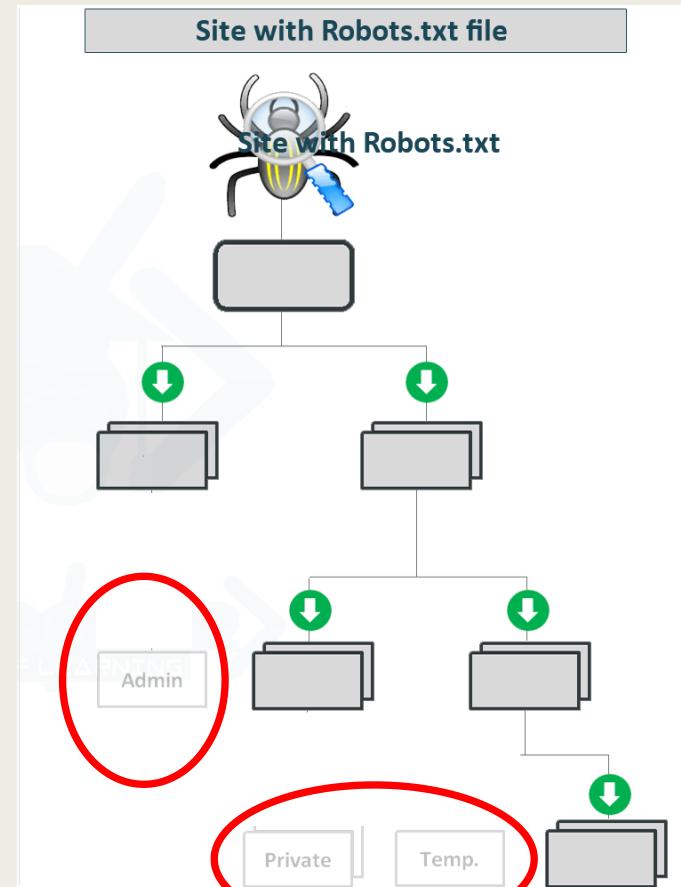
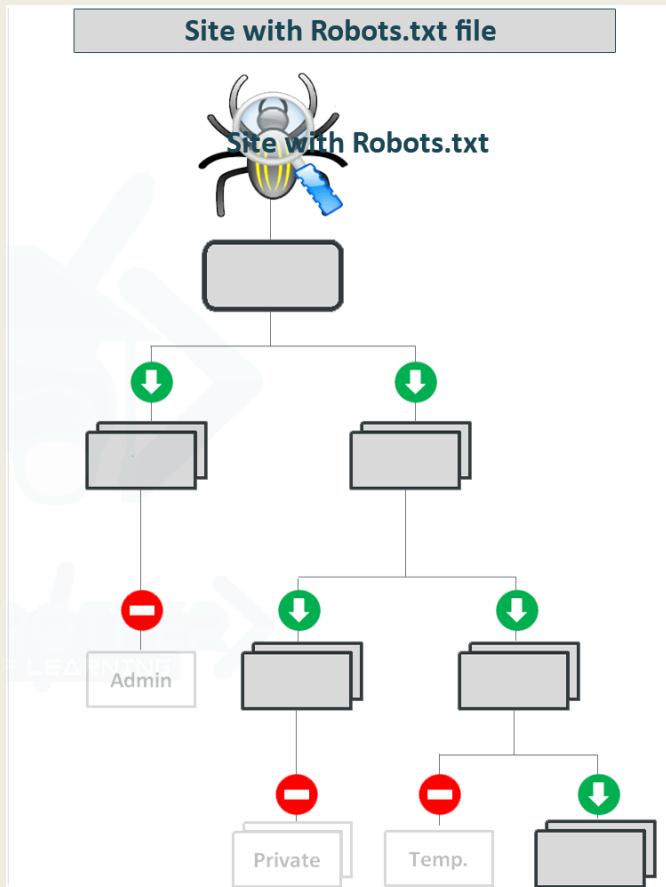
Google:

<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)

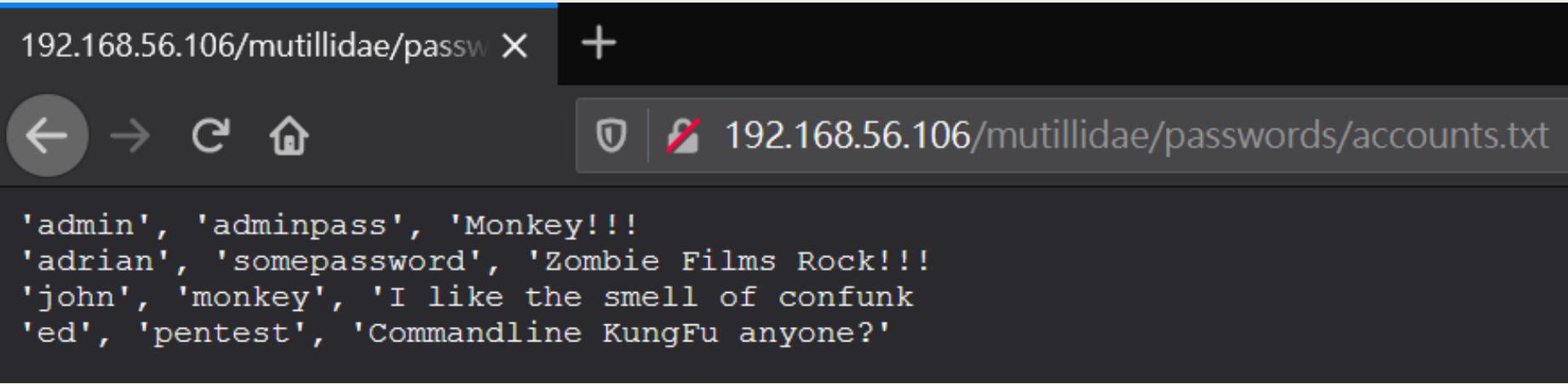
Current number of running threads: 10

Back Pause Stop Report

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

Screenshot-02: Change 欄位放上學號

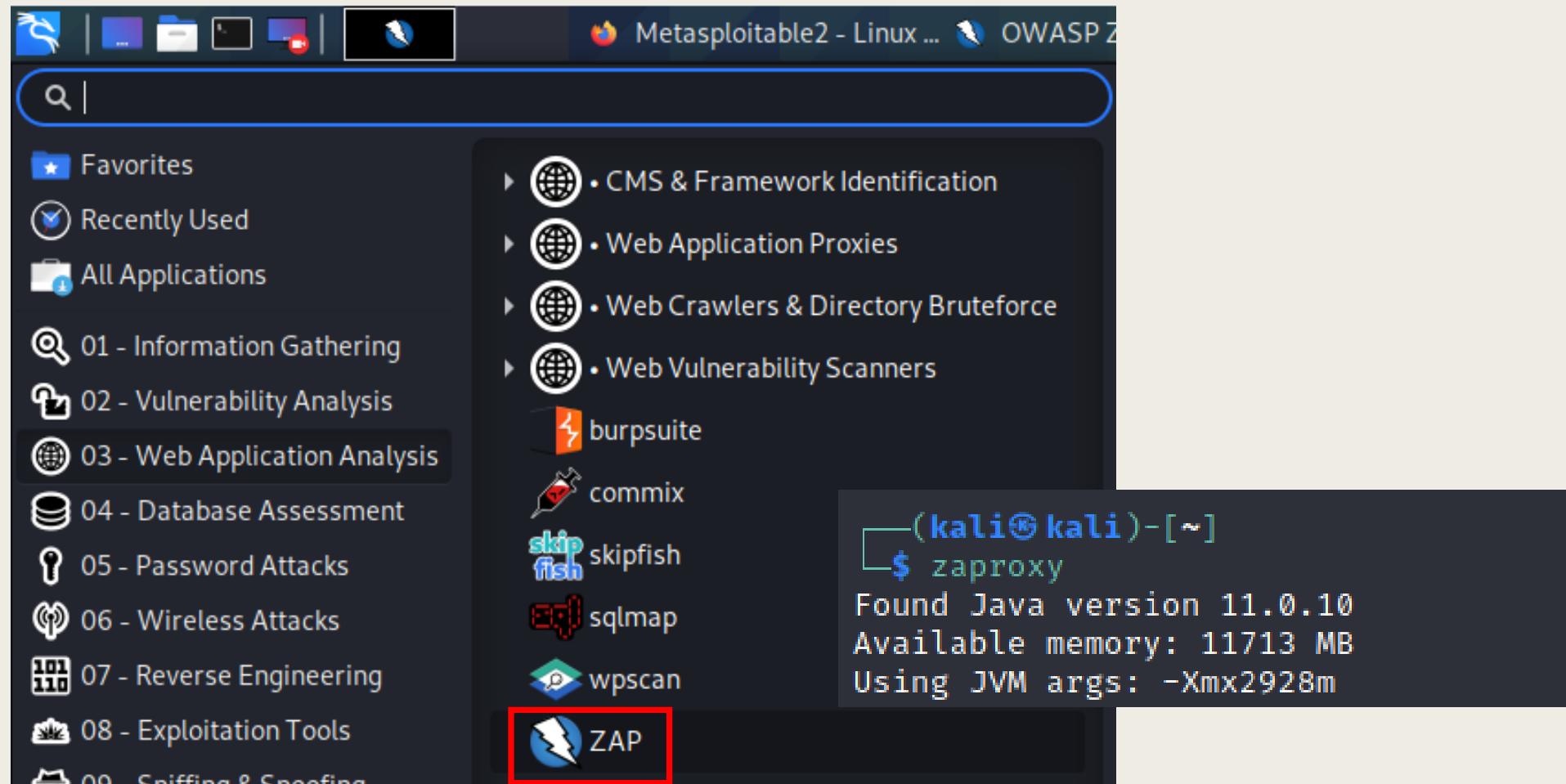
Hidden resource



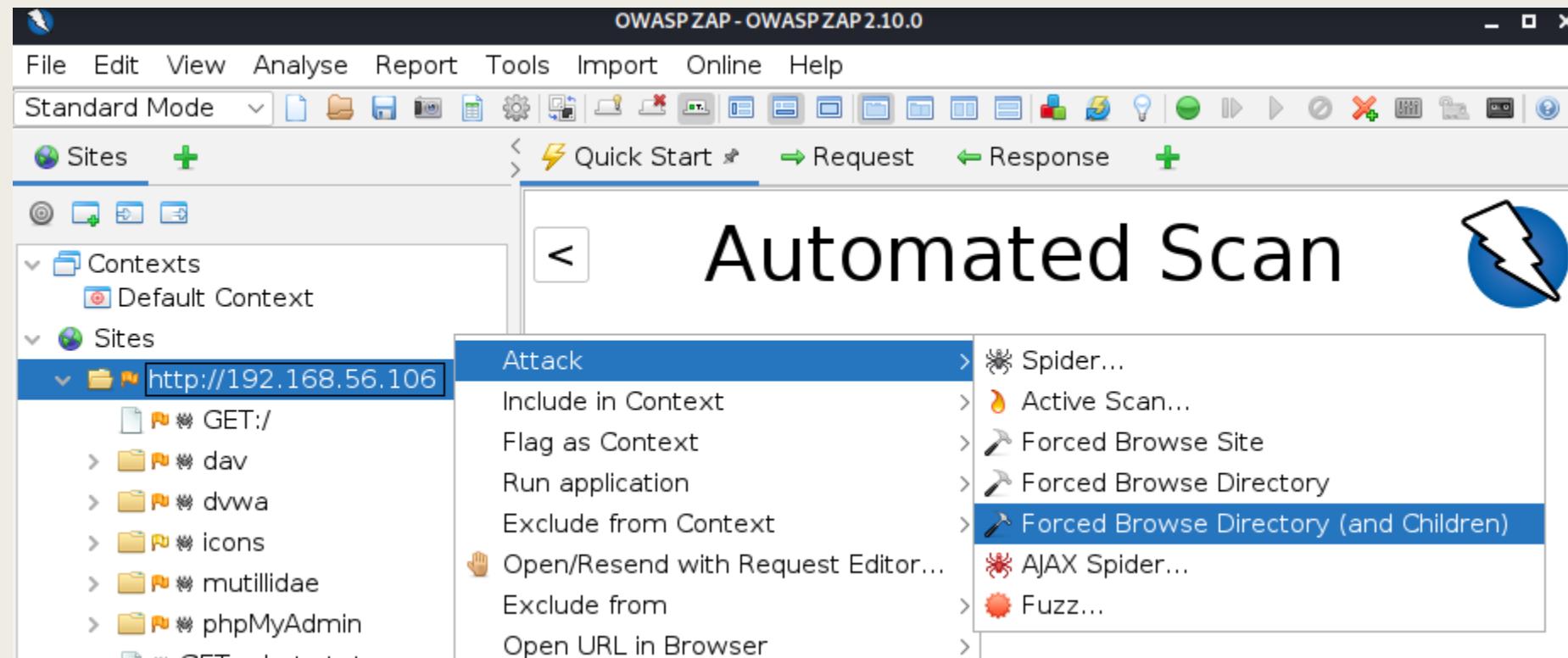
A screenshot of a web browser window. The address bar shows the URL `192.168.56.106/mutillidae/passw`. Below the address bar are standard navigation buttons: back, forward, refresh, and home. To the right of the address bar is a status bar showing the URL `192.168.56.106/mutillidae/passwords/accounts.txt`. 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?'
```

ZAP (Zed Attack Proxy)



ZAP (Zed Attack Proxy)



Information Disclosure

- Backup files
 - backup.zip
 - www.tar.gz
 - index.php.bak
- Temporary files
 - .index.php.swp
 - index.php~
 - #index.php#
- Log Files
 - *.log
- Version Control Systems
 - .git / .svn / .hg
 - Tools (可還原 Source Code)
 - github.com/denny0223/scrabble
 - github.com/lijiejie/GitHack
- Google Hacking Database
 - <https://www.exploit-db.com/google-hacking-database>

Information Disclosure

- intitle:"index of" "auth.log"
- Logging sensitive data
 - db password
 - user password
 - secret key
 - ...

Index of /var/log

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory			-
 alternatives.log	2020-09-12 00:28	117K	
 apt/	2020-12-25 06:03		-
 aptitude	2020-09-12 00:29	50K	
 auth.log	2021-03-24 08:52	864M	
 btmp	2021-03-24 08:52	6.6G	
 cfingerd.log	2021-03-21 06:47	0	
 cfingerd.log.0	2021-03-14 06:47	0	
 cfingerd.log.2.gz	2021-02-28 06:47	35	
 daemon.log	2021-03-24 08:40	130M	

Information Disclosure

- inurl: ./git

The screenshot shows a browser window with the title "Index of ./git/" and the URL "https://codemirror.net/.git/". The page lists several directory entries:

名稱	修改日期
.. /	19-Aug-2015 13:43
branches /	21-Oct-2016 06:17
hooks /	20-Mar-2021 11:12
info /	

Below the table, the browser's address bar shows the path: "nmlab > Web Security > GitHack > codemirror.net". A file explorer sidebar on the right lists the following folders:

- .github
- addon
- demo
- doc
- keymap
- lib

The screenshot shows the GitHub repository page for "codemirror / CodeMirror". The repository has 390 issues and 7 branches. The master branch is selected. The repository was created by "marijnh" on March 20, 2021, at 5.60.0. The repository structure includes .github, addon, and bin.

<https://github.com/codemirror/CodeMirror>

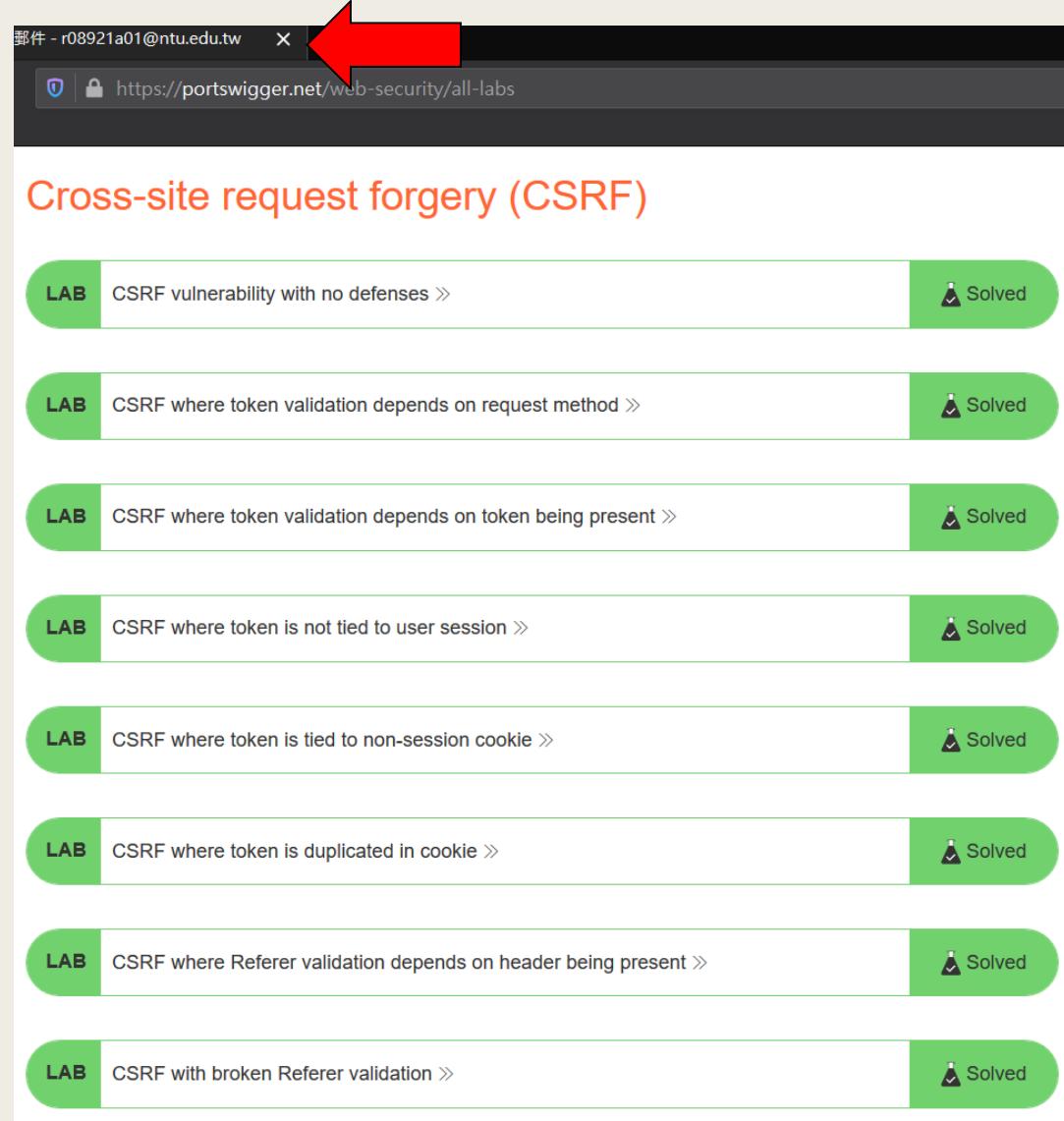
HW

- 10 Labs in XSS:
 - <https://portswigger.net/web-security/cross-site-scripting>
- 8 Labs in CSRF:
 - <https://portswigger.net/web-security/csrf>
- 5 Labs in SSRF (First 5):
 - <https://portswigger.net/web-security/ssrf>

HW

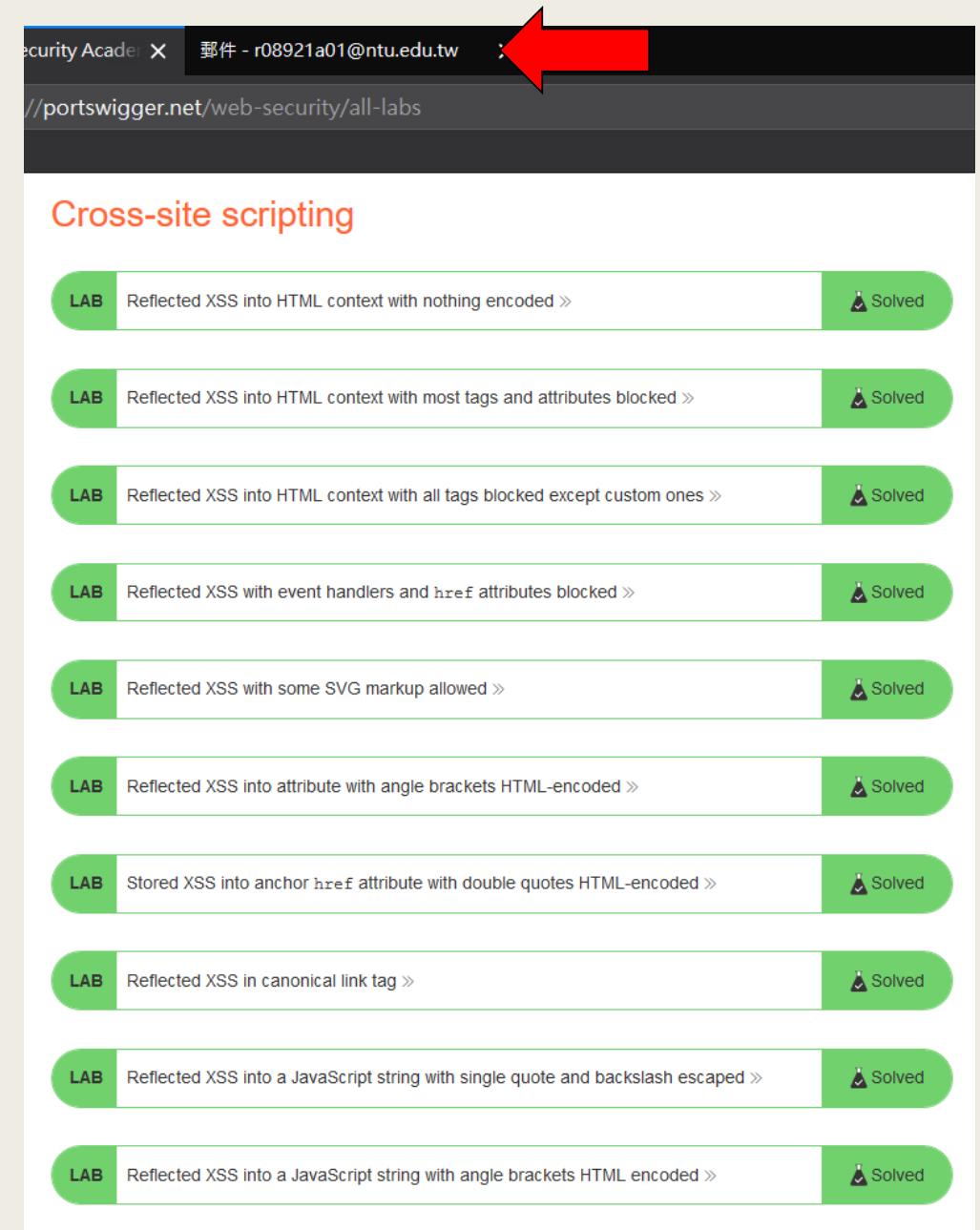
- 上傳 .pdf 檔，包含：
 - 1. Screenshot-01: Payload1 放上學號
 - 2. Screenshot-02: Change 欄位放上學號
 - 3~5. (XSS, CSRF, SSRF) @ <https://portswigger.net/web-security/all-labs>
 - 6. Lab reflection

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". A red arrow points to the browser tab. The main content is titled "Cross-site request forgery (CSRF)" and lists nine solved labs:

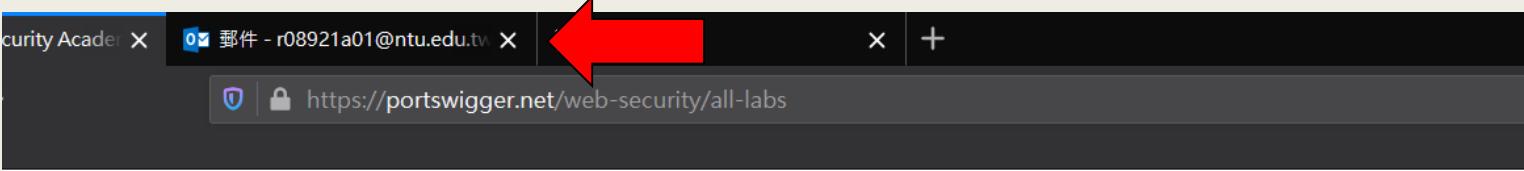
- LAB CSRF vulnerability with no defenses » Solved
- LAB CSRF where token validation depends on request method » Solved
- LAB CSRF where token validation depends on token being present » Solved
- LAB CSRF where token is not tied to user session » Solved
- LAB CSRF where token is tied to non-session cookie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



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". A red arrow points to the browser tab. The main content is titled "Cross-site scripting" and lists twelve solved labs:

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

With your Email logged in



A screenshot of a web browser window. The address bar shows the URL <https://portswigger.net/web-security/all-labs>. A large red arrow points from the top of the slide towards the browser window. The main content area displays a list of five SSRF labs, each with a green 'Solved' status indicator.

LAB	Description	Status
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