Task 1: Scan Your Local Network for Open Ports

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

Tools: Nmap (free), Wireshark (optional), VirtualBox virtualization (Type 2), Kali Linux distro, Metasploitable

1. Full Network Scan – nmap 192.168.1.0/24

```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.1.0/24 -oN network_scan.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-26 09:55 EDT
Nmap scan report for Unit (192.168.1.1)
Host is up (0.0028s latency).
Not shown: 984 filtered tcp ports (no-response), 2 filtered tcp ports (port-unreach)
PORT     STATE  SERVICE
22/tcp   closed ssh
443/tcp  open   https
445/tcp  closed microsoft-ds
8099/tcp closed unknown
49152/tcp closed unknown
49153/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
49163/tcp closed unknown
MAC Address: 24:DE:8A:06:51:C1 (Unknown)
```

Nmap scan report for 192.168.1.2

Host is up (0.0078s latency).

All 1000 scanned ports on 192.168.1.2 are in ignored states.

Not shown: 1000 closed tcp ports (reset)

MAC Address: 62:DB:70:9B:F5:1F (Unknown)

Nmap scan report for 192.168.1.4

Host is up (0.013s 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: 80:5E:4F:86:74:86 (FN-Link Technology Limited)

Nmap scan report for 192.168.1.5

Host is up (0.18s latency).

All 1000 scanned ports on 192.168.1.5 are in ignored states.

Not shown: 1000 closed tcp ports (reset)

MAC Address: CA:16:CF:76:2D:F4 (Unknown)

Nmap scan report for 192.168.1.6

Host is up (0.0099s latency).

All 1000 scanned ports on 192.168.1.6 are in ignored states.

Not shown: 1000 closed tcp ports (reset)

MAC Address: EE:2F:96:08:3A:72 (Unknown)

Nmap scan report for 192.168.1.7

Host is up (0.00040s latency).

All 1000 scanned ports on 192.168.1.7 are in ignored states.

Not shown: 1000 filtered tcp ports (no-response)

MAC Address: A0:29:42:68:67:D0 (Intel Corporate)

Nmap scan report for 192.168.1.9

Host is up (0.0010s latency).

Not shown: 977 closed tcp ports (reset)

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: 08:00:27:76:14:CB (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.1.12

Host is up (0.0000030s latency).

Not shown: 999 closed tcp ports (reset)

PORT STATE SERVICE

21/tcp open ftp

Nmap done: 256 IP addresses (8 hosts up) scanned in 110.75 seconds

Scan results summary

IP Address	Open Ports & Services
192.168.1.1 Router	443/https
192.168.1.2	None (all closed)
192.168.1.4 IOS device	8008/http, 8009/ajp13, 8443/https-alt, 9000/cslistener
192.168.1.5	None (all closed)
192.168.1.6	None (all closed)
192.168.1.7 **Host Machine**	None (all filtered)
192.168.1.9 Metasploitable VM	21/ftp, 22/ssh, 23/telnet, 25/smtp, 53/domain, 80/http, 111/rpcbind, 139/netbios-ssn, 445/microsoft-ds, 512/exec, 513/login, 514/shell, 1099/rmiregistry, 1524/ingreslock, 2049/nfs, 2121/ccproxy-ftp, 3306/mysql, 5432/postgresql, 5900/vnc, 6000/X11, 6667/irc, 8009/ajp13, 8180/unknown
192.168.1.12 Kali linux VM	21/ftp

2. Metasploitable machine is running on IP 192.168.1.9

```
┌──(kali㉿kali)-[~]
└─$ nmap -sT 192.168.1.9
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-26 10:06 EDT
Nmap scan report for 192.168.1.9
Host is up (0.011s 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: 08:00:27:76:14:CB (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

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

WireShark usage in the analysis

- To effectively utilize Wireshark, I apply the filter **ip.src == 192.168.1.12 && ip.dst == 192.168.1.9**, which allows me to focus exclusively on the network traffic between the specific source and destination of interest.
- While this filter significantly reduces extraneous data and streamlines the analysis process, an additional step is often required to gain a comprehensive view of the communication.
- By following the TCP stream, I can reconstruct the exact sequence of interactions and packet exchanges that occurred between the scanner and the target, providing detailed insight into the underlying protocol behavior and connection establishment.

3. Tcp Connect scan -sT
Full tcp three-way handshake(SYN, SYN-ACK, ACK OR RST-ACK)

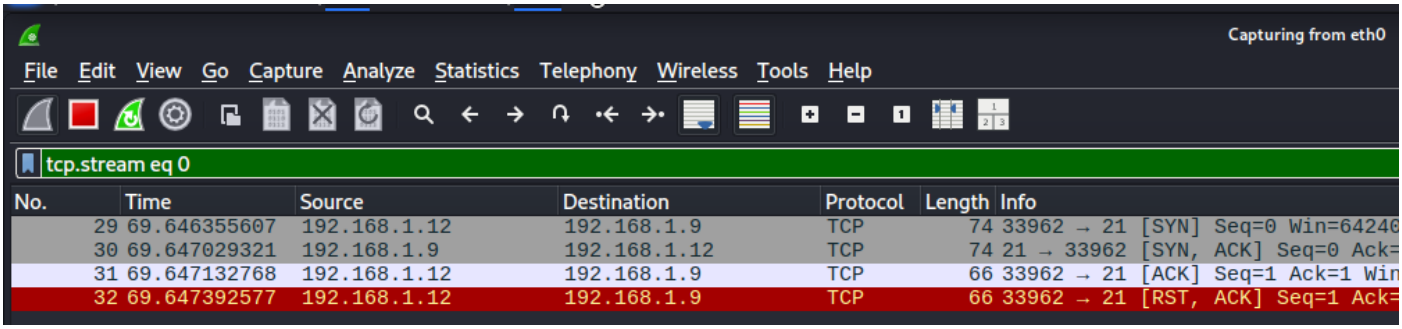
```
┌──(kali㉿kali)-[~]
└─$ nmap -sT 192.168.1.9 -p 21
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-26 10:11 EDT
Nmap scan report for 192.168.1.9
Host is up (0.0011s latency).

PORT   STATE SERVICE
21/tcp open  ftp
MAC Address: 08:00:27:76:14:CB (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.16 seconds

┌──(kali㉿kali)-[~]
└─$
```

- **Traffic captured on wireshark**



As you can see, the Nmap scanner is completing the handshake itself after receiving the SYN-ACK from the target.

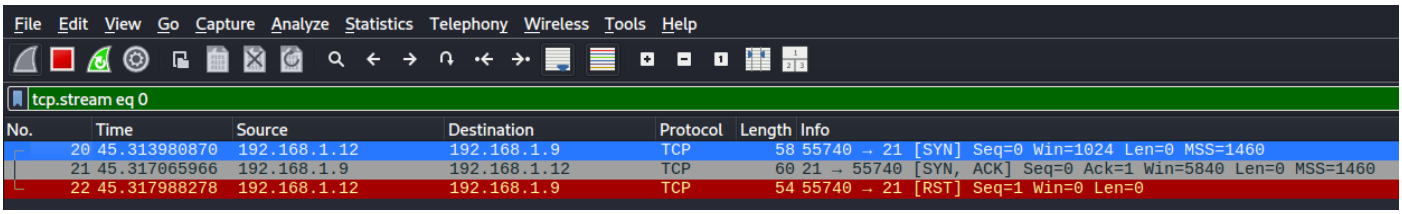
4. Tcp SYN SCAN -sS

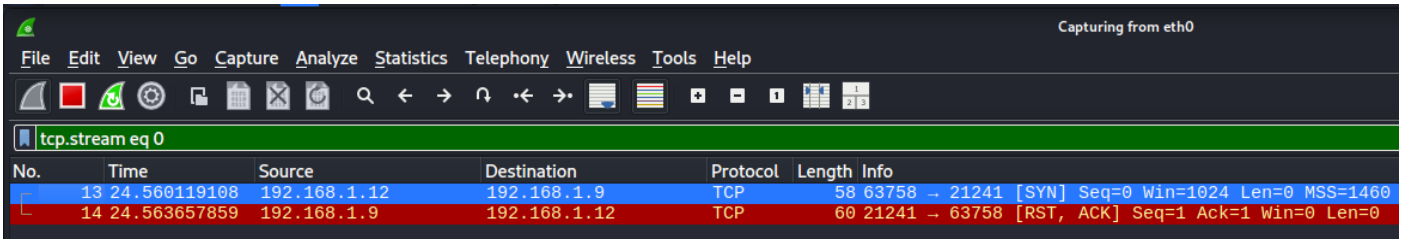
```
┌──(kali㉿kali)-[~]
└─$ nmap -sS 192.168.1.9 -p 21
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-26 11:01 EDT
Nmap scan report for 192.168.1.9
Host is up (0.0041s latency).

PORT   STATE SERVICE
21/tcp open  ftp
MAC Address: 08:00:27:76:14:CB (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.37 seconds
```

- **Traffic captured on wireshark**





The target sent back a [SYN, ACK], indicating that the port is open. If the port were closed, the target would have replied with a [RST, ACK].

5. Common Services Running on Open Ports

a. the services running on open ports. Each port typically corresponds to a specific protocol or service. For example:

- **Port 21 (FTP):** Used for File Transfer Protocol, enabling file uploads and downloads, but transmits data and credentials in plain text.
- **Port 22 (SSH):** Secure Shell for encrypted remote administration; while secure by design, it is a frequent target for brute-force attacks if left on the default port.
- **Port 23 (Telnet):** Provides remote shell access but transmits all data, including credentials, in clear text, making it highly insecure.
- **Port 25 (SMTP):** Used for email transmission; if not properly secured, it can be exploited for spam or to intercept sensitive information.
- **Port 53 (DNS):** Handles domain name resolution; vulnerable to DDoS and amplification attacks if exposed.
- **Port 80 (HTTP) and 443 (HTTPS):** Web services; susceptible to web application vulnerabilities such as XSS, SQL injection, and DDoS attacks.
- **Other ports (e.g., 139/445 for SMB, 3306 for MySQL, 5900 for VNC):** Each represents a specific service, often with its own set of vulnerabilities if exposed to untrusted networks

6. Based on the open ports detected in my scan, my network is exposed to several well-known risks:

- Plain text protocols (**FTP, Telnet, SMTP**) expose credentials and sensitive data.
- Default ports **(SSH, SMB, HTTP**) are frequent targets for automated attacks and exploits.
- Open database and remote desktop ports can lead to data breaches and unauthorized access if not properly secured.
- Unpatched or misconfigured services could be exploited for lateral movement, privilege escalation, or denial-of-service attacks.