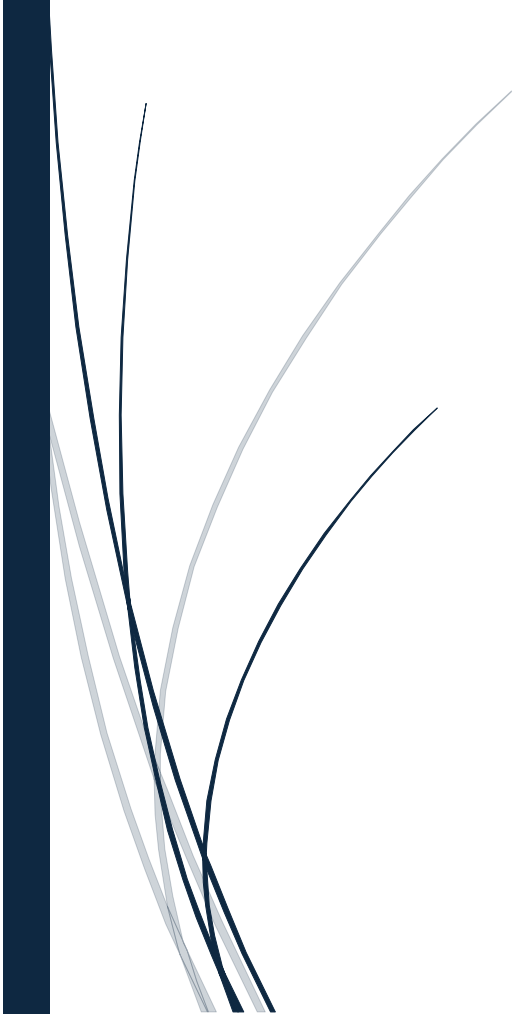
August 12,2025

Jawad Ali

VAPT Assignment



VAPT Assignment

Host Discovery Commands:

1. **ICMP Echo (Ping) Scan:**
   ICMP Echo Scan sends an echo request to see if the host is alive. We use it to discover live hosts.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn -PE 192.168.72.134
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 09:30 EDT
Nmap scan report for 192.168.72.134
Host is up (0.00055s latency).
MAC Address: 00:0C:29:BF:C8:86 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
```

   Here, we can see that the host is alive, and we also got the MAC address.

2. **TCP ACK Ping:**
   This type of scan sends TCP packet with only the ACK flag set to specific ports. We use it to check host availability when ICMP is blocked but certain TCP ports are allowed through a firewall.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn -PA80,443 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 09:37 EDT
Nmap scan report for 192.168.72.134
Host is up (0.00071s latency).
MAC Address: 00:0C:29:BF:C8:86 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.13 seconds
```

   Here, we can observe that the host is available.

3. **SCTP Init Ping:**
   It sends an SCTP INIT chunk to check if the host responds. An SCTP INIT chunk is the very first message sent when establishing an SCTP (Stream Control Transmission Protocol) connection. We use it for networks/services using SCTP e.g. telecom protocols. This is also useful in specialized environments.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn -PY132 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 09:51 EDT
Nmap scan report for 192.168.72.134
Host is up (0.00096s latency).
MAC Address: 00:0C:29:BF:C8:86 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.13 seconds
```

4. **ICMP Timestamp Ping:**
   It sends an ICMP Timestamp Request to get the target's system time. We use it when normal ping is blocked, but timestamp replies may still be enabled. It can also help detect OS clock differences.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn -PP 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 10:02 EDT
Nmap scan report for 192.168.72.134
Host is up (0.00061s latency).
MAC Address: 00:0C:29:BF:C8:86 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
```

5. **ICMP Address Mask Ping:**
   It requests the subnet mask from the target. We use it for network mapping in poorly secured networks.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn -PM 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 10:16 EDT
Nmap scan report for 192.168.72.134
Host is up (0.00061s latency).
MAC Address: 00:0C:29:BF:C8:86 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
```

6. **ARP Ping (For LAN-based networks):**
   It is used to send ARP requests to find active hosts on a local network. We use it for network discovery within the same subnet.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn -PR 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 10:19 EDT
Nmap scan report for 192.168.72.134
Host is up (0.00068s latency).
MAC Address: 00:0C:29:BF:C8:86 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
```

7. **Find MAC Address of Victim:**
   It is basically the physical hardware address of the target. We use it for device identification.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sn 192.168.72.134
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 10:23 EDT
Nmap scan report for 192.168.72.134
Host is up (0.00057s latency).
MAC Address: 00:0C:29:BF:C8:86 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
```

Here, we got the MAC address of our target which is a virtual machine in this case.

OS Discovery Commands:

1. **OS Detection (via TTL analysis or appropriate method):**

Here we detect the type of OS the target is using by their TTL value.

```
┌──(kali㉿kali)-[~]
└─$ nmap -Pn -p80 --reason 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 10:30 EDT
Nmap scan report for 192.168.72.134
Host is up, received arp-response (0.00074s latency).

PORT   STATE SERVICE REASON
80/tcp open  http    syn-ack ttl 64
MAC Address: 00:0C:29:BF:C8:86 (VMware)
```

Here, we got TTL value that is 64, so we came to know that the target has a Linux, or a Unix based system.

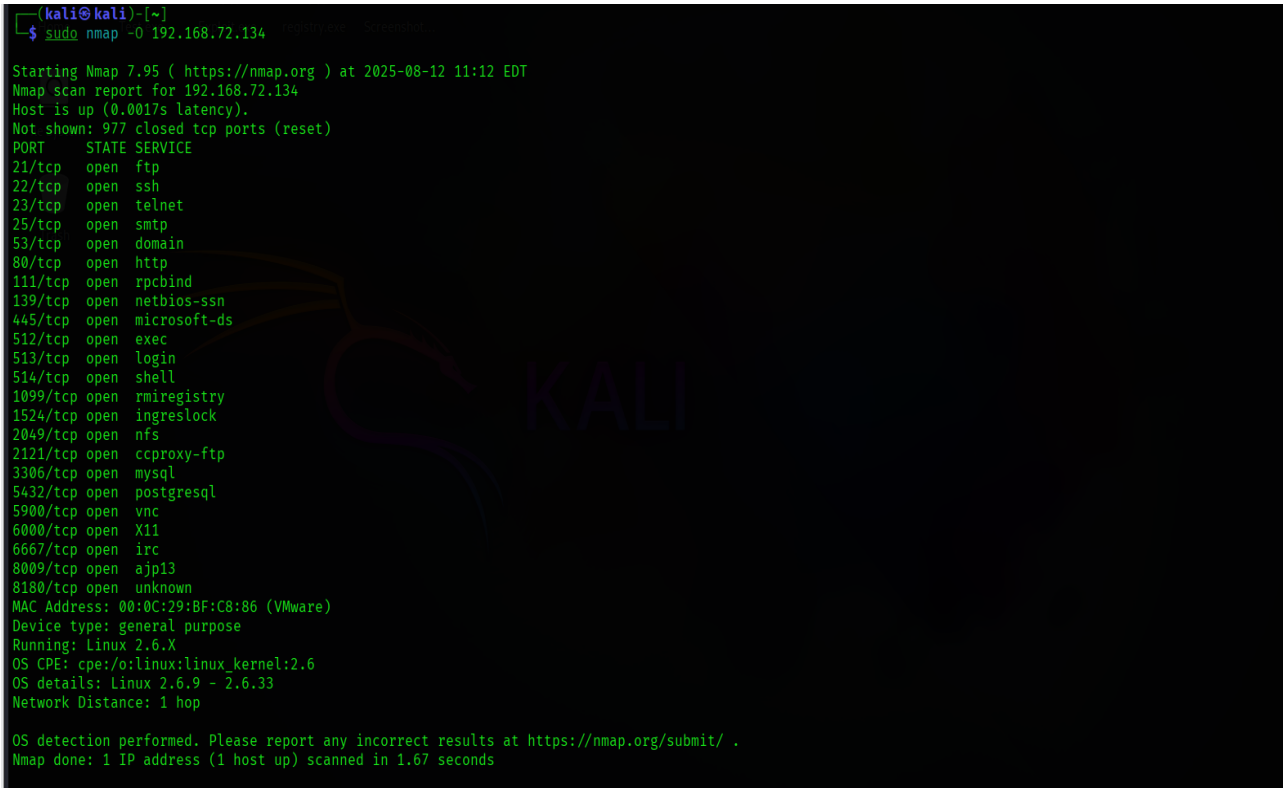
2. **OS Fingerprinting using Unicornscan:**

It sends specially crafted TCP/UDP packets to gather OS-specific responses. Unicornscan is a bit faster than nmap.

```
┌──(kali㉿kali)-[~]
└─$ sudo unicornscan -Iv 192.168.72.134

adding 192.168.72.134/32 mode `TCPscan' ports `7,9,11,13,18,19,21-23,25,37,39,42,49,50,53,65,67-70,79-81,88,98,100,105-107,
109-111,113,118,119,123,129,135,137-139,143,150,161-164,174,177-179,191,199-202,204,206,209,210,213,220,345,346,347,369-372
,389,406,407,422,443-445,487,500,512-514,517,518,520,525,533,538,548,554,563,587,610-612,631-634,636,642,653,655,657,666,70
6,750-752,765,779,808,873,901,923,941,946,992-995,1001,1023-1030,1080,1210,1214,1234,1241,1334,1349,1352,1423-1425,1433,143
4,1524,1525,1645,1646,1649,1701,1718,1719,1720,1723,1755,1812,1813,2048-2050,2101-2104,2140,2150,2233,2323,2345,2401,2430,2
431,2432,2433,2583,2628,2776,2777,2988,2989,3050,3130,3150,3232,3306,3389,3456,3493,3542-3545,3632,3690,3801,4000,4400,4321
,4567,4899,5002,5136-5139,5150,5151,5222,5269,5308,5354,5355,5422-5425,5432,5503,5555,5556,5678,6000-6007,6346,6347,6543,65
44,6789,6838,6666-6670,7000-7009,7028,7100,7983,8079-8082,8088,8787,8879,9090,9101-9103,9325,9359,10000,10026,10027,10067,1
0080,10081,10167,10498,11201,15345,17001-17003,18753,20011,20012,21554,22273,26274,27374,27444,27573,31335-31338,31787,3178
9,31790,31791,32668,32767-32780,33390,47262,49301,54320,54321,57341,58008,58009,58666,59211,60000,60006,61000,61348,61466,6
1603,63485,63808,63809,64429,65000,65506,65530-65535' pps 300
using interface(s) eth0
scaning 1.00e+00 total hosts with 3.38e+02 total packets, should take a little longer than 8 Seconds
sender statistics 296.8 pps with 338 packets sent total
listener statistics 338 packets recieved 0 packets droped and 0 interface drops
```

There is an error in the output, I tried to resolve it but could not do it.

3. **OS Discovery using Nmap Scripting Engine:**
   Here we are using nmap's built in scripts to detect the operating system. It can combine service fingerprinting with OS guesses.



Here, we can see various ports that are open along with the OS that is Linux.

Port and Service Discovery:

1. **TCP Connect Scan:**
   It is a full TCP 3-way handshake with each scanned port. It works even if SYN scan isn't allowed. We use it when we don't have raw packet privileges or need guaranteed accurate results.

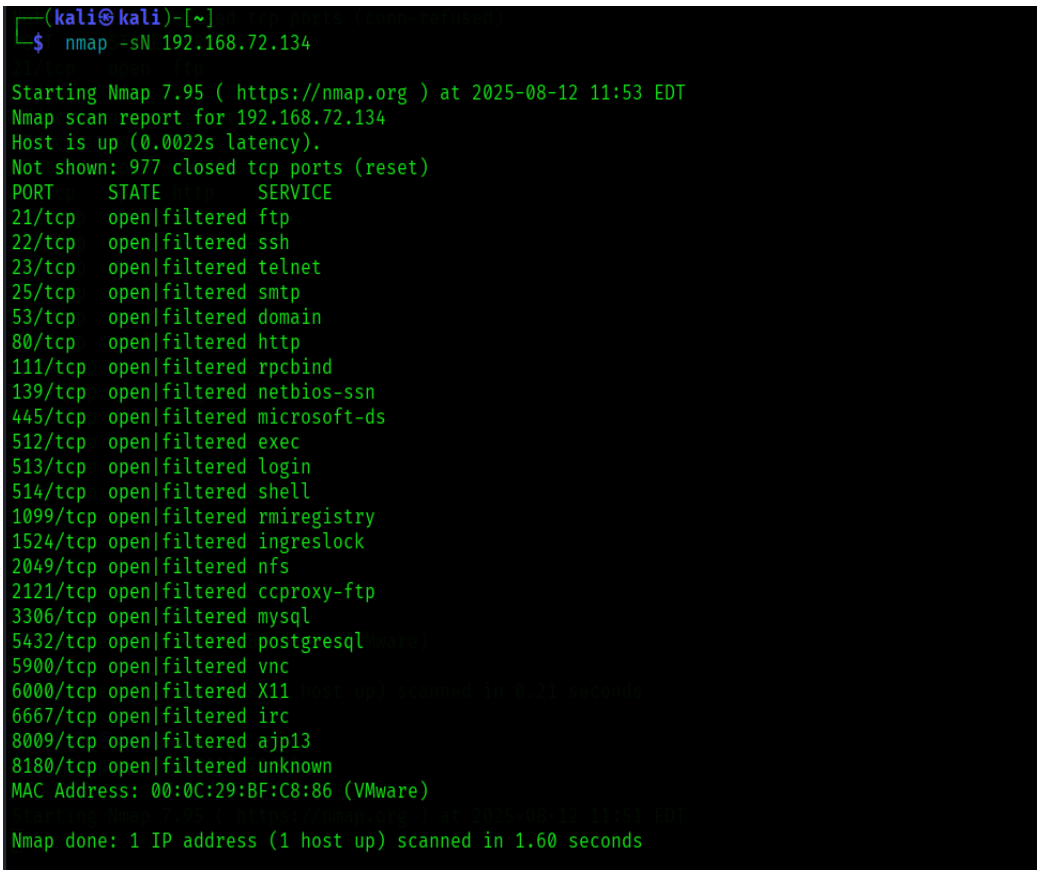
```
┌──(kali㉿kali)-[~]
└─$ nmap -sT 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 11:49 EDT
Nmap scan report for 192.168.72.134
Host is up (0.00094s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 00:0C:29:BF:C8:86 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.21 seconds
```

2. **UDP Scan:**
   It sends UDP packets to detect open UDP ports. We use it when we want to find UDP based services.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sU 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 12:05 EDT
Nmap scan report for 192.168.72.134
Host is up (0.0011s latency).
Not shown: 990 open|filtered udp ports (no-response)
PORT      STATE  SERVICE
53/udp    open   domain
111/udp   open   rpcbind
137/udp   open   netbios-ns
2049/udp  open   nfs
18835/udp closed unknown
36778/udp closed unknown
47624/udp closed directplaysrvr
49165/udp closed unknown
49178/udp closed unknown
58797/udp closed unknown
MAC Address: 00:0C:29:BF:C8:86 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 8.84 seconds
```

Here, the output tells use about the UDP based services that are open.

3. **TCP Null Scan:**
   It sends packets with no TCP flags set. We use it for stealth scanning to bypass basic IDS or firewalls.



   Here, we can observe the open port services along with their states.

4. **TCP FIN Scan:**
   It sends FIN flag packets to close non-existent connections. We use it when there is a system that responds differently to FIN probes.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sF 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 11:56 EDT
Nmap scan report for 192.168.72.134
Host is up (0.0010s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE         SERVICE
21/tcp   open|filtered ftp
22/tcp   open|filtered ssh
23/tcp   open|filtered telnet
25/tcp   open|filtered smtp
53/tcp   open|filtered domain
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
139/tcp  open|filtered netbios-ssn
445/tcp  open|filtered microsoft-ds
512/tcp  open|filtered exec
513/tcp  open|filtered login
514/tcp  open|filtered shell
1099/tcp open|filtered rmiregistry
1524/tcp open|filtered ingreslock
2049/tcp open|filtered nfs
2121/tcp open|filtered ccproxy-ftp
3306/tcp open|filtered mysql
5432/tcp open|filtered postgresql
5900/tcp open|filtered vnc
6000/tcp open|filtered X11
6667/tcp open|filtered irc
8009/tcp open|filtered ajp13
8180/tcp open|filtered unknown
MAC Address: 00:0C:29:BF:C8:86 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.55 seconds
```

5. **Xmas Scan:**
   It is used for stealth detection. It sorts of lights up like a Christmas tree in packet flags. We used it when we attempt to evade detection or exploit older TCP/IP stack quirks.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sX 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 11:58 EDT
Nmap scan report for 192.168.72.134
Host is up (0.0033s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE         SERVICE
21/tcp   open|filtered ftp
22/tcp   open|filtered ssh
23/tcp   open|filtered telnet
25/tcp   open|filtered smtp
53/tcp   open|filtered domain
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
139/tcp  open|filtered netbios-ssn
445/tcp  open|filtered microsoft-ds
512/tcp  open|filtered exec
513/tcp  open|filtered login
514/tcp  open|filtered shell
1099/tcp open|filtered rmiregistry
1524/tcp open|filtered ingreslock
2049/tcp open|filtered nfs
2121/tcp open|filtered ccproxy-ftp
3306/tcp open|filtered mysql
5432/tcp open|filtered postgresql
5900/tcp open|filtered vnc
6000/tcp open|filtered X11
6667/tcp open|filtered irc
8009/tcp open|filtered ajp13
8180/tcp open|filtered unknown
MAC Address: 00:0C:29:BF:C8:86 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.57 seconds
```

6. **TCP ACK Scan:**
   It sends ACK packets to check firewall rules. We use it if a firewall is blocking packets without scanning for open ports. We use it for firewall rule mapping.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sA 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 12:00 EDT
Nmap scan report for 192.168.72.134
Host is up (0.0037s latency).
All 1000 scanned ports on 192.168.72.134 are in ignored states.
Not shown: 1000 unfiltered tcp ports (reset)
MAC Address: 00:0C:29:BF:C8:86 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.36 seconds
```

   This tells us that all the ports are in ignored state.

7. **TCP Window Scan:**
   It is similar to ACK scan but examines TCP Window size to guess port states. It can sometimes reveal ports when ACK scan cannot. We use it when we suspect that firewall filtering is in place.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sW 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 12:01 EDT
Nmap scan report for 192.168.72.134
Host is up (0.0013s latency).
All 1000 scanned ports on 192.168.72.134 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 00:0C:29:BF:C8:86 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.35 seconds
```

8. **TCP Maimon Scan:**
   It sends FIN/ACK probes and observes if the system responds with RST. We use it because it can bypass certain packet filters and IDS. We use it when we are trying advanced evasion against older systems.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sM 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 12:03 EDT
Nmap scan report for 192.168.72.134
Host is up (0.0018s latency).
All 1000 scanned ports on 192.168.72.134 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: 00:0C:29:BF:C8:86 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.36 seconds
```

9. **IP Protocol Scan:**
   It detects IP protocols supported by the host. We use it to identify non-TCP/UDP protocols that are being used. It is used in deep reconnaissance to find unusual services.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sO 192.168.72.134

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 12:04 EDT
Nmap scan report for 192.168.72.134
Host is up (0.00083s latency).
Not shown: 253 open|filtered n/a protocols (no-response)
PROTOCOL STATE  SERVICE
1        open   icmp
6        open   tcp
212      closed unknown
MAC Address: 00:0C:29:BF:C8:86 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 2.44 seconds
```

Here, it shows that icmp and tcp services are open.

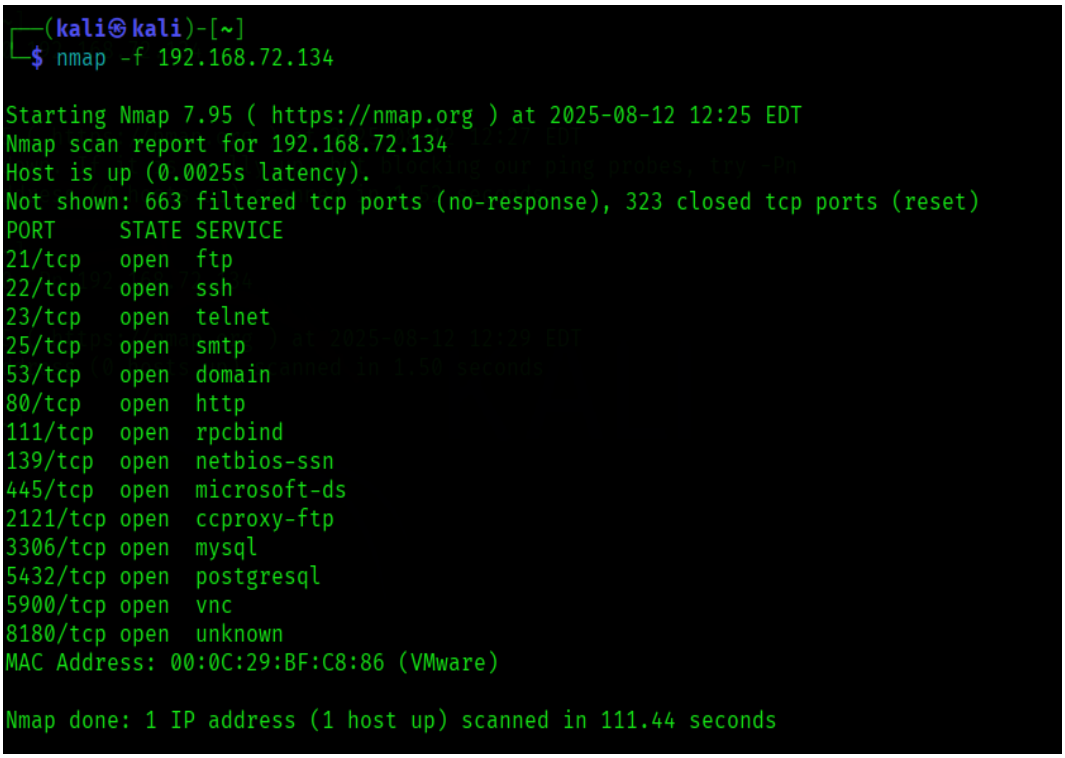
Service Enumeration:

1. **SNMP Enumeration using Nmap:**
   It is used to extract detailed information from SNMP service such as device name and description, operating system and version, network interface, etc.

```
┌──(kali㉿kali)-[~]
└─$ nmap -sU -p 161 192.168.1.10

Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-12 12:13 EDT
Nmap scan report for 192.168.1.10
Host is up (0.00076s latency).

PORT    STATE         SERVICE
161/udp open|filtered snmp

Nmap done: 1 IP address (1 host up) scanned in 0.50 seconds
```

Firewall/IDS Evasion Scans:

1. **Fragmented Packet Scan:**
   It is used to split the TCP packets in small fragments so that the IDS or firewall cannot detect them.



2. **Decoy Scan:**
   It sends packets that appear to come from multiple IP addresses. It makes it harder for the defenders to know our real IP.

3. **Source Port Spoofing:**
   It forges the source port of our packets, so they appear to come from a trusted service port. We use it to bypass firewall rules that whitelist certain ports.