PRACTICAL-7

AIM:

Perform port scanning using nmap on a single port and capture the packets using wireshark and analyze the output

THEORY:

1. **Nmap:**

- Nmap is a free and open-source network scanner created by Gordon Lyon. Nmap is used to discover hosts and services on a computer network by sending packets and analyzing the responses.
- Nmap provides a number of features for probing computer networks, including host discovery and service and operating system detection.
- These features are extensible by scripts that provide more advanced service detection, vulnerability detection, and other features.
- Nmap can adapt to network conditions including latency and congestion during a scan.
- Nmap started as a Linux utility and was ported to other systems including Windows, macOS, and BSD. It is most popular on Linux, followed by Windows.

2. **Wireshark:**

- Wireshark is a free and open-source packet analyser.
- It is used for network troubleshooting, analysis, software and communications protocol development, and education.
- Originally named Ethereal, the project was renamed Wireshark in May 2006 due to trademark issues.
- Wireshark is cross-platform, using the Qt widget toolkit in current releases to implement its user interface, and using pcap to capture packets; it runs on Linux, macOS, BSD, Solaris, some other Unix-like operating systems, and Microsoft Windows.

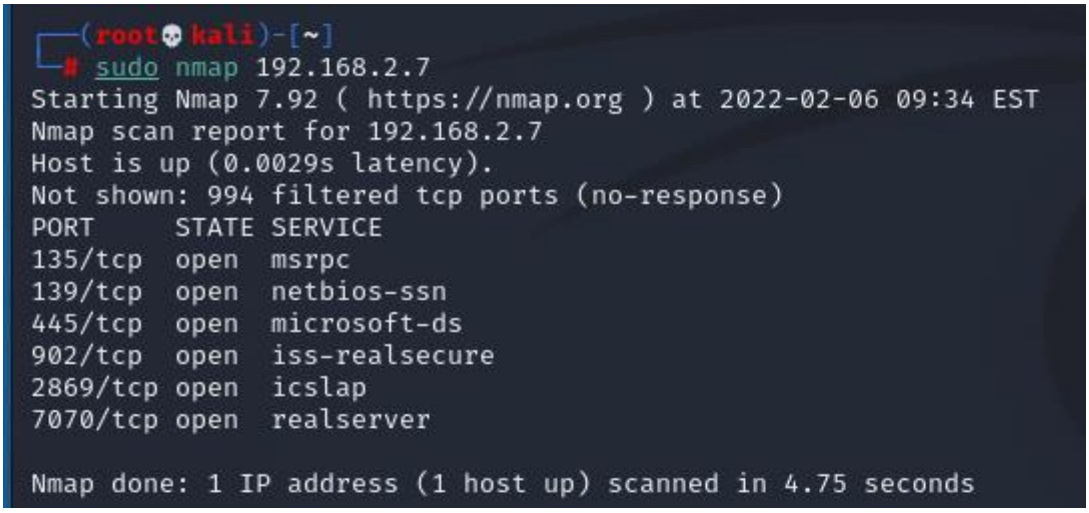
- There is also a terminal-based (non-GUI) version called TShark. Wireshark, and the other programs distributed with it such as TShark, are free software, released under the terms of version 2 of the GNU General Public License.

3. **open**
    - An application is actively accepting TCP connections, UDP datagrams or SCTP associations on this port.
    - Finding these is often the primary goal of port scanning. Security-minded people know that each open port is an avenue for attack.
    - Attackers and pen-testers want to exploit the open ports, while administrators try to close or protect them with firewalls without thwarting legitimate users.
    - Open ports are also interesting for non-security scans because they show services available for use on the network.
4. **closed**
    - A closed port is accessible (it receives and responds to Nmap probe packets), but there is no application listening on it.
    - They can be helpful in showing that a host is up on an IP address (host discovery, or ping scanning), and as part of OS detection. Because closed ports are reachable, it may be worth scanning later in case some open up.
    - Administrators may want to consider blocking such ports with a firewall. Then they would appear in the filtered state, discussed next.
5. **filtered**
    - Nmap cannot determine whether the port is open because packet filtering prevents its probes from reaching the port.
    - The filtering could be from a dedicated firewall device, router rules, or host-based firewall software.
    - These ports frustrate attackers because they provide so little information. Sometimes they respond with ICMP error messages such as type 3 code 13 (destination unreachable: communication administratively prohibited), but filters that simply drop probes without responding are far more common.
    - This forces Nmap to retry several times just in case the probe was dropped due to network congestion rather than filtering. This slows down the scan dramatically.

6. **Unfiltered**
    - The unfiltered state means that a port is accessible, but Nmap is unable to determine whether it is open or closed.
    - Only the ACK scan, which is used to map firewall rulesets, classifies ports into this state.
    - Scanning unfiltered ports with other scan types such as Window scan, SYN scan, or FIN scan, may help resolve whether the port is open.
7. **open|filtered**
    - Nmap places ports in this state when it is unable to determine whether a port is open or filtered. This occurs for scan types in which open ports give no response.
    - The lack of response could also mean that a packet filter dropped the probe or any response it elicited.
    - So Nmap does not know for sure whether the port is open or being filtered. The UDP, IP protocol, FIN, NULL, and Xmas scans classify ports this way.
8. **closed|filtered**
    - This state is used when Nmap is unable to determine whether a port is closed or filtered.
    - It is only used for the IP ID idle scan.

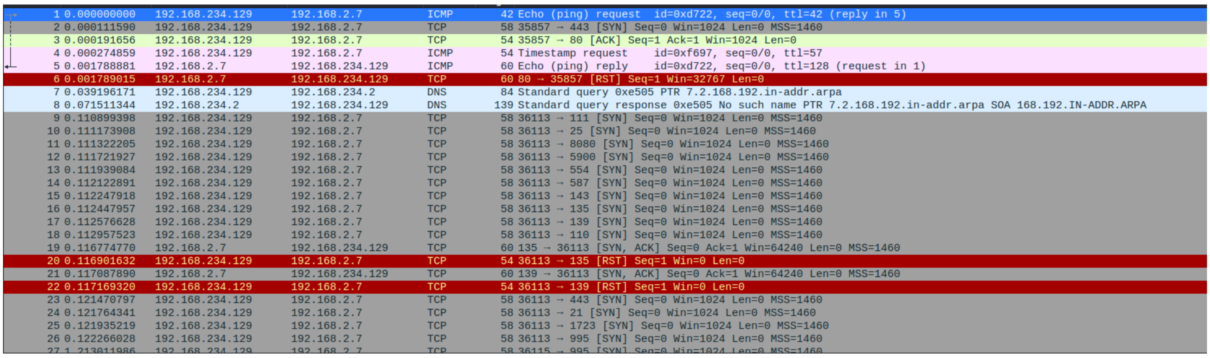
IMPLEMENTATION:

- Start the wireshark and start capturing the packets
- Firstly, we will write the following command:
- Write sudo nmap ip address of device
- This is the basic format for **Nmap**, and it will return information about the ports on that system.



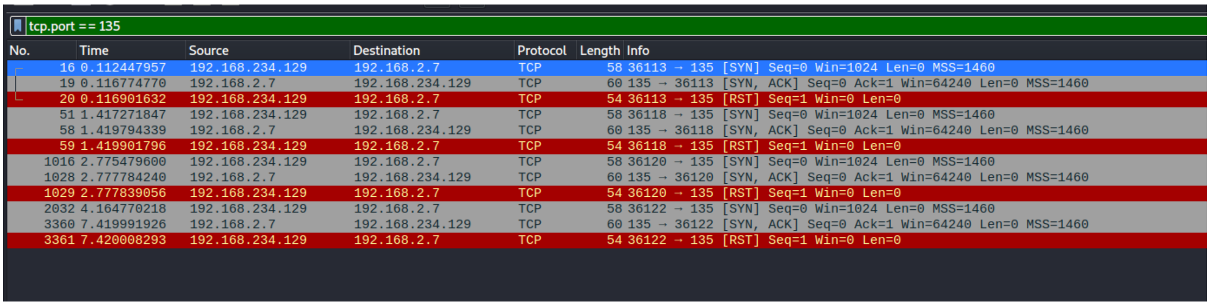
```
┌──(root💀kali)-[~]
└─# sudo nmap 192.168.2.7
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-06 09:34 EST
Nmap scan report for 192.168.2.7
Host is up (0.0029s latency).
Not shown: 994 filtered tcp ports (no-response)
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
902/tcp  open  iss-realsecure
2869/tcp open  icslap
7070/tcp open  realserver

Nmap done: 1 IP address (1 host up) scanned in 4.75 seconds
```

- Below are the glimpses of the packets captured by wireshark when the above command was executed



```
 1 0.000000000  192.168.234.129  192.168.2.7      ICMP   42 Echo (ping) request  id=0xd722, seq=0/0, ttl=42 (reply in 5)
 2 0.000111590  192.168.234.129  192.168.2.7      TCP    58 35857 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
 3 0.000191656  192.168.234.129  192.168.2.7      TCP    54 35857 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
 4 0.000274859  192.168.234.129  192.168.2.7      ICMP   54 Timestamp request    id=0xf697, seq=0/0, ttl=57
 5 0.001788881  192.168.2.7      192.168.234.129  ICMP   60 Echo (ping) reply    id=0xd722, seq=0/0, ttl=128 (request in 1)
 6 0.001789015  192.168.2.7      192.168.234.129  TCP    60 80 → 35857 [RST] Seq=1 Win=32767 Len=0
 7 0.039196171  192.168.234.129  192.168.234.2    DNS    84 Standard query 0xe505 PTR 7.2.168.192.in-addr.arpa
 8 0.071511344  192.168.234.2    192.168.234.129  DNS   139 Standard query response 0xe505 No such name PTR 7.2.168.192.in-addr.arpa SOA 168.192.IN-ADDR.ARPA
 9 0.110899398  192.168.234.129  192.168.2.7      TCP    58 36113 → 111 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
10 0.111173908  192.168.234.129  192.168.2.7      TCP    58 36113 → 25 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11 0.111322205  192.168.234.129  192.168.2.7      TCP    58 36113 → 8080 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
12 0.111721927  192.168.234.129  192.168.2.7      TCP    58 36113 → 5900 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
13 0.111939084  192.168.234.129  192.168.2.7      TCP    58 36113 → 554 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
14 0.112122891  192.168.234.129  192.168.2.7      TCP    58 36113 → 587 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
15 0.112247918  192.168.234.129  192.168.2.7      TCP    58 36113 → 143 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
16 0.112447957  192.168.234.129  192.168.2.7      TCP    58 36113 → 135 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
17 0.112576628  192.168.234.129  192.168.2.7      TCP    58 36113 → 139 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
18 0.112957523  192.168.234.129  192.168.2.7      TCP    58 36113 → 110 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
19 0.116774770  192.168.2.7      192.168.234.129  TCP    60 135 → 36113 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
20 0.116901632  192.168.234.129  192.168.2.7      TCP    54 36113 → 135 [RST] Seq=1 Win=0 Len=0
21 0.117087890  192.168.2.7      192.168.234.129  TCP    60 139 → 36113 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
22 0.117169320  192.168.234.129  192.168.2.7      TCP    54 36113 → 139 [RST] Seq=1 Win=0 Len=0
23 0.121470797  192.168.234.129  192.168.2.7      TCP    58 36113 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
24 0.121764341  192.168.234.129  192.168.2.7      TCP    58 36113 → 21 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
25 0.121935219  192.168.234.129  192.168.2.7      TCP    58 36113 → 1723 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
26 0.122266028  192.168.234.129  192.168.2.7      TCP    58 36113 → 995 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
27 1.213011986  192.168.234.129  192.168.2.7      TCP    58 36115 → 995 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
```

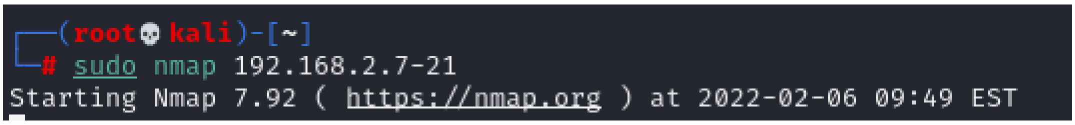
- Below are the packets captured for PORT 135



tcp.port == 135

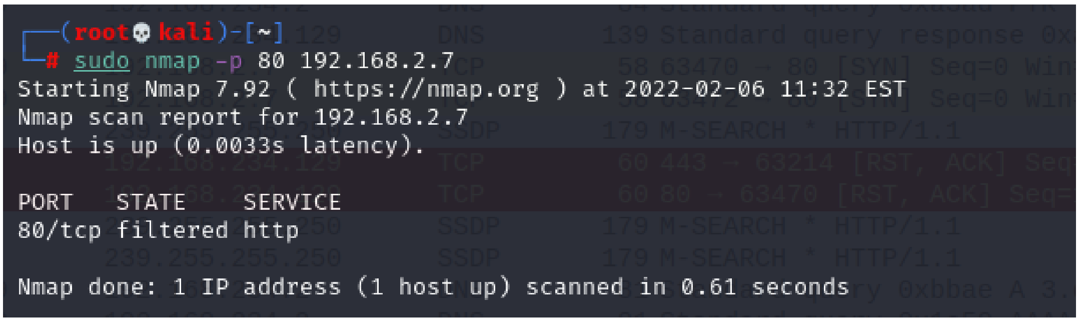
No.	Time	Source	Destination	Protocol	Length	Info
16	0.112447957	192.168.234.129	192.168.2.7	TCP	58	36113 → 135 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
19	0.116774770	192.168.2.7	192.168.234.129	TCP	60	135 → 36113 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
20	0.116901632	192.168.234.129	192.168.2.7	TCP	54	36113 → 135 [RST] Seq=1 Win=0 Len=0
51	1.417271847	192.168.234.129	192.168.2.7	TCP	58	36118 → 135 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
58	1.419794339	192.168.2.7	192.168.234.129	TCP	60	135 → 36118 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
59	1.419901796	192.168.234.129	192.168.2.7	TCP	54	36118 → 135 [RST] Seq=1 Win=0 Len=0
1016	2.775479600	192.168.234.129	192.168.2.7	TCP	58	36120 → 135 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
1028	2.777784240	192.168.2.7	192.168.234.129	TCP	60	135 → 36120 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
1029	2.777839056	192.168.234.129	192.168.2.7	TCP	54	36120 → 135 [RST] Seq=1 Win=0 Len=0
2032	4.164770218	192.168.234.129	192.168.2.7	TCP	58	36122 → 135 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
3360	7.419991926	192.168.2.7	192.168.234.129	TCP	60	135 → 36122 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
3361	7.420008293	192.168.234.129	192.168.2.7	TCP	54	36122 → 135 [RST] Seq=1 Win=0 Len=0

- If we want to scan for a range of ip address then, enter the following command
- Write sudo nmap ip address range



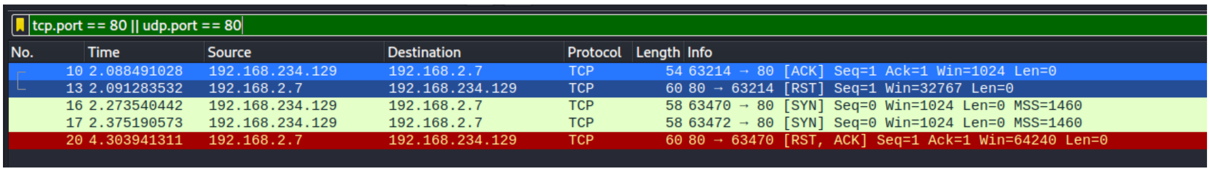
```
┌──(root💀kali)-[~]
└─# sudo nmap 192.168.2.7-21
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-06 09:49 EST
```

- You will get the result of scan for the whole range
- To know the status of a particular port, enter the following command



```
┌──(root💀kali)-[~]
└─# sudo nmap -p 80 192.168.2.7
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-06 11:32 EST
Nmap scan report for 192.168.2.7
Host is up (0.0033s latency).

PORT   STATE    SERVICE
80/tcp filtered http

Nmap done: 1 IP address (1 host up) scanned in 0.61 seconds
```

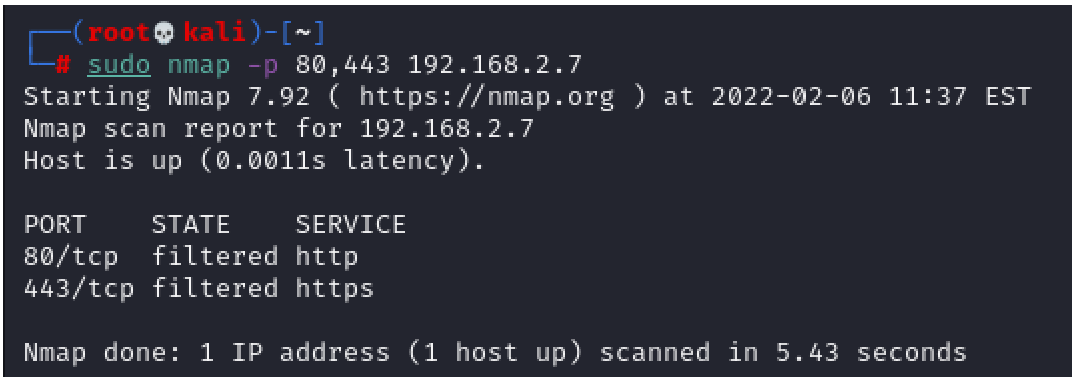
- Packets captured in wireshark



tcp.port == 80 || udp.port == 80

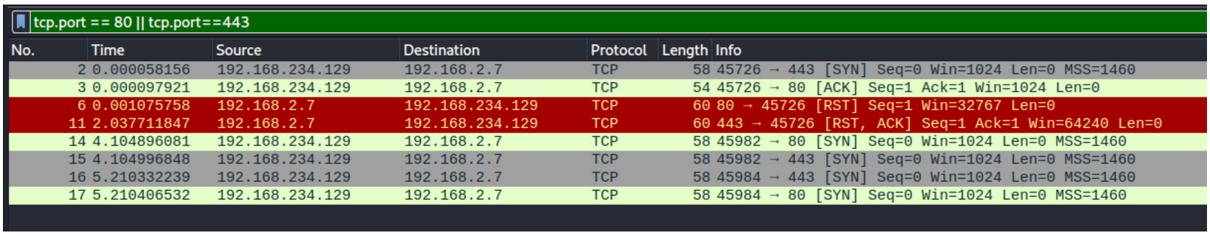
No.	Time	Source	Destination	Protocol	Length	Info
10	2.088491028	192.168.234.129	192.168.2.7	TCP	54	63214 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
13	2.091283532	192.168.2.7	192.168.234.129	TCP	60	80 → 63214 [RST] Seq=1 Win=32767 Len=0
16	2.273540442	192.168.234.129	192.168.2.7	TCP	58	63470 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
17	2.375190573	192.168.234.129	192.168.2.7	TCP	58	63472 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
20	4.303941311	192.168.2.7	192.168.234.129	TCP	60	80 → 63470 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0

- For multiple ports, type the following command



```
┌──(root💀kali)-[~]
└─# sudo nmap -p 80,443 192.168.2.7
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-06 11:37 EST
Nmap scan report for 192.168.2.7
Host is up (0.0011s latency).

PORT    STATE    SERVICE
80/tcp  filtered http
443/tcp filtered https

Nmap done: 1 IP address (1 host up) scanned in 5.43 seconds
```

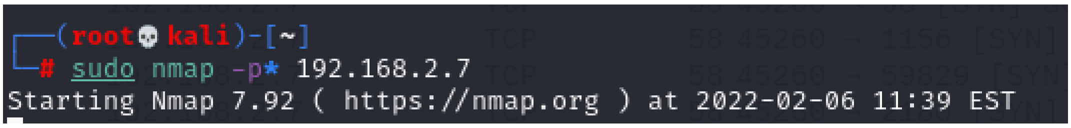
- Packets captured by Wireshark



tcp.port == 80 || tcp.port==443

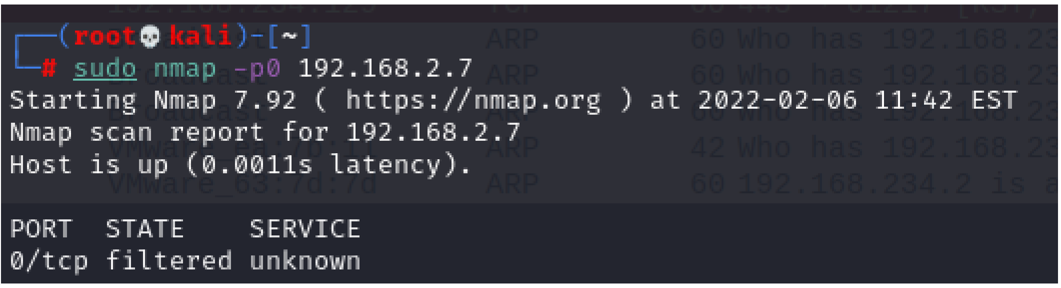
No.	Time	Source	Destination	Protocol	Length	Info
2	0.000058156	192.168.234.129	192.168.2.7	TCP	58	45726 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
3	0.000097921	192.168.234.129	192.168.2.7	TCP	54	45726 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
6	0.001075758	192.168.2.7	192.168.234.129	TCP	60	80 → 45726 [RST] Seq=1 Win=32767 Len=0
11	2.037711847	192.168.2.7	192.168.234.129	TCP	60	443 → 45726 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0
14	4.104896081	192.168.234.129	192.168.2.7	TCP	58	45982 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
15	4.104996848	192.168.234.129	192.168.2.7	TCP	58	45982 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
16	5.210332239	192.168.234.129	192.168.2.7	TCP	58	45984 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
17	5.210406532	192.168.234.129	192.168.2.7	TCP	58	45984 → 80 [SYN] Seq=0 Win=1024 Len=0 MSS=1460

- To scan all the possible ports, write the following command



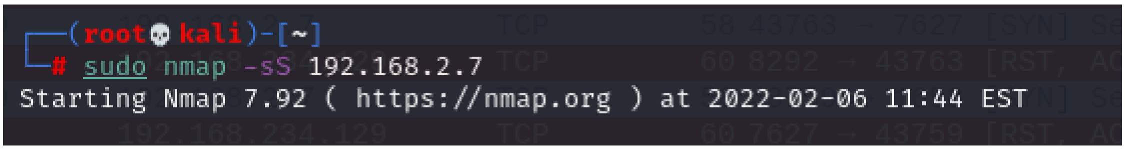
```
┌──(root💀kali)-[~]
└─# sudo nmap -p* 192.168.2.7
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-06 11:39 EST
```

- To scan for all available TCP ports, enter the following command



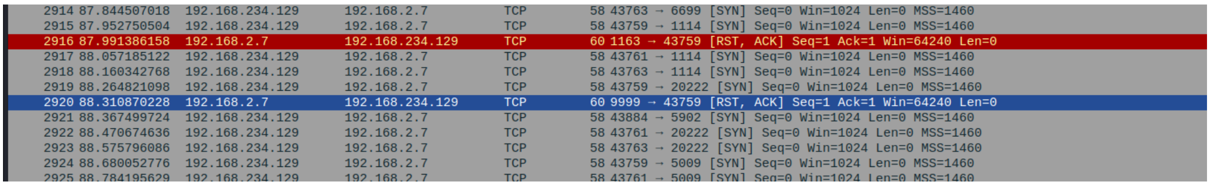
```
┌──(root💀kali)-[~]
└─# sudo nmap -p0 192.168.2.7
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-06 11:42 EST
Nmap scan report for 192.168.2.7
Host is up (0.0011s latency).

PORT  STATE    SERVICE
0/tcp filtered unknown
```

- To go for tcp syn scan, enter the following command



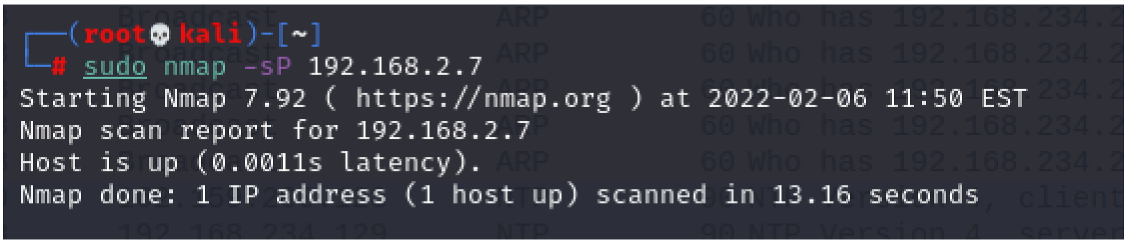
```
┌──(root💀kali)-[~]
└─# sudo nmap -sS 192.168.2.7
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-06 11:44 EST
```

- Some of the packets captured in wireshark

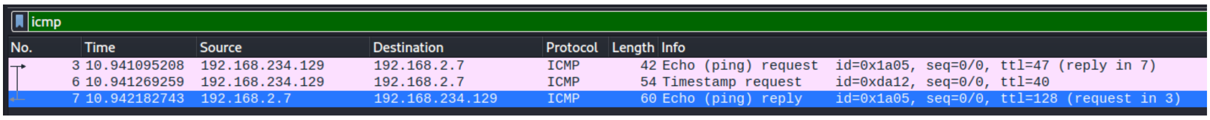


```
2914 87.844507018  192.168.234.129   192.168.2.7       TCP   58 43763 → 6699 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2915 87.952750504  192.168.234.129   192.168.2.7       TCP   58 43759 → 1114 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2916 87.991386158  192.168.2.7       192.168.234.129   TCP   60 1163 → 43759 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0
2917 88.057185122  192.168.234.129   192.168.2.7       TCP   58 43761 → 1114 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2918 88.160342768  192.168.234.129   192.168.2.7       TCP   58 43763 → 1114 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2919 88.264821098  192.168.234.129   192.168.2.7       TCP   58 43759 → 20222 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2920 88.310870228  192.168.2.7       192.168.234.129   TCP   60 9999 → 43759 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0
2921 88.367499724  192.168.234.129   192.168.2.7       TCP   58 43884 → 5902 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2922 88.470674636  192.168.234.129   192.168.2.7       TCP   58 43761 → 20222 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2923 88.575796086  192.168.234.129   192.168.2.7       TCP   58 43763 → 20222 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2924 88.680052776  192.168.234.129   192.168.2.7       TCP   58 43759 → 5009 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
2925 88.784195629  192.168.234.129   192.168.2.7       TCP   58 43761 → 5009 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
```

- To scan for ping scan, enter the following command

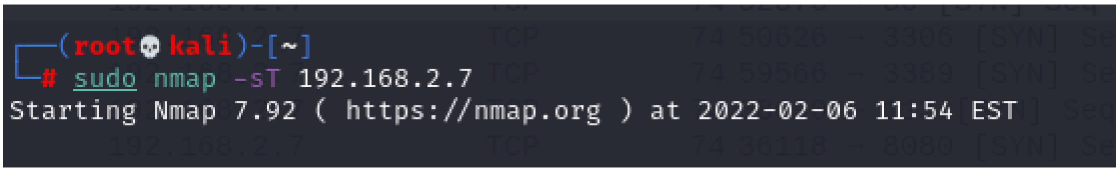


```
┌──(root💀kali)-[~]
└─# sudo nmap -sP 192.168.2.7
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-06 11:50 EST
Nmap scan report for 192.168.2.7
Host is up (0.0011s latency).
Nmap done: 1 IP address (1 host up) scanned in 13.16 seconds
```



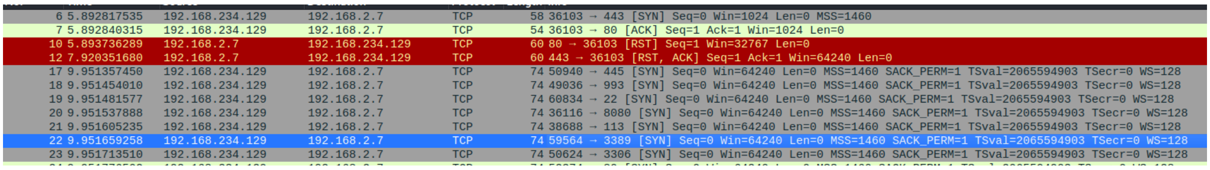
```
No.   Time          Source            Destination       Protocol  Length Info
    3 10.941095208  192.168.234.129   192.168.2.7       ICMP      42 Echo (ping) request   id=0x1a05, seq=0/0, ttl=47 (reply in 7)
    6 10.941269259  192.168.234.129   192.168.2.7       ICMP      54 Timestamp request     id=0xda12, seq=0/0, ttl=40
    7 10.942182743  192.168.2.7       192.168.234.129   ICMP      60 Echo (ping) reply     id=0x1a05, seq=0/0, ttl=128 (request in 3)
```

- For TCP Connect scan, enter the following command



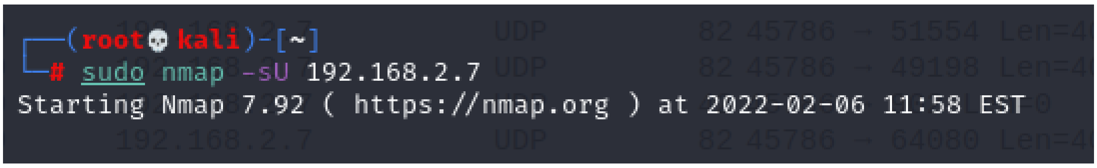
```
┌──(root💀kali)-[~]
└─# sudo nmap -sT 192.168.2.7
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-06 11:54 EST
```

- Wireshark packets captured



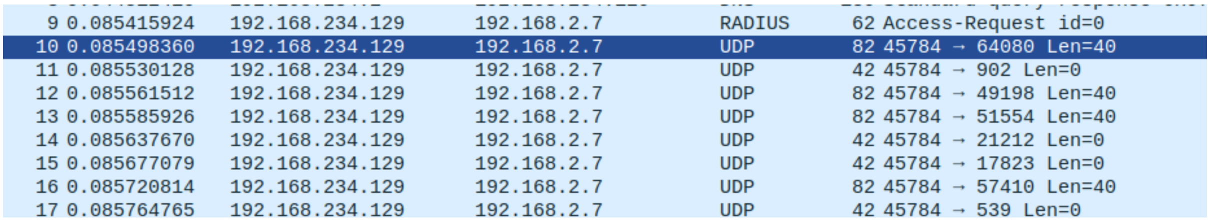
```
 6 5.892817535  192.168.234.129  192.168.2.7      TCP  58 36103 → 443 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
 7 5.892840315  192.168.234.129  192.168.2.7      TCP  54 36103 → 80 [ACK] Seq=1 Ack=1 Win=1024 Len=0
10 5.893736289  192.168.2.7      192.168.234.129  TCP  60 80 → 36103 [RST] Seq=1 Win=32767 Len=0
12 7.920351680  192.168.2.7      192.168.234.129  TCP  60 443 → 36103 [RST, ACK] Seq=1 Ack=1 Win=64240 Len=0
17 9.951357450  192.168.234.129  192.168.2.7      TCP  74 50940 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2065594903 TSecr=0 WS=128
18 9.951454010  192.168.234.129  192.168.2.7      TCP  74 49036 → 993 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2065594903 TSecr=0 WS=128
19 9.951481577  192.168.234.129  192.168.2.7      TCP  74 60834 → 22 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2065594903 TSecr=0 WS=128
20 9.951537888  192.168.234.129  192.168.2.7      TCP  74 36116 → 8080 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2065594903 TSecr=0 WS=128
21 9.951605235  192.168.234.129  192.168.2.7      TCP  74 38688 → 113 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2065594903 TSecr=0 WS=128
22 9.951659258  192.168.234.129  192.168.2.7      TCP  74 59564 → 3389 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2065594903 TSecr=0 WS=128
23 9.951713510  192.168.234.129  192.168.2.7      TCP  74 50624 → 3306 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=2065594903 TSecr=0 WS=128
```

- For UDP Scan, use the following command



```
┌──(root💀kali)-[~]
└─# sudo nmap -sU 192.168.2.7
Starting Nmap 7.92 ( https://nmap.org ) at 2022-02-06 11:58 EST
```

- From the wireshark



```
 9 0.085415924   192.168.234.129   192.168.2.7   RADIUS   62 Access-Request id=0
10 0.085498360   192.168.234.129   192.168.2.7   UDP      82 45784 → 64080 Len=40
11 0.085530128   192.168.234.129   192.168.2.7   UDP      42 45784 → 902 Len=0
12 0.085561512   192.168.234.129   192.168.2.7   UDP      82 45784 → 49198 Len=40
13 0.085585926   192.168.234.129   192.168.2.7   UDP      82 45784 → 51554 Len=40
14 0.085637670   192.168.234.129   192.168.2.7   UDP      42 45784 → 21212 Len=0
15 0.085677079   192.168.234.129   192.168.2.7   UDP      42 45784 → 17823 Len=0
16 0.085720814   192.168.234.129   192.168.2.7   UDP      82 45784 → 57410 Len=40
17 0.085764765   192.168.234.129   192.168.2.7   UDP      42 45784 → 539 Len=0
```

CONCLUSION:

- By performing the above practical, I learnt about the basics of nmap and it's functionalities