Task 1

Objective: Learn to discover open ports on devices in your local network to understand network exposure.

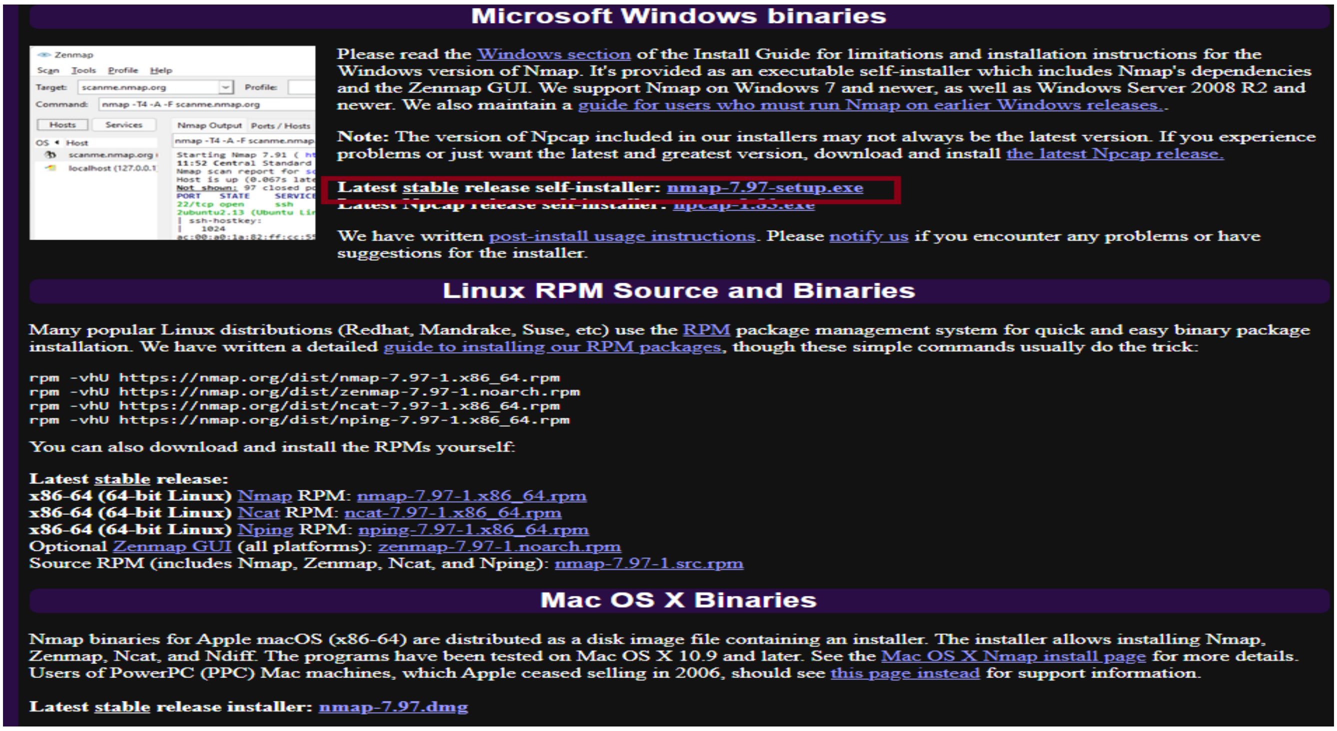
Tools: Nmap, Wireshark, Kali Linux (OS).

Key Concepts: Port scanning, TCP SYN scan, IP ranges, network reconnaissance, open ports, network security basics

Execution:

Note: I completed the task in kali linux (Debian based OS, installed in windows through virtual box)

Step-1: Installing Nmap from official website. For windows, we have to install it from the official site. But in Kali Linux Nmap is already registered. Make sure it is update to latest level.



Microsoft Windows binaries

Please read the Windows section of the Install Guide for limitations and installation instructions for the Windows version of Nmap. It's provided as an executable self-installer which includes Nmap's dependencies and the Zenmap GUI. We support Nmap on Windows 7 and newer, as well as Windows Server 2008 R2 and newer. We also maintain a guide for users who must run Nmap on earlier Windows releases.

Note: The version of Npcap included in our installers may not always be the latest version. If you experience problems or just want the latest and greatest version, download and install the latest Npcap release.

Latest stable release self-installer: nmap-7.97-setup.exe
Latest Npcap release self-installer: npcap-1.83.exe

We have written post-install usage instructions. Please notify us if you encounter any problems or have suggestions for the installer.

Linux RPM Source and Binaries

Many popular Linux distributions (Redhat, Mandrake, Suse, etc) use the RPM package management system for quick and easy binary package installation. We have written a detailed guide to installing our RPM packages, though these simple commands usually do the trick:

```
rpm -vhU https://nmap.org/dist/nmap-7.97-1.x86_64.rpm
rpm -vhU https://nmap.org/dist/zenmap-7.97-1.noarch.rpm
rpm -vhU https://nmap.org/dist/ncat-7.97-1.x86_64.rpm
rpm -vhU https://nmap.org/dist/nping-7.97-1.x86_64.rpm
```

You can also download and install the RPMs yourself:

Latest stable release:
x86-64 (64-bit Linux) Nmap RPM: nmap-7.97-1.x86_64.rpm
x86-64 (64-bit Linux) Ncat RPM: ncat-7.97-1.x86_64.rpm
x86-64 (64-bit Linux) Nping RPM: nping-7.97-1.x86_64.rpm
Optional Zenmap GUI (all platforms): zenmap-7.97-1.noarch.rpm
Source RPM (includes Nmap, Zenmap, Ncat, and Nping): nmap-7.97-1.src.rpm

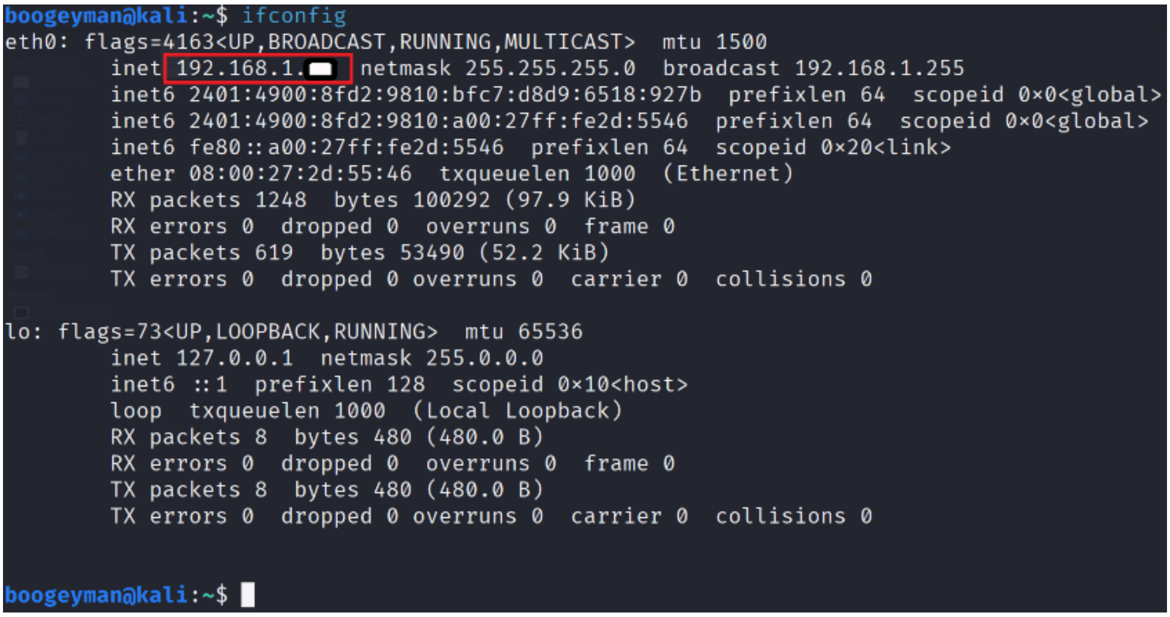
Mac OS X Binaries

Nmap binaries for Apple macOS (x86-64) are distributed as a disk image file containing an installer. The installer allows installing Nmap, Zenmap, Ncat, and Ndiff. The programs have been tested on Mac OS X 10.9 and later. See the Mac OS X Nmap install page for more details. Users of PowerPC (PPC) Mac machines, which Apple ceased selling in 2006, should see this page instead for support information.

Latest stable release installer: nmap-7.97.dmg

(*Fig. 1*)

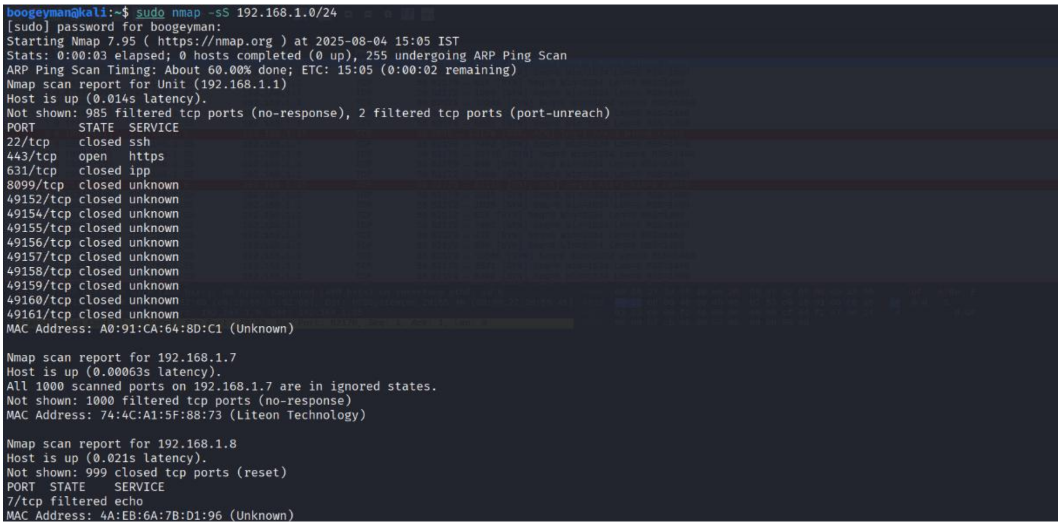
Step-2: Once updated, we can find our local IP range. Use *ifconfig* command to get the IP range.



```
boogeyman@kali:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.[redacted]  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 2401:4900:8fd2:9810:bfc7:d8d9:6518:927b  prefixlen 64  scopeid 0x0<global>
        inet6 2401:4900:8fd2:9810:a00:27ff:fe2d:5546  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::a00:27ff:fe2d:5546  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:2d:55:46  txqueuelen 1000  (Ethernet)
        RX packets 1248  bytes 100292 (97.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 619  bytes 53490 (52.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 8  bytes 480 (480.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 480 (480.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

boogeyman@kali:~$ ▌
```

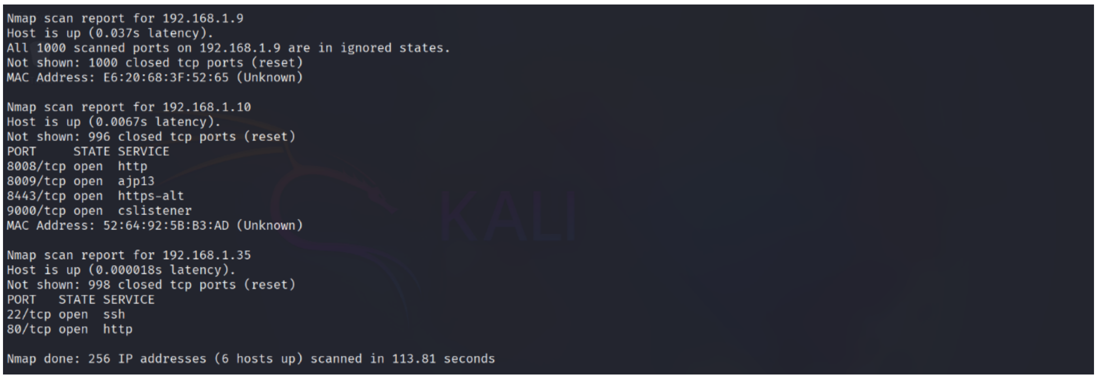
(*Fig. 2*)

Step-3: Now for the perform TCP Syn scan. Use *sudo nmap -sS 192.168.1.0/24*. The command "sudo" is required to execute commands that requires administrator level permission.



```
boogeyman@kali:~$ sudo nmap -sS 192.168.1.0/24
[sudo] password for boogeyman:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-04 15:05 IST
Stats: 0:00:03 elapsed; 0 hosts completed (0 up), 255 undergoing ARP Ping Scan
ARP Ping Scan Timing: About 60.00% done; ETC: 15:05 (0:00:02 remaining)
Nmap scan report for Unit (192.168.1.1)
Host is up (0.014s latency).
Not shown: 985 filtered tcp ports (no-response), 2 filtered tcp ports (port-unreach)
PORT      STATE  SERVICE
22/tcp    closed ssh
443/tcp   open   https
631/tcp   closed ipp
8099/tcp  closed unknown
49152/tcp closed unknown
49154/tcp closed unknown
49155/tcp closed unknown
49156/tcp closed unknown
49157/tcp closed unknown
49158/tcp closed unknown
49159/tcp closed unknown
49160/tcp closed unknown
49161/tcp closed unknown
MAC Address: A0:91:CA:64:8D:C1 (Unknown)

Nmap scan report for 192.168.1.7
Host is up (0.00063s latency).
All 1000 scanned ports on 192.168.1.7 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
MAC Address: 74:4C:A1:5F:88:73 (Liteon Technology)

Nmap scan report for 192.168.1.8
Host is up (0.021s latency).
Not shown: 999 closed tcp ports (reset)
PORT  STATE    SERVICE
7/tcp filtered echo
MAC Address: 4A:EB:6A:7B:D1:96 (Unknown)
```

(*Fig.3*)

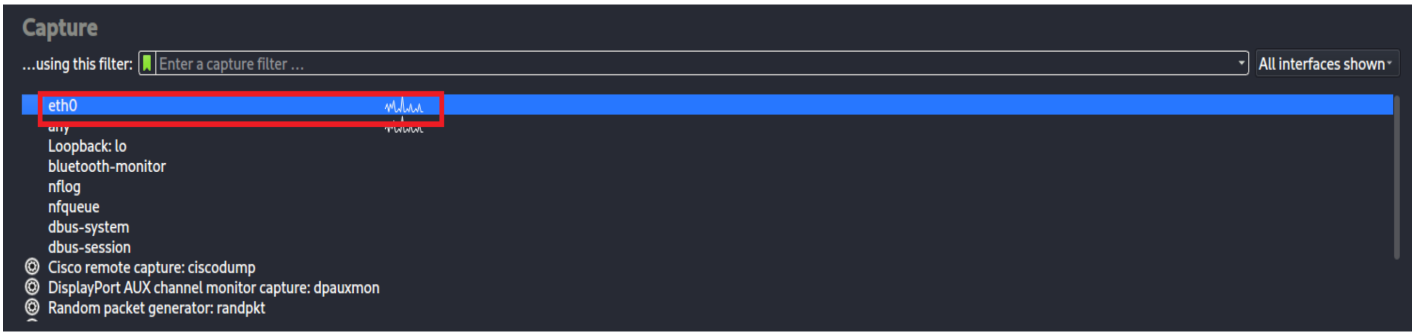


```
Nmap scan report for 192.168.1.9
Host is up (0.037s latency).
All 1000 scanned ports on 192.168.1.9 are in ignored states.
Not shown: 1000 closed tcp ports (reset)
MAC Address: E6:20:68:3F:52:65 (Unknown)

Nmap scan report for 192.168.1.10
Host is up (0.0067s latency).
Not shown: 996 closed tcp ports (reset)
PORT     STATE SERVICE
8008/tcp open  http
8009/tcp open  ajp13
8443/tcp open  https-alt
9000/tcp open  cslistener
MAC Address: 52:64:92:5B:B3:AD (Unknown)

Nmap scan report for 192.168.1.35
Host is up (0.000018s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 256 IP addresses (6 hosts up) scanned in 113.81 seconds
```

(*Fig. 4*)

Step-4: Now for packet analysis, we need to sue **wireshark**. We can install wireshark from official website for windows. For Kali linux, it is preinstalled. Make sure it is update to latest level.

Step-5: We can start wireshark from terminal or we can start it from start option.



Capture

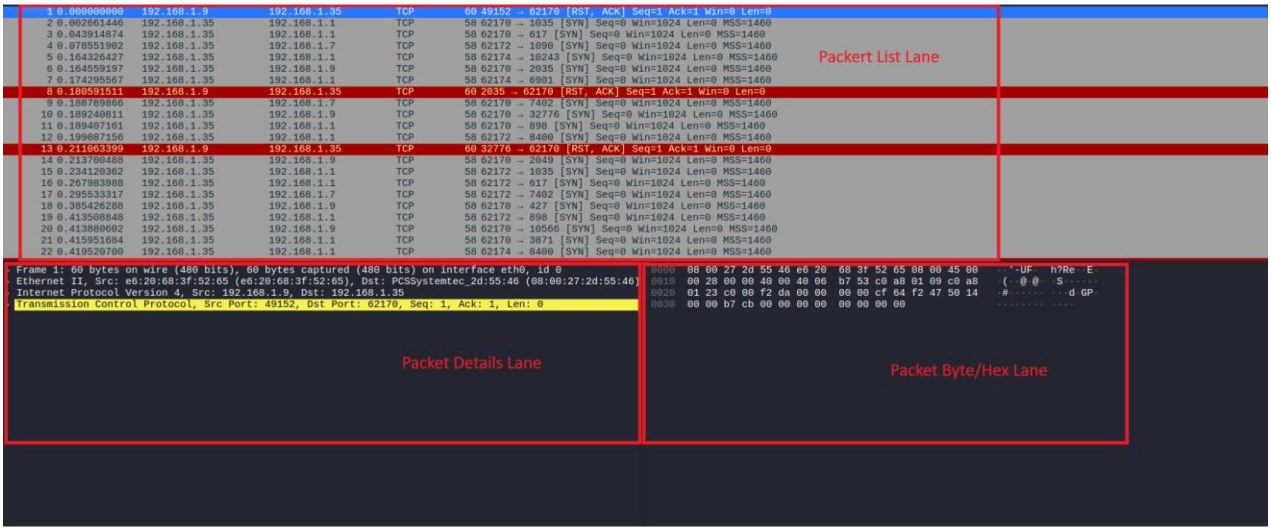
...using this filter: Enter a capture filter ... All interfaces shown

eth0
any
Loopback: lo
bluetooth-monitor
nflog
nfqueue
dbus-system
dbus-session
Cisco remote capture: ciscodump
DisplayPort AUX channel monitor capture: dpauxmon
Random packet generator: randpkt

(*Fig. 5*)

Note: The eth0 is the interface for our localhost to capture packets.

Step-7: After selecting the *eth0* interface, the packet capturing starts.



```
 1 0.000000000  192.168.1.9   192.168.1.35  TCP  60 49152 → 62170 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
 2 0.002661446  192.168.1.35  192.168.1.1   TCP  58 62170 → 1035 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
 3 0.043914874  192.168.1.35  192.168.1.1   TCP  58 62170 → 617 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
 4 0.078551902  192.168.1.35  192.168.1.7   TCP  58 62172 → 1090 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
 5 0.164326427  192.168.1.35  192.168.1.1   TCP  58 62174 → 10243 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
 6 0.164559197  192.168.1.35  192.168.1.9   TCP  58 62170 → 2035 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
 7 0.174295567  192.168.1.35  192.168.1.1   TCP  58 62174 → 6901 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
 8 0.180591511  192.168.1.9   192.168.1.35  TCP  60 2035 → 62170 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
 9 0.188789866  192.168.1.35  192.168.1.7   TCP  58 62170 → 7402 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
10 0.189240811  192.168.1.35  192.168.1.9   TCP  58 62170 → 32776 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
11 0.189407161  192.168.1.35  192.168.1.1   TCP  58 62170 → 898 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
12 0.199087156  192.168.1.35  192.168.1.1   TCP  58 62172 → 8400 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
13 0.211063399  192.168.1.9   192.168.1.35  TCP  60 32776 → 62170 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
14 0.213700488  192.168.1.35  192.168.1.9   TCP  58 62170 → 2049 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
15 0.234120362  192.168.1.35  192.168.1.1   TCP  58 62172 → 1035 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
16 0.267983988  192.168.1.35  192.168.1.1   TCP  58 62172 → 617 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
17 0.295533317  192.168.1.35  192.168.1.7   TCP  58 62172 → 7402 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
18 0.385426288  192.168.1.35  192.168.1.9   TCP  58 62170 → 427 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
19 0.413508848  192.168.1.35  192.168.1.1   TCP  58 62172 → 898 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
20 0.413880602  192.168.1.35  192.168.1.9   TCP  58 62170 → 10566 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
21 0.415951684  192.168.1.35  192.168.1.1   TCP  58 62170 → 3871 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
22 0.419520700  192.168.1.35  192.168.1.1   TCP  58 62174 → 8400 [SYN] Seq=0 Win=1024 Len=0 MSS=1460
```

```
Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface eth0, id 0
Ethernet II, Src: e6:20:68:3f:52:65 (e6:20:68:3f:52:65), Dst: PCSSystemtec_2d:55:46 (08:00:27:2d:55:46)
Internet Protocol Version 4, Src: 192.168.1.9, Dst: 192.168.1.35
Transmission Control Protocol, Src Port: 49152, Dst Port: 62170, Seq: 1, Ack: 1, Len: 0
```

```
0000  08 00 27 2d 55 46 e6 20  68 3f 52 65 08 00 45 00   ··'-UF·  h?Re··E·
0010  00 28 00 00 40 00 40 06  b7 53 c0 a8 01 09 c0 a8   ·(··@·@·  ·S······
0020  01 23 c0 00 f2 da 00 00  00 00 cf 64 f2 47 50 14   ·#······  ···d·GP·
0030  00 00 b7 cb 00 00 00 00  00 00 00 00               ········  ····
```

(Fig. 6)

Note: The highlighted region in the wireshark interface described in the image. The packets coloured in red at the *rst* flag/request.

Step-6: Some common services running on those ports. The open ports are Ssh (Secure Shell) on port 22 and Apache2 on port 80.

- The SSH (Secure Shell) is a cryptographic network protocol primarily used for securely accessing and managing devices and servers over an unsecured network. It is widely used by system administrators and developers for several purposes.
- Apache2 (Apache HTTP Server), which operates on port 80, is a crucial component of web hosting. Port 80 is the standard port for HTTP (Hypertext Transfer Protocol) traffic on the web. When one enters a web address (e.g., http://example.com) in your web browser, your web browser is connected to the web server on port 80 by default.

Step-7: Identification of potential security risks from open ports

1. Port 22 (SSH) –
   a. Brute-Force Attacks: Attackers might try guessing passwords and usernames through automated brute-force scripts.
   b. Taking advantage of Vulnerabilities: If SSH server software is outdated, known vulnerabilities can be exploited to obtain unauthorized access.
   c. Weak Authentication: Utilizing weak passwords or password-based authentication (rather than key-based) poses threat.
2. Port 80 (HTTP) –
   a. **Threats Unencrypted Traffic:** All, such as credentials or sensitive data, is sent in plaintext and can be intercepted and therefore insecure against man-in-the-middle or sniffing attacks.
   b. **Web Application Vulnerabilities:** Open port 80 usually refers to a web server (i.e., Apache). If the web site or server software is vulnerable (e.g., XSS, SQL injection, code execution vulnerabilities), these can be exploited by attackers.
   c. Reconnaissance involves attackers probing the web server to obtain version information and known vulnerabilities, thereby preparing for targeted attacks.

Step-8: Save scan results as a text or HTML file.

The scan results are saved in a file named nmap_res.txt