

Data Communications and Systems Lab File

Name: Kshitij Chandrakar
SAP: 500124827
Batch: 5
Submitted To: Richa Kumari

Date	Content	Page. No	Attendance	Signature
22/01/25	Experiment 1	1 - 5	Present	
29/01/25	Experiment 2	6 - 8	Present	
3/02/25	Experiment 3	9 - 14	Absent. Present	
5/02/25	Experiment 4	15 - 18	Present	
5/02/25	Experiment 5	19 - 22	Present	Richa 2/2/25
19/02/25	Experiment 6	23 - 30	Present	
12/03/25	Experiment 7	31 - 37	Present	
12/03/25	Experiment 8	38 - 39	Present	
26/03/25	Experiment 9	40 - 53	Present	
02/04/25	Experiment 10	55 - 62	Present	
09/04/25	Experiment 11	63 - 77	Absent. Present	
16/04/25	Experiment 12	78 - 87	Present	
23/04/25	Class Test and Viva	88 - 91	Present	

Remarks — 12/04

Assignment - 1

Q1 5-Networking Devices :-

1. Router :- It is used to connect multiple networks and route packets between them. Provides security features like firewalls, NAT, and VPN support.
2. Switch :- Used to connect multiple devices in a LAN. Uses MAC addresses to forward data frames and reduces network congestion by sending data only to the intended recipient. It supports VLANs.
3. Modem :- Converts digital signals from a computer into analog signals for transmission over telephone lines. Connects to an ISP to provide Internet access. Integrated with routers in modern setups.
4. Wireless Access Point :- Allows wireless devices to connect to a wired network via WiFi. Extends network coverage without more cabling.
5. Firewalls :- Monitors and controls incoming/outgoing network traffic based on security rules. Blocks unauthorized access while permitting legitimate communication and protects against cyber attacks.

Q8:- Difference Between a Switch and Hub.

Hub

Switch

- | | |
|--|--|
| 1. Works on the Physical layers | works on Data-Link layer. |
| 2. Broadcasts Data to all connected Devices | forwards only to Intended Recipient |
| 3. Slower Due to collisions and causes Congestion | faster Due to No collisions and full Duplex Connection |
| 4. All Devices can see Traffic and is Therefore, less Secure | Isolation of Traffic leads to more Security. |

A. Difference Between Router and Modem

Modem

Router.

- 1. Connects to ISP and converts signal Routes Data Between Devices and Network and Internet
- 2. Physical Layer Network Layer.
- 3. Converts Digital Signals to Analog and vice versa Works with Digital Data only
- 4. Gets a public IP from ISP Assigns private IP Addresses and uses NAT to Share a single public IP
- 5. Doesn't include security features Includes firewall, VPN, port forwarding and access control
- 6. Doesn't include WLAN WiFi. Includes WiFi

B. NIC :- a Network Interface Card allows a computer to connect to a network. Its primary role is to facilitate communication between the computer and the network devices. It assigns a MAC Address, and supports network protocols.

Q3 A. How to connect a computer to a switch
Using an Ethernet cable.

1. Gather the Hardware
 - Ethernet cable
 - Switch
 - Computer

2. Power on the switch using its Adapter
3. Plug in one end of the Ethernet cable into the computer's Ethernet port
4. Plug other end into an available port on the switch
5. Check status light on the switch to check its status.

B. Configure IP on ~~Windows~~ windows
use the following to check IP

PS> ipconfig

Configure IP

PS> netsh interface ip set address name="Ethernet"
Static 192.168.1.100 255.255.255.0 192.168.1.1
Set IP Subnet Mask Default Gateway

~~Q~~ A. Modem:- Connects to the ISP and converts
Digital connection.

Switch:- connects Multiple devices together

WAP:- Provides Wi-Fi for Portable Devices

Firewall:- filters Incoming and Outgoing traffic.

VPN Server:- Encrypts Remote Connections.

B. What is a firewall?

A firewall acts as a Digital Barrier Between a trusted Internal Network and External Connections. It does the following:

1. Blocks Unauthorised Access
2. Filters Malicious Traffic
3. Prevents Data Breaches
4. Controls Network Access

How Does it Work?

It uses Various techniques like

1. Packet filtering
2. Stateful Inspection
3. Application-level Inspection
4. Next Generation firewall like Deep packet Inspection, Intrusion Prevention

Assignment - 2

Q1 What is Bit Stuffing?

Bit Stuffing is used in Data communication protocols to ensure Reliable transmission by inserting Extra bits into the Data Stream, it helps the receiver Distinguish Between Data and control signals.

Why is it Necessary?

1. Prevents frame Delimiter Conflicts
2. Ensures Synchronization
3. Avoids Control Signal Ambiguity
4. Improves Error Detection.

Q2 Difference Between Bit Stuffing and Byte Stuffing.

1. Bit Stuffing :- Used in Bit Oriented protocols

→ Inserts an Extra bit to prevent confusion with reserved Bit Patterns

Ex :- Using HDLC Protocol

Original Data - 0 1 1 1 1 0 1

Stuffed Data - 0 1 1 1 1 0 0 1

Q2 Byte Stuffing :- Insert an Extra Byte before reserved bytes like ESC, STX etc and is used in Byte oriented protocols.

for Example :-

Original Data :- Flag 7E
~~Flag~~ Stuffed Data :- ESC SD

Q3

Given Data

0 111110 1101111 11011110 10

Stuffed Data

[Insert 0 after 5 '1's]

0 111110 1011011110 110111010
 Stuffed Bits

Destuffed Data :-

0111110 1101111 1101111010

Assignment 3

Q5

Advantages and Disadvantages of Bit Stuffing.

Advantages

1. Prevents Delimiter Conflicts
2. Simple Hardware Implementation
3. Low Overhead
4. Works in continuous B.T Stream
5. Improves Synchronization

Disadvantages

1. Adds small overhead because transmission time is slightly increased.
2. Not Human Readable
3. Limited to Bit Oriented Protocols
4. Inefficient for Random Data

Assignment - 3

Q1 Compute CRC for 1110 001 with polynomial 1001

Polynomial 1001 Represents $x^3 + 1$, which is degree 3. Therefore padded Bits = 3

Padded Message = 111 001 000

Binary Division

$$\begin{array}{r}
 1001 \quad | \quad 1110 \quad 0100 \quad 0 \quad | \\
 \underline{\oplus} \quad 1001 \\
 \hline
 01110 \\
 \underline{\oplus} \quad 1001 \\
 \hline
 0111 \\
 \underline{\oplus} \quad 1001 \\
 \hline
 01100 \\
 \underline{\oplus} \quad 1001 \\
 \hline
 01010 \\
 \underline{\oplus} \quad 1001 \\
 \hline
 0011
 \end{array}$$

Remainder = 011

$$\therefore CRC = 011$$

Transmitted Message = 111 001 011

Q2 Given Generator Polynomial = 111
 Message = 1001101

Find CRC and Transm. Hd. Message

Generator polynomial = 111. Represents $x^2 + x + 1$
 of degree 2
 \therefore CRC length = 2

\therefore Padded Message = 1001101100

Binary Division

111 | 100 110 100

111

0111

111

000101

111

0100

111

011

Remainder = 011

CRC = 11

Transm. Hd. Message = 100110111

Q3

Given Message = 101 001

Polynomial = 11011

Received Message = 101 001 1001

find CRC and verify Received Message.

Polynomial is 11011 which is of Degree 4

\therefore CRC bits = 4

Padded Message = 101 001 00 00

Division 11011 | 101 001 00 00
 100 11

$$\begin{array}{r}
 011\ 11 \\
 110\ 11 \\
 \hline
 101\ 00 \\
 110\ 11 \\
 \hline
 101\ 00 \\
 110\ 11 \\
 \hline
 001000
 \end{array}$$

Remainder = 1010
 \therefore CRC = 1010

Transmitted Message = 101 001 1010

Verification of Received Message

Division 11011 | 101 001 1001

$$\begin{array}{r}
 11011 \\
 \hline
 01111 \\
 11011 \\
 \hline
 10100 \\
 11011 \\
 \hline
 01111 \\
 11011 \\
 \hline
 10100 \\
 11011 \\
 \hline
 0111
 \end{array}$$

Remainder = 0111

\therefore Remainder is Non zero
 Error is Detected.

Q5

Data = 1011

Generate Hamming code

Bit position	1	2	3	4	5	6	7
Type	P_1	P_2	0	P_3	0	0	0
Value	0	0	1	0	0	1	1

$$P_1 = P_1 \oplus 1 \oplus 0 \oplus 1 = 0$$

$$\therefore P_1 = 0$$

$$P_2 = P_2 \oplus 1 \oplus 1 \oplus 1 = 0$$

$$\therefore P_2 = 1$$

$$P_3 = P_3 \oplus 0 \oplus 1 \oplus 1 = 0$$

$$P_3 = 0$$

∴ Transmitted Message is

0110011

Q5

Received Message 101110

Bit position	1	2	3	4	5	6	7
Type	P ₁	P ₂	D	P ₄	D	D	D
Received	1	0	1	1	1	0	1

To verify Errors and Even Parity

$$P_1 = P_1 \oplus D_3 \oplus D_5 \oplus D_7 = 0$$

→ No Errors

$$P_2 = P_2 \oplus D_3 \oplus D_6 \oplus D_7 = 0$$

→ No Errors

$$P_3 = P_1 \oplus D_5 \oplus D_6 \oplus D_7 = 1$$

→ Errors Detected

Error position is P₁ P₂ P₃ in Binary

∴ Error at $(100)_2$ at positions

∴ correct Message by flipping bit 5

Correct Message = 1010 101

Q6

Given Data 11001

Using Even Parity

Bit Position: 1 2 3 4 5 6 7 8 9

Type: $P_1 \ P_2 \ D_3 \ P_4 \ D_5 \ D_6 \ D_7 \ D_8 \ D_9$

Data: 1 1 0 0 1 1

$$P_1 = P_1 \oplus D_2 \oplus D_5 \oplus D_7 = 0$$

$\therefore P_1 = 0$

$$P_2 = P_2 \oplus D_3 \oplus D_4 \oplus D_5 = 0$$

$P_2 = 1$

$$P_4 = D_2 \oplus D_3 \oplus D_4 = 0$$

$P_4 = 0$

$$P_8 = D_5$$

$$P_8 = 1$$

-- Transmitted Message is
11100 1101

Assignment 4

Theory

1. IP Addressing Basics IPv4 and IPv6 Addressing

- IPv4 (Internet Protocol Version 4):
 - Uses a 32-bit addressing scheme
 - allows around 4.3 billion unique addresses
 - written in decimal format, separated by dots (For Ex: 192.168.1.1)
- IPv6 (Internet Protocol Version 6):
 - Uses a 128-bit addressing scheme
 - written in hexadecimal format, separated by colons (For Ex: 1011:09pa:00a2:0000:0000:9fe3:9102:4332). Public vs. Private IP Addresses
- Public IP Addresses:
 - Routable on the internet and assigned by ISPs (For Ex 8.8.8).
- Private IP Addresses:
 - Used within local networks and not routable on the internet. Their Address Ranges are:
Class A: 10.0.0.0 – 10.255.255.255
Class B: 172.16.0.0 – 172.31.255.255
Class C: 192.168.0.0 – 192.168.255.255
 - Subnet Masks
- A subnet mask defines the network and host portions of an IP address.
- For Ex: 255.255.0.0 indicates that the first two octets represent the network portion, and the last two octets represent hosts.

2. Subnetting CIDR Notation and Its Role in Subnetting:

- CIDR: Classless Inter-Domain Routing
- A method to allocate IP addresses efficiently by eliminating the rigid class-based addressing system.
- Expressed using a suffix (For Ex: 192.168.1.0/24 means 24 bits for the network, 8 bits for hosts).
- Subnetting allows networks to be divided into smaller subnetworks, improving security and reducing IP wastage.
- Example: Splitting 192.168.1.0/24 into four subnets:
 - 192.168.1.0/26
 - 192.168.1.64/26
 - 192.168.1.128/26
 - 192.168.1.192/26

3. Supernetting Definition and Advantages:

- Supernetting: The process of combining multiple smaller networks into a larger one by modifying the subnet mask.
- Reduces the number of routing table entries, improving efficiency.
- Helps ISPs aggregate multiple customer networks. Real-World Applications:
- Used in ISP route aggregation to reduce the number of advertised routes.
- Helps in enterprise network management by consolidating multiple subnets.
- Example: Combining four /26 networks (192.168.1.0/26, 192.168.1.64/26, 192.168.1.128/26, 192.168.1.192/26) into a single /24 network (192.168.1.0/24). ## Practical

1. **Subnetting a Corporate Network** Departments:
 1. HR: 50 Hosts
 2. IT: 100 Hosts
 3. Finance: 30 Hosts Subnets Sizes Subnets Are Sized by Powers of 2. Therefore nearest power of 2 for each department. Subnets are determined by the formula: Subnet = $32 - \text{Number of Bits}$
 4. HR: 64 [6 Bits, /26 Subnet]
 5. IT: 128 [7 Bits, /25 Subnet]
 6. Finance: 32 [5 Bits, /27 Subnet] Subnet Assignment: (Allocating Sequentially We Have)
2. IT (/25)
 - Network Address: 192.168.1.0/25
 - Broadcast Address: 192.168.1.127
 - Subnet Mask: 255.255.255.128
 - Usable IPs: 192.168.1.1 – 192.168.1.126
2. HR (/26)
 - Network Address: 192.168.1.128/26
 - Broadcast Address: 192.168.1.191
 - Subnet Mask: 255.255.255.192
 - Usable IPs: 192.168.1.129 – 192.168.1.190
3. Finance (/27)
 - Network Address: 192.168.1.192/27
 - Broadcast Address: 192.168.1.223
 - Subnet Mask: 255.255.255.224
 - Usable IPs: 192.168.1.193 – 192.168.1.222
2. Subnetting an ISP Network Supernetting combines multiple contiguous networks into a single network. This reduces the number of routing table entries, Thus improving efficiency. Given Networks:
 - 192.168.10.0/24
 - 192.168.11.0/24
 - 192.168.12.0/24
 - 192.168.13.0/24 Finding Common Bits in the Network Address Since the subnet is /24, the third octet changes and is relevant. Thus, we find common bits in the third octet. Given Values in 8 Bits:
 - 10 : 00001010
 - 11 : 00001011
 - 12 : 00001100
 - 13 : 00001101 We can See here, that only the last 6 bits remain constant. thus, to include all 4 networks we need a /22 Mask. ($16 + 6 = 22$) 16 for the first 2 octets, 6 for the 3rd one New Network addresses Since the new Subnet mask is /22, The Network Address 192.168.10.0/22 would cover all the four networks. Advantages

- For the router, it reduces the number of entries in the routing table. (from 4 /24 Entries to 1 /22)
 - The lower entry count improves efficiency and simplifies the network Management for Larger Networks.
3. Subnetting a University Network Departments:
1. Engineering: 2,000 hosts
 2. Medical: 1,500 hosts
 3. Management: 1,000 hosts
 4. Library: 500 hosts
 5. Admin: 300 hosts Subnets Sizes Subnets Are Sized by Powers of 2. Therefore nearest power of 2 for each department. Subnets are determined by the formula: Number of Bits are the nearest larger power of 2. Subnet = $32 - \text{Number of Bits}$
 6. Engineering: 2048 [11 Bits, /21 Subnet]
 7. Medical: 2048 [11 Bits, /21 Subnet]
 8. Management: 1024 [10 Bits, /22 Subnet]
 9. Library: 512 [9 Bits, /23 Subnet]
 10. Admin: 512 [9 Bits, /23 Subnet] Subnet Assignment: (Allocating Sequentially We Have)
 11. Engineering (/21)
 - Network Address: 172.16.0.0/21
 - Broadcast Address: 172.16.7.255
 - Usable IPs: 172.16.0.1 – 172.16.7.254
 - Subnet Mask: 255.255.248.0
 12. Medical (/21)
 - Network Address: 172.16.8.0/21
 - Broadcast Address: 172.16.15.255
 - Usable IPs: 172.16.8.1 – 172.16.15.254
 - Subnet Mask: 255.255.248.0
 13. Management (/22)
 - Network Address: 172.16.16.0/22
 - Broadcast Address: 172.16.19.255
 - Usable IPs: 172.16.16.1 – 172.16.19.254
 - Subnet Mask: 255.255.252.0
 14. Library (/23)
 - Network Address: 172.16.20.0/23
 - Broadcast Address: 172.16.21.255
 - Usable IPs: 172.16.20.1 – 172.16.21.254
 - Subnet Mask: 255.255.254.0
 15. Admin (/23)
 - Network Address: 172.16.22.0/23

- Broadcast Address: 172.16.23.255
- Usable IPs: 172.16.22.1 – 172.16.23.254
- Subnet Mask: 255.255.254.0

Assignment 5

SAP: 500124827

Name: Kshitij Chandrakar

Batch: 5

ifconfig: Deprecated, would not run on my machine but displays information about the connected networks. **ip addr:** Displays information about the connected networks.

```
asus@debian-asusrog:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host noprefixroute
            valid_lft forever preferred_lft forever
2: eno2: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default
    link/ether 04:d4:c4:eb:74:97 brd ff:ff:ff:ff:ff:ff
        altname enp3s0
3: wlo1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 28:7f:cf:e5:6b:e5 brd ff:ff:ff:ff:ff:ff
        altname wlp0s20f3
        inet 192.168.0.102/24 brd 192.168.0.255 scope global dynamic noprefixroute wlo1
            valid_lft 6538sec preferred_lft 6538sec
        inet6 fe80::4d56:ec7:7abb:abe4/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
4: CloudflareWARP: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1280 qdisc mq state UP
    link/none
        inet 172.16.0.2/32 scope global CloudflareWARP
            valid_lft forever preferred_lft forever
        inet6 2606:4700:110:8865:6078:763e:be97:f2f3/128 scope global
            valid_lft forever preferred_lft forever
        inet6 fe80::d28f:8bef:b8ff:28f1/64 scope link stable-privacy
            valid_lft forever preferred_lft forever
```

hostname: displays the hostname of the machine.

```
asus@debian-asusrog:~$ hostname
debian-asusrog
asus@debian-asusrog:~$
```

ping: pings the given address

```
asus@debian-asusrog:~$ ping google.com
PING google.com(bom12s08-in-x0e.1e100.net (2404:6800:4009:813::200e)) 56 data bytes
64 bytes from bom12s08-in-x0e.1e100.net (2404:6800:4009:813::200e): icmp_seq=1 ttl=59
64 bytes from bom12s08-in-x0e.1e100.net (2404:6800:4009:813::200e): icmp_seq=2 ttl=59
64 bytes from bom12s08-in-x0e.1e100.net (2404:6800:4009:813::200e): icmp_seq=3 ttl=59
64 bytes from bom12s08-in-x0e.1e100.net (2404:6800:4009:813::200e): icmp_seq=4 ttl=59
^C
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 49.602/107.369/167.864/49.303 ms
```

ip route: Shows the route a given packet takes.

```
asus@debian-asusrog:~$ ip route
default via 192.168.0.1 dev wlo1 proto dhcp src 192.168.0.102 metric 600
169.254.0.0/16 dev wlo1 scope link metric 1000
192.168.0.0/24 dev wlo1 proto kernel scope link src 192.168.0.102 metric 600
```

netstat: Print network connections, routing tables, interface statistics, masquerade connections, and multicast memberships

```
asus@debian-asusrog:~% netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp      0      0 debian-asusrog:38442    lb-140-82-112-25-:https ESTABLISHED
tcp      0      0 debian-asusrog:51868     ec2-52-89-11-84.u:https ESTABLISHED
tcp      0      25 debian-asusrog:48074     162.159.192.1:https     LAST_ACK
tcp      0      0 debian-asusrog:35158     server-18-67-195-:https ESTABLISHED
tcp      0      0 debian-asusrog:59318     93.243.107.34.bc:https TIME_WAIT
tcp      0      0 debian-asusrog:54006     140.227.186.35.bc:https ESTABLISHED
tcp      0      0 debian-asusrog:45268     a23-38-59-250.depl:http ESTABLISHED
```

traceroute: Shows the route a given packet takes.

```
asus@debian-asusrog:~$ traceroute google.com
traceroute to google.com (142.250.192.110), 30 hops max, 60 byte packets
```

```
1 * *
2 * *
3 * *
4 * *
5 * *
6 * *
7 * *
8 * *
9 * *
10 * *
11 * *
12 * *
13 * *
14 * *
15 * *
16 * *
17 * *
18 * *
19 * *
20 * *
21 * *
22 * *
23 * *
24 * *
25 * *
26 * *
27 * *
28 * *
29 * *
30 * *
```

nmcli: cli tool to manage the network.

```
wlo1: connected to 1st Floor TPlink
    "Intel Cannon Lake PCH CNVi"
    wifi (iwlwifi), 28:7F:CF:E5:6B:E5, hw, mtu 1500
    ip4 default
    inet4 192.168.0.102/24
    route4 192.168.0.0/24 metric 600
    route4 default via 192.168.0.1 metric 600
    route4 169.254.0.0/16 metric 1000
    inet6 fe80::4d56:ec7:7abb:abe4/64
    route6 fe80::/64 metric 1024

CloudflareWARP: connected (externally) to CloudflareWARP
    "CloudflareWARP"
    tun, sw, mtu 1280
    inet4 172.16.0.2/32
    route4 32.0.0.0/3 metric 0
    route4 64.0.0.0/3 metric 0
    route4 128.0.0.0/3 metric 0
    route4 16.0.0.0/4 metric 0
    route4 112.0.0.0/4 metric 0
    route4 176.0.0.0/4 metric 0
    route4 208.0.0.0/4 metric 0
    route4 0.0.0.0/5 metric 0
    route4 104.0.0.0/5 metric 0
    route4 200.0.0.0/5 metric 0
    route4 232.0.0.0/5 metric 0
    route4 12.0.0.0/6 metric 0
    route4 96.0.0.0/6 metric 0
    route4 164.0.0.0/6 metric 0
    route4 196.0.0.0/6 metric 0
    route4 228.0.0.0/6 metric 0
    route4 8.0.0.0/7 metric 0
    route4 102.0.0.0/7 metric 0
    route4 160.0.0.0/7 metric 0
    route4 170.0.0.0/7 metric 0
    route4 174.0.0.0/7 metric 0
    route4 194.0.0.0/7 metric 0
    route4 226.0.0.0/7 metric 0
    route4 11.0.0.0/8 metric 0
    route4 101.0.0.0/8 metric 0
    route4 163.0.0.0/8 metric 0
    route4 168.0.0.0/8 metric 0
    route4 173.0.0.0/8 metric 0
    route4 193.0.0.0/8 metric 0
    route4 225.0.0.0/8 metric 0
    route4 100.128.0.0/9 metric 0
    route4 162.0.0.0/9 metric 0
    route4 169.0.0.0/9 metric 0
    route4 172.128.0.0/9 metric 0
    route4 224.128.0.0/9 metric 0
    route4 100.0.0.0/10 metric 0
    route4 162.192.0.0/10 metric 0
    route4 169.128.0.0/10 metric 0
    route4 172.64.0.0/10 metric 0
```

lines 1-54

dhclient: a tool to manage network interfaces using DHCP.

```
asus@debian-asusrog:~$ sudo dhclient
RTNETLINK answers: File exists
asus@debian-asusrog:~$ █
```

ssh: used to ssh into other machines using the ssh protocol

```
asus@debian-asusrog:~$ ssh
usage: ssh [-46AaCfGgKkMNnqsTtVvXxYy] [-B bind_interface]
           [-b bind_address] [-c cipher_spec] [-D [bind_address:]port]
           [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
           [-i identity_file] [-J [user@]host[:port]] [-L address]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-Q query_option] [-R address] [-S ctl_path] [-W host:port]
           [-w local_tun[:remote_tun]] destination [command [argument ...]]
asus@debian-asusrog:~$ █
```

scp: Secure File Copy over SSH in a given network.

```
asus@debian-asusrog:~$ scp
usage: scp [-346ABC0pqRrsTv] [-c cipher] [-D sftp_server_path] [-F ssh_config]
           [-i identity_file] [-J destination] [-l limit] [-o ssh_option]
           [-P port] [-S program] [-X sftp_option] source ... target
```

EXPERIMENT - 6

Theoretical Understanding

1) Define Network Topology.

→ Network topology refers to the physical or logical arrangement of network devices and connections. It defines how computers, cables, switches, and other devices are interconnected to facilitate communication and data exchange.

2) Advantages & disadvantages of -

- Bus Topology

Advantages -

- Cost-effective as it requires less cabling.
- Easy to set up and extend.
- Works well for small networks.

Disadvantages -

- A single point of failure (main cable) can bring down the entire network.
- Performance degrades as more devices are added.
- Data collisions can occur, reducing efficiency.

- Ring topology

Advantages -

- No data collisions due to the unidirectional or bidirectional data flow.
- Predictable data transmission due to sequential passing of data.

Disadvantages -

- Failure of a single node or cable can disrupt the entire network.
- Troubleshooting is difficult due to the dependency on all devices in the ring.
- Adding or removing devices requires temporarily shutting down the network.
- Star topology

Advantages -

- High fault tolerance; failure of one device does not affect the entire network.
- Easy to manage, scale, and troubleshoot.
- Centralized control provides better performance.

Disadvantages -

- Expensive due to the requirement of a central switch or hub.
- If the central hub/switch fails, the entire network is affected.

Mesh topologyAdvantages -

- High redundancy; failure of one connection does not disrupt communication.

- Offers excellent reliability & security.
- Data transmission is faster due to multiple paths.

Disadvantages -

- expensive due to expensive cabling and hardware requirements.
- Complex installation and maintenance.

3) Which topology is most fault-tolerant and why?

→ Mesh topology is the most fault-tolerant because it has multiple redundant paths for data transmission. Even if one link or device fails, alternative paths ensure uninterrupted communication.

4) Which topology is cost-effective & why?

→ Bus topology is the most cost-effective because it requires minimal cabling and no expensive central devices, making it suitable for small networks.

5) Compare Mesh and Star Topology
in terms of Scalability & reliability.

→ Scalability -

Mesh topology is difficult to scale
due to high wiring and configuration
complexity.

Star topology is easier to scale by
adding more devices to the central
switch.

Reliability -

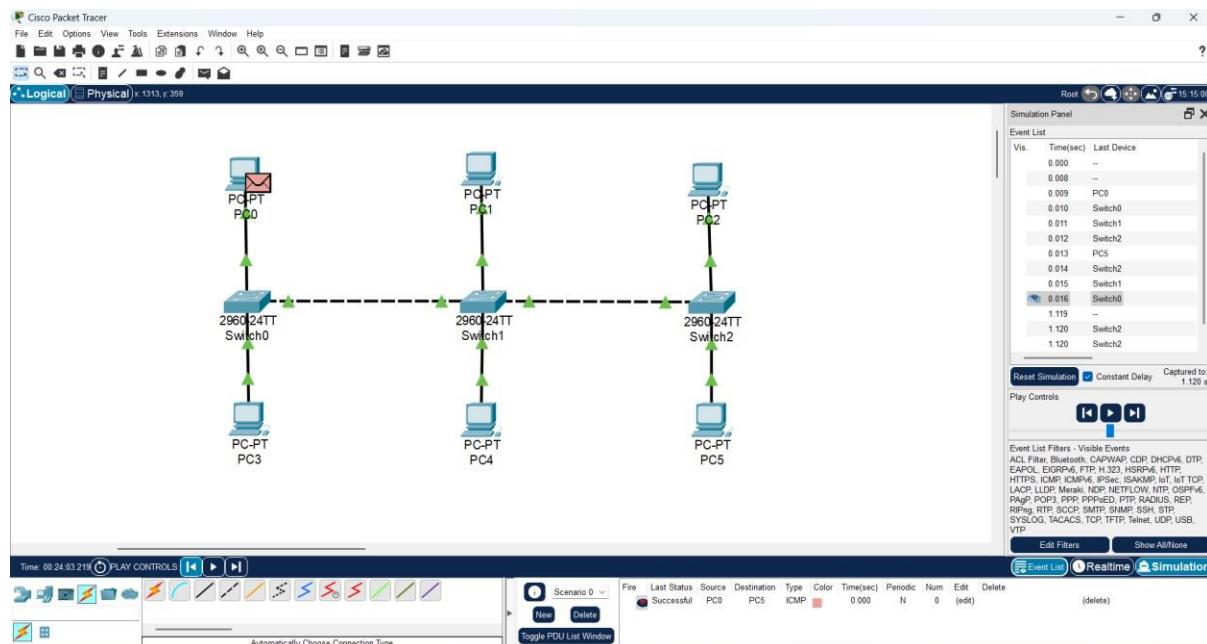
Mesh topology
Highly reliable due to multiple
redundant paths.

Star topology is also reliable
but dependent on the central
hub/switch, which is a single
point of failure.

Task 2: Network Topology Design in Cisco Packet Tracer

Design and simulate the following network topologies using Cisco Packet Tracer:

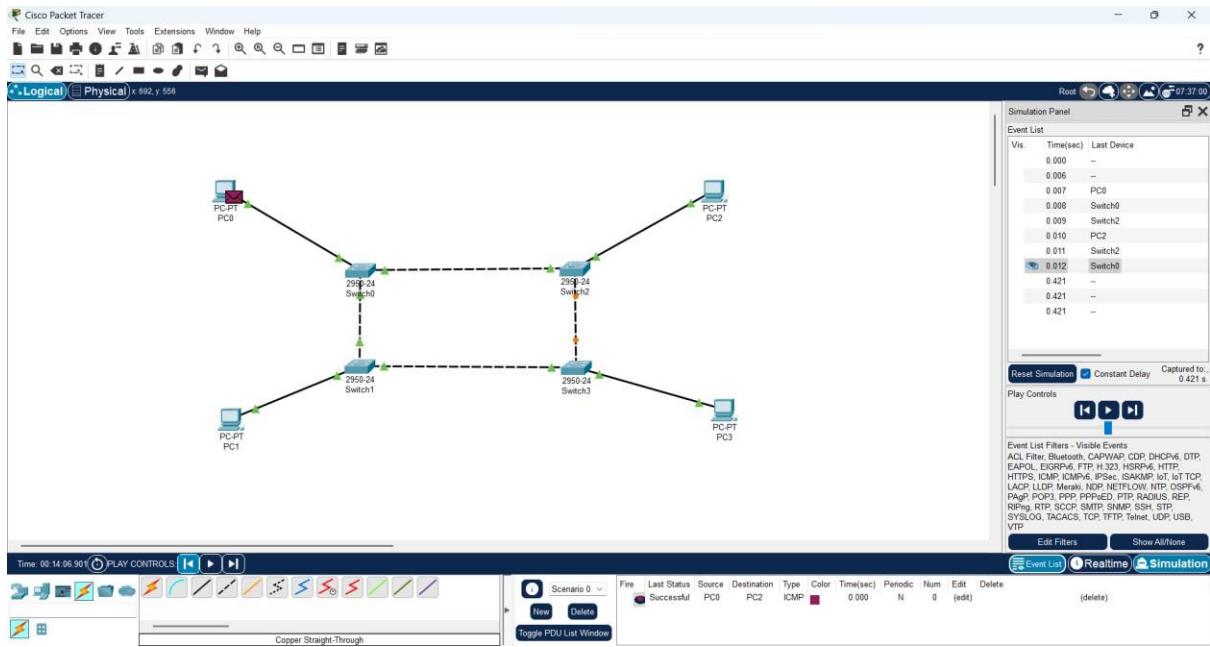
- Bus Topology (Using a single backbone cable)



Data Transmission Flow:

- Data travels along the backbone in both directions.
- Only the intended recipient processes the data.
- If two devices send data simultaneously, a collision occurs, requiring a retransmission.

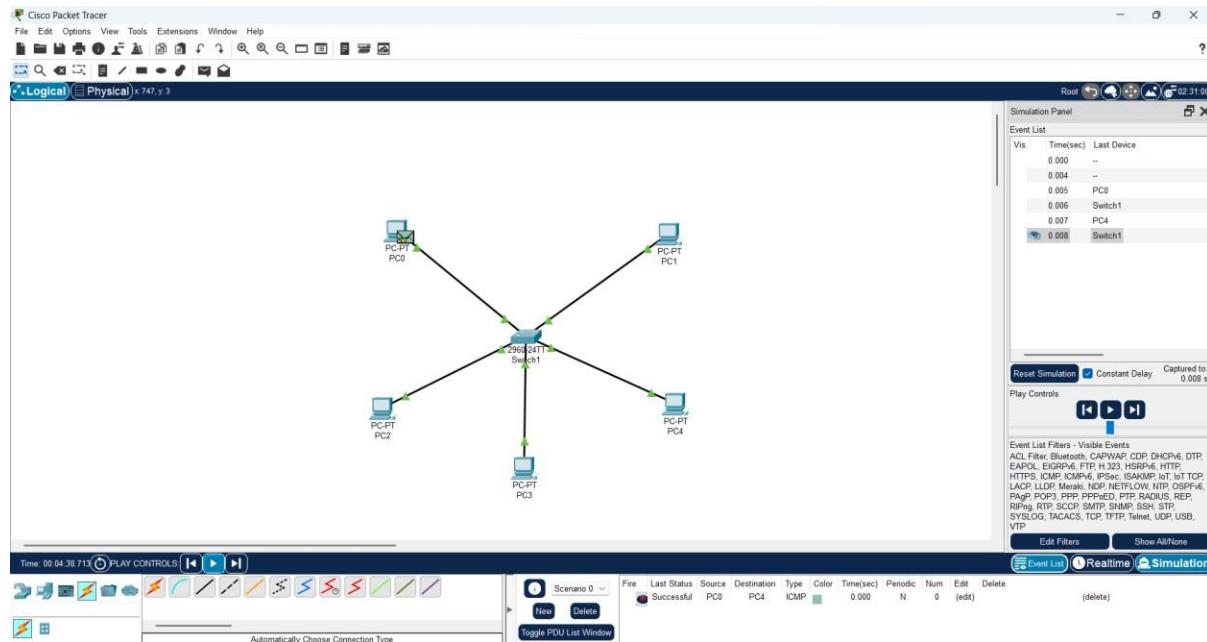
- Ring Topology (Connecting all devices in a loop)



Data Transmission Flow:

- Data moves in one direction (unidirectional) or both directions (bidirectional in modern implementations).
- A token-passing mechanism is used to prevent collisions.

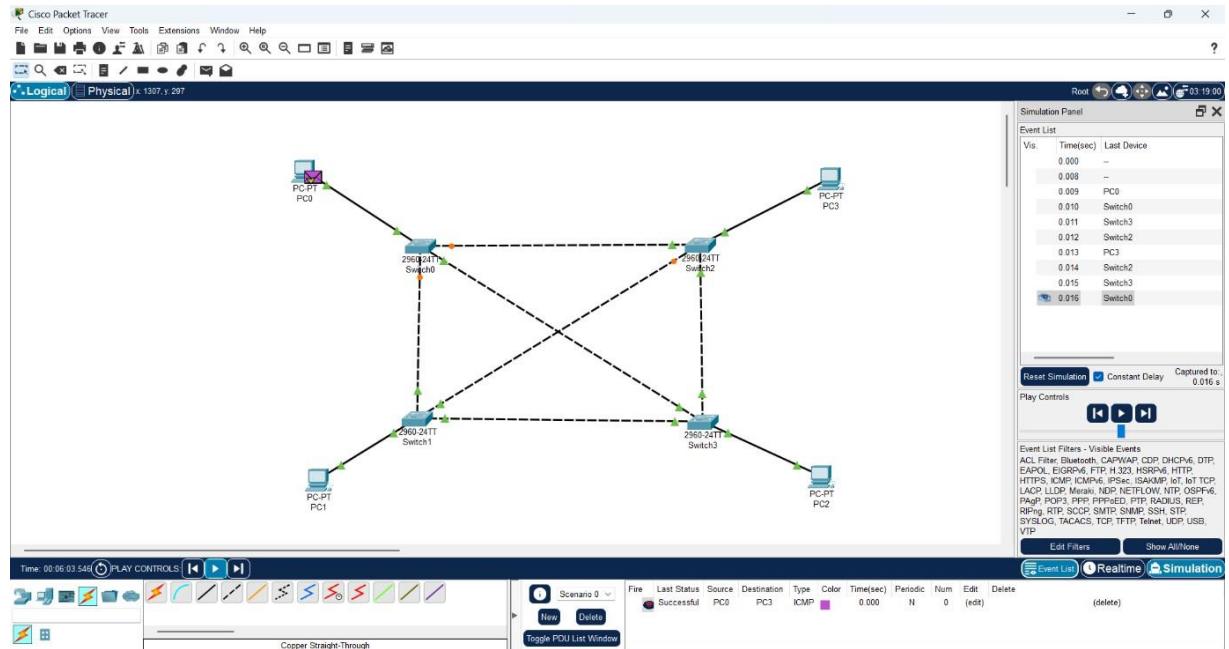
- Star Topology (Using a central switch or hub)



Data Transmission Flow:

- Data travels from the sender to the switch/hub, then forwarded to the recipient.
- Switches use MAC addresses to forward packets, while hubs broadcast data to all devices.

- Mesh Topology (Full or partial, as per requirement)



Data Transmission Flow:

- Data follows multiple paths, ensuring redundancy and fault tolerance.
- Routing protocols like OSPF or RIP can be used for dynamic path selection.

Assignment 7

Name: Kshitij Chandrakar SAP: 500124827

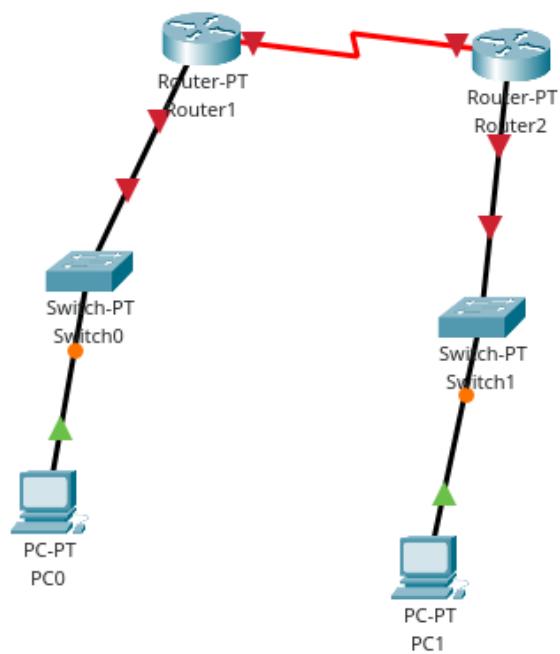
Theory Section

ASSIGNMENT-7	
1. Explain the difference between static routing and dynamic routing.	→ Static Routing (i) Manually configured by a network administrator. (ii) No extra CPU or bandwidth usage for route updates. (iii) Small, stable networks where routes don't change often. (iv) Does not adapt to network changes unless manually updated.
	Dynamic Routing (i) Automatically updates routing tables using routing protocols. (ii) Consumes CPU and bandwidth due to routing protocol updates. (iii) Large, complex networks with frequent topology changes. (iv) Adapts automatically to network changes.
Q. Why is crossover cable used for router-to-router connections, while a straight-through cable is used for PC-to-switch connections?	→ Crossover cable: → used when connecting two similar devices (e.g. router-to-router, switch-to-switch) → It swaps the transmission (TX) and reception (RX) wires so that the sending end of one device connects to the receiving end of the other.

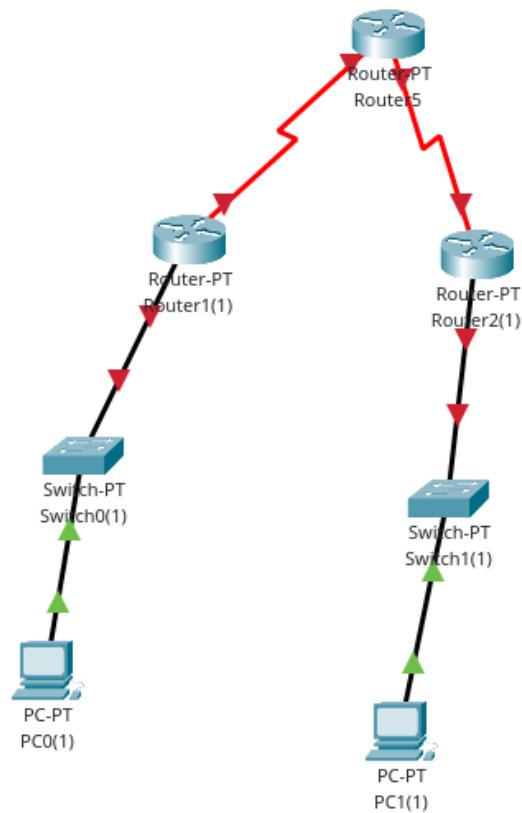
- Straight - Through cable:-
 - used when connecting different devices like PC to switch, PC to Router etc.
 - ensures proper communication without the need for signal crossover.
3. Describe the role of a default gateway in inter-network communication.
- A default gateway is a router that acts as an access point for a local network to communicate with external such as internet.
 - when a device wants to send data to be destination outside its subnet, it forwards the packet to the default gateway.
 - It then determines the best route to forward the packet to its final destination.

Practical Section

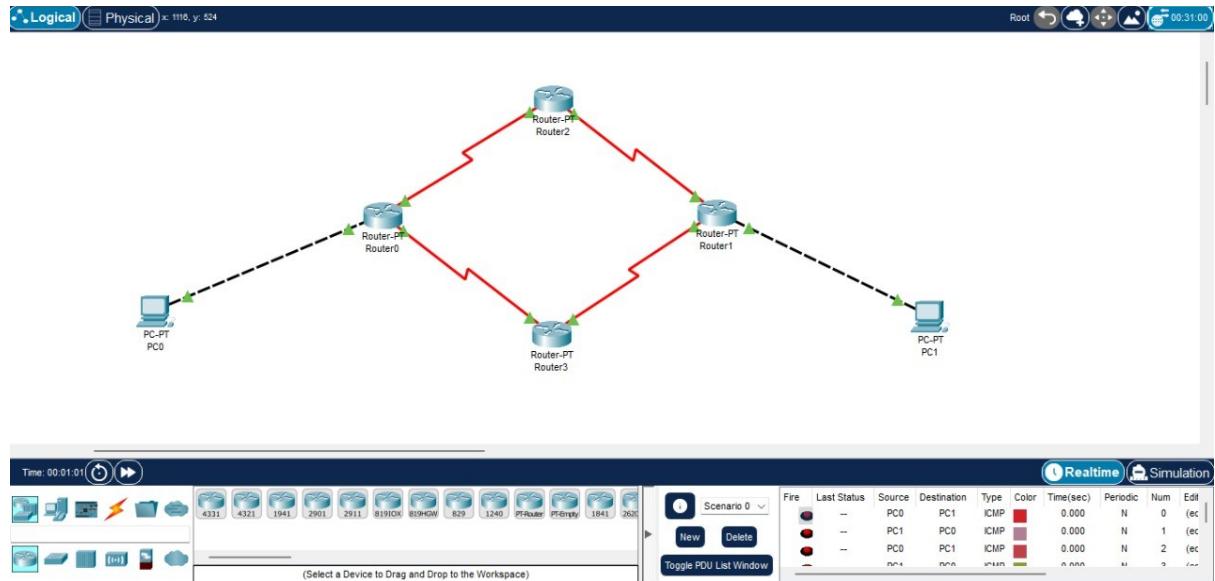
Question 1



Question 2



Distance Vector Routing (DVR) protocol using Packet Tracker



IP CONFIGURATION

PC0

Physical Config Desktop Programming Attributes

IP Configuration

Interface: FastEthernet0

IP Configuration

DHCP Static IP Address: 192.168.1.1 Subnet Mask: 255.255.255.0 Default Gateway: 192.168.1.10 DNS Server: 0.0.0.0

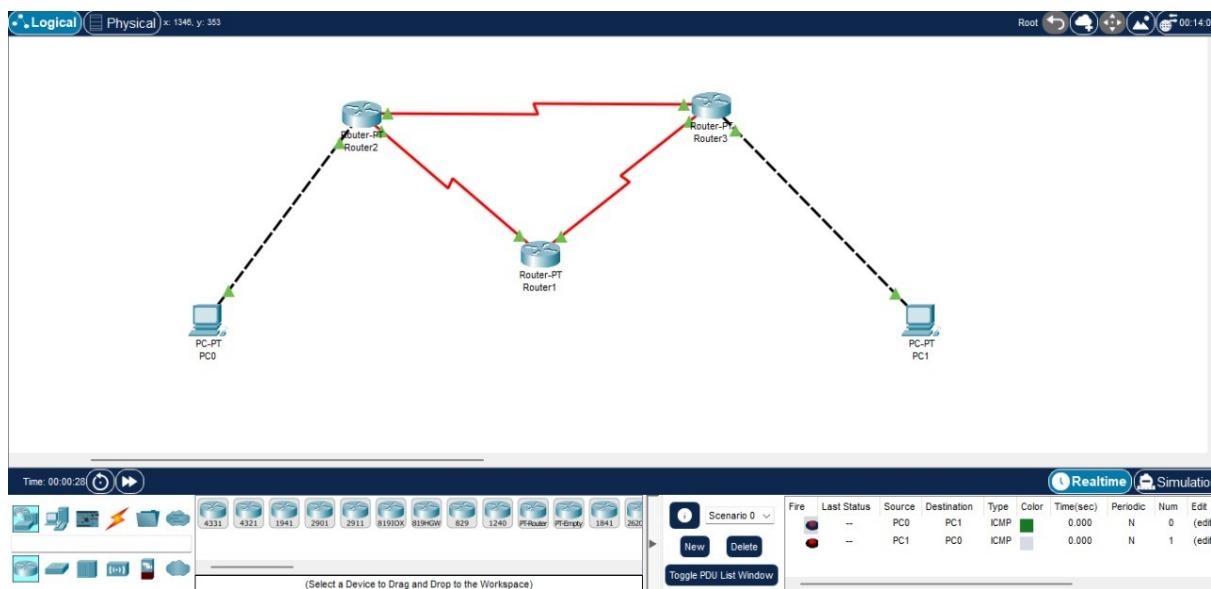
IPv6 Configuration

Automatic Static IPv6 Address: / Link Local Address: FE80::201:63FF:FE74:BA37 Default Gateway: DNS Server:

802.1X

Use 802.1X Security Authentication: MD5 Username: Password:

X



IP CONFIGURATION

PC0

Physical Config **Desktop** Programming Attributes

IP Configuration

Interface: FastEthernet0

DHCP Static

IPv4 Address: 192.168.1.1

Subnet Mask: 255.255.255.0

Default Gateway: 192.168.1.10

DNS Server: 0.0.0.0

IPv6 Configuration

Automatic Static

IPv6 Address: /

Link Local Address: FE80::204:9AFF:FE66:BAB

Default Gateway:

DNS Server:

802.1X

Use 802.1X Security

Authentication: MD5

Username:

Password:

THEORY QUESTIONS

EXPERIMENT - 8

Date _____
Page No. _____

1. Explain the working of the distance vector routing protocol?

→ It is a routing algorithm used in computer networks to determine the best path for data packets. Each router maintains a routing table that contains the distance (cost) and direction (next-hop) to reach all network destination. Routers periodically exchange their routing with connected neighbours. They update their own table using the Bellman-Ford Algorithm, selecting the shortest path to each destination.

2. What is the significance of periodic updates in DVR?

→ Periodic updates ensure that all routers are aware of network topology changes. These updates help routers detect between broken links, new routes, or changes in path costs. However, excessive updates can lead to high bandwidth consumption and slow convergence.

3. Discuss the key differences between distance vector and link-state Routing?

→ Distance vector routing

- i) Entry routing table shared with neighbours.
- ii) Slower due to periodic updates.
- iii) Computational overhead is low.
- iv) It is less scalable due to high bandwidth usage.

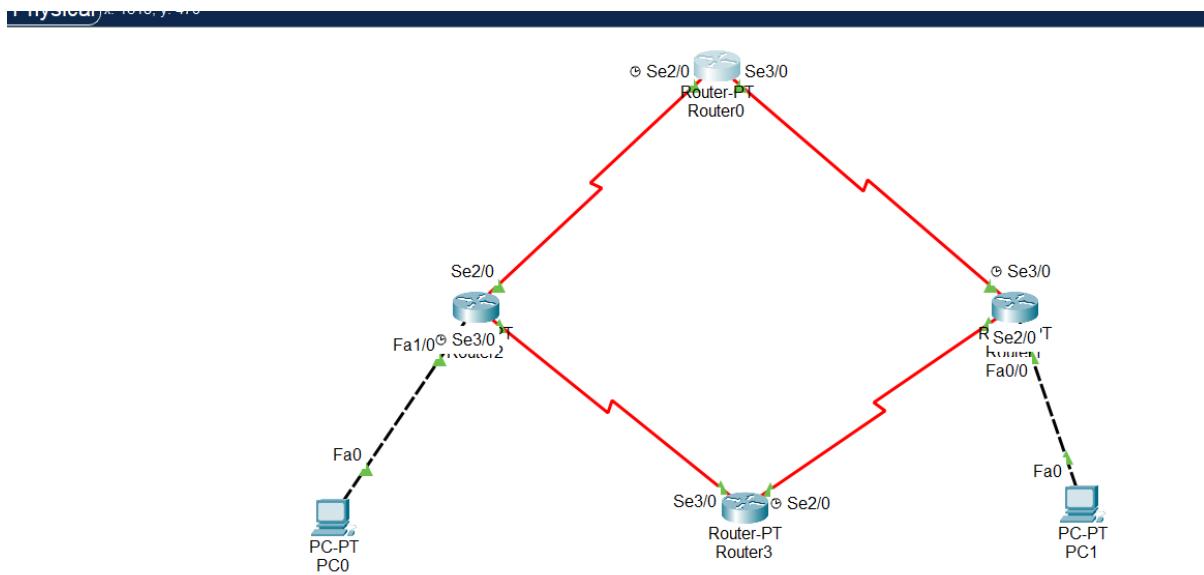
link-state routing

- i) Only link-state information is shared.
- ii) Faster due to triggered updates.
- iii) Computational overhead is low.
- iv) It is more scalable for large networks.

Q. What are the limitations of the DSR protocol, and how can they be mitigated?

- i) It takes time to adapt to network changes, causing delays.
- ii) Incorrect routes may loop indefinitely.
- iii) Periodic updates consume excessive network resources.
- iv) Becomes inefficient in large networks due to frequency updates.

Assignment 9: Link-State Vector Routing Protocol Using Cisco Packet Tracer



Router0

Physical Config **CLI** Attributes

IOS Command Line Interface

```
32K bytes of non-volatile configuration memory.  
63488K bytes of ATA CompactFlash (Read/Write)

--- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: no

Press RETURN to get started!

Router> enable
Router# configure
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router ospf 1
OSPF process 1 cannot start. There must be at least one "up" IP interface
Router(config-router)# network

Router con0 is now available

Press RETURN to get started.
```

Router con0 is now available

Press RETURN to get started.

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to down  
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to down  
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up  
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to up
```

Router1

Physical Config **CLI** Attributes

IOS Command Line Interface

```
Router(config-if)#exit
Router(config)#router rip
Router(config-router)#
Router(config-router)#end
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface FastEthernet1/0
Router(config-if)#
%SYS-5-CONFIG_I: Configured from console by console

Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address
% Incomplete command.
Router(config-if)#ip address
% Incomplete command.
Router(config-if)Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#interface FastEthernet0/0
Router(config-if) Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#interface FastEthernet0/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial3/0
Router(config-if)Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#interface Serial3/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial3/0
Router(config-if)#no ip address
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
```

```
Router(config)#interface Serial3/0
Router(config-if)#Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#interface Serial3/0
Router(config-if)#ip address 30.30.30.1 255.0.0.0
Router(config-if)#ip address 30.30.30.1 255.0.0.0
Router(config-if)#no shutdown
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#no ip address
Router(config-if)#ip address 40.40.40.1 255.0.0.0
Router(config-if)#ip address 40.40.40.1 255.0.0.0
Router(config-if)#
Router(config-if)#exit
Router(config)#router rip
Router(config-router)#
Router(config-router)#end
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface Serial2/0
Router(config-if)#
%SYS-5-CONFIG_I: Configured from console by console

Router(config-if)#exit
Router(config)#router rip
Router(config-router)#network 50.0.0.0
Router(config-router)#
Router(config-router)#end
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface Serial2/0
Router(config-if)#
%SYS-5-CONFIG_I: Configured from console by console

Router(config-if)#exit
Router(config)#interface Serial3/0
Router(config-if)#ip address 30.30.30.1 255.0.0.0
Router(config-if)#
Router(config-if)#exit
```

Router2

Physical Config **CLI** Attributes

IOS Command Line Interface

```
Press RETURN to get started!

Router>enable
Router#configure
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router ospf 1
OSPF process 1 cannot start. There must be at least one "up" IP interface
Router(config-router)#network 20.0.0.0 0.255.255.255 area 0
Router(config-router)#network 30.0.0.0 0.255.255.255 area 0
Router(config-router)#network 50.0.0.0 0.255.255.255 area 0
Router(config-router)#no shut down
^
% Invalid input detected at '^' marker.

Router(config-router)#no shutdown
^
% Invalid input detected at '^' marker.

Router(config-router)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#ip address 20.20.1 255.0.0.0
^
% Invalid input detected at '^' marker.

Router(config-if)#ip address 20.20.20.1 255.0.0.0
Router(config-if)#
Router#
%SYS-5-CONFIG_I: Configured from console by console

Router#enable
Router#configure
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router ospf 1
Router(config-router)#network 20.0.0.0 0.255.255.255 area 0
Router(config-router)#network 30.0.0.0 0.255.255.255 area 0
Router(config-router)#network 50.0.0.0 0.255.255.255 area 0
Router(config-router)#exit
Router(config)#interface FastEthernet1/0
^
% Invalid input detected at '^' marker.
```

```
Router(config-if)#ip address 20.20.20.1 255.0.0.0
Router(config-if)#Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#interface FastEthernet1/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address 20.20.20.1 255.0.0.0
Router(config-if)#n0 shutdown
^
% Invalid input detected at '^' marker.

Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

Router(config-if)#exit
Router(config)#interfaceEnter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#interface FastEthernet0/0
Router(config-if)#interface serial3/0
Router(config-if)#ip address 30.30.30.1 255.0.0.0
Router(config-if)# interface serial2/0
Router(config-if)# ip address 50.50.50.1 255.0.0.0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface Serial2/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up
```

```
Router con0 is now available
```

Press RETURN to get started.

```
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#interface FastEthernet0/0
Router(config-if)#Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#interface FastEthernet0/0
Router(config-if)#no ip address
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial3/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#
Router(config-if)#exit
```

```
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial3/0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial3/0, changed state to up

Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet1/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to up
%IP-4-DUPADDR: Duplicate address 20.20.20.1 on FastEthernet1/0, sourced by 0060.2F11.364E

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to up
```

Router con0 is now available

Press RETURN to get started.

Router3

Physical Config **CLI** Attributes

IOS Command Line Interface

```
System Bootstrap, Version 12.1(3r)T2, RELEASE SOFTWARE (fc1)
Copyright (c) 2000 by cisco Systems, Inc.
PT 1001 (PTSC2005) processor (revision 0x200) with 60416K/5120K bytes of memory

 Readonly ROMMON initialized

Self decompressing the image :
#####
# [OK]

      Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134-1706

Cisco Internetwork Operating System Software
IOS (tm) PT1000 Software (PT1000-I-M), Version 12.2(28), RELEASE SOFTWARE (fc5)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2005 by cisco Systems, Inc.
Compiled Wed 27-Apr-04 19:01 by miwang

PT 1001 (PTSC2005) processor (revision 0x200) with 60416K/5120K bytes of memory
.
Processor board ID PT0123 (0123)
PT2005 processor: part number 0, mask 01
Bridging software.
X.25 software, Version 3.0.0.
4 FastEthernet/IEEE 802.3 interface(s)
2 Low-speed serial(sync/async) network interface(s)
32K bytes of non-volatile configuration memory.
63488K bytes of ATA CompactFlash (Read/Write)

--- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: no
```

Press RETURN to get started!

```
Router>enable
Router#configure
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#router ospf 1
OSPF process 1 cannot start. There must be at least one "up" IP interface
Router(config-router)#network 40.0.0.0 0.255.255.255 area 0
Router(config-router)#network 50.0.0.0 0.255.255.255 area 0
Router(config-router)#exit
Router(config)#interface serial3/0
Router(config-if)#ip address 40.40.40.1 255.0.0.0
                                         ^
% Invalid input detected at '^' marker.

Router(config-if)#40.40.40.1 255.0.0.0
                                         ^
% Invalid input detected at '^' marker.

Router(config-if)#ip address 40.40.40.1 255.0.0.0
Router(config-if)#interface serial2/0
Router(config-if)#ip address 50.50.50.1 255.0.0.0
Router(config-if)#Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#interface Serial3/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config-if)#
Router(config-if)#interface Serial3/0
Router(config-if)#
Router(config-if)#exit
Router(config-if)#
Router(config-if)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#
Router(config-if)#exit
Router(config-if)#
Router(config-if)#interface Serial2/0
```

```
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial3/0
Router(config-if)#no shutdown
Router(config-if)#
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#no shutdown
Router(config-if)#
%LINK-5-CHANGED: Interface Serial2/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up

02:50:56: %OSPF-5-ADJCHG: Process 1, Nbr 30.30.30.1 on Serial2/0 from LOADING to FULL, Loading Done

%LINK-5-CHANGED: Interface Serial3/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to up
```

Router con0 is now available

Press RETURN to get started.

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to down
03:23:57: %OSPF-5-ADJCHG: Process 1, Nbr 30.30.30.1 on Serial2/0 from FULL to DOWN, Neighbor Down: Interface down or detached

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up

03:24:17: %OSPF-5-ADJCHG: Process 1, Nbr 30.30.30.1 on Serial2/0 from LOADING to FULL, Loading Done
```

```
Router(config)#router rip
Router(config-router)#network 30.0.0.0
Router(config-router)#
Router(config-router)#end
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#
%SYS-5-CONFIG_I: Configured from console by console

Router(config-if)#exit
Router(config)#router rip
Router(config-router)#network 40.0.0.0
Router(config-router)#
Router(config-router)#end
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#
%SYS-5-CONFIG_I: Configured from console by console

Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial3/0
Router(config-if)#
%LINK-5-CHANGED: Interface Serial2/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up

Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
02:51:17: %OSPF-5-ADJCHG: Process 1, Nbr 50.50.50.1 on Serial2/0 from LOADING to FULL, Loading Done

Router(config-if)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#

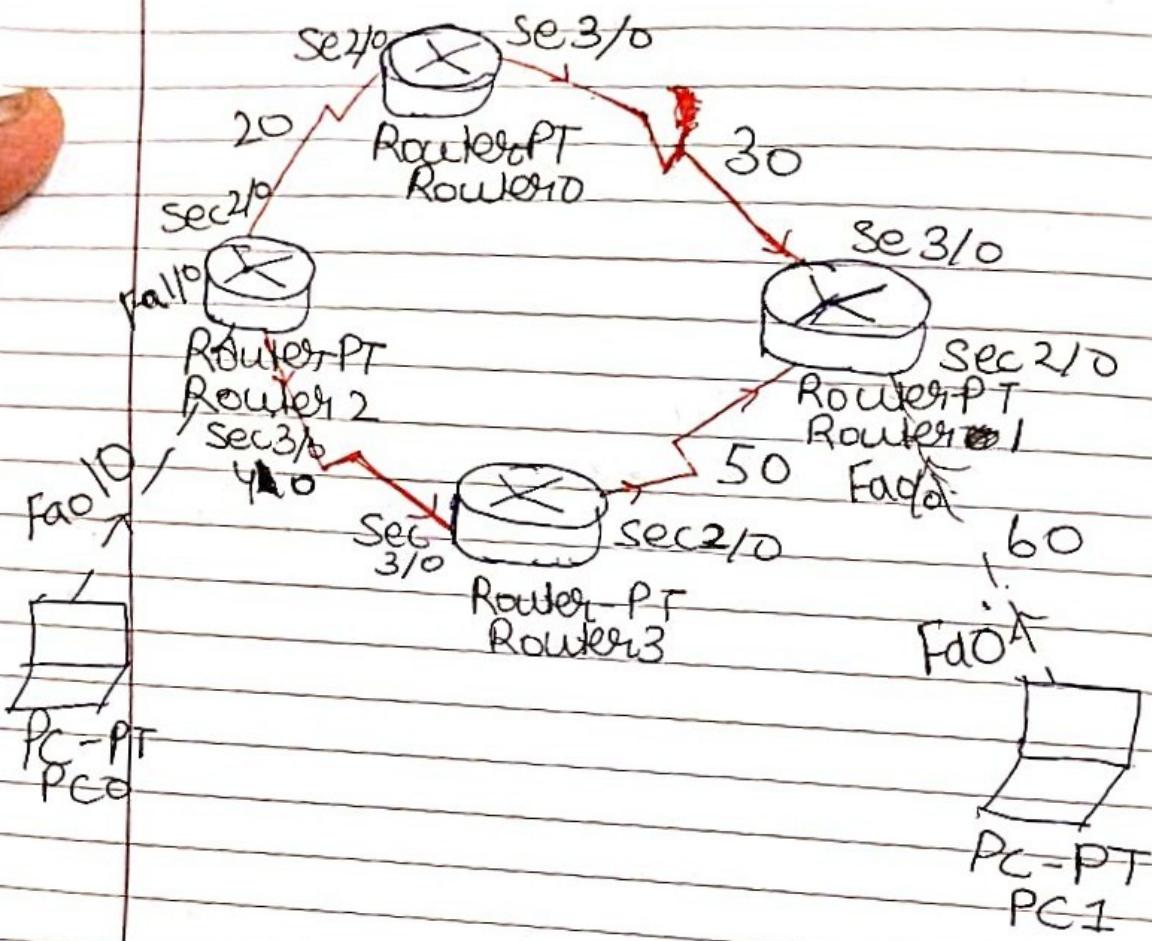
```

6/4/25

Date _____
Page _____

Assignment 9 : Link-state vector Routing Protocol using Cisco

Packet Tracer



```
Router(config)#interface FastEthernet0/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial3/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial3/0
Router(config-if)#
%LINK-5-CHANGED: Interface Serial3/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to up
```

Router con0 is now available

Press RETURN to get started.

Section 1: Nmap (Network Mapper)

1. What is Nmap, and what is its primary purpose in network monitoring?

- Nmap (Network Mapper) is an open-source network scanning tool used for network discovery and security auditing. Its primary purpose is to identify active devices, detect open ports, discover services running on a network and assess security vulnerabilities.

2. Explain the different types of scans available in Nmap. Provide examples.

- TCP Connect Scan (nmap -ST): Performs a full TCP three-way handshake to determine open ports.
- SYN Scan (nmap -sS): A stealthy scan that sends SYN packets and detects open ports without completing the handshake.
- UDP Scan (nmap -sU): Scans for open UDP ports, useful for detecting services like DNS and SNMP.

3. How does Nmap help in identifying open ports and services on a network?

- Nmap sends specially crafted packets to target systems and analyzes the responses to determine which ports are open, closed or filtered. It also uses detection (-SV) to identify the specific applications running on open ports.

4. What is OS detection in Nmap and how does it work?

- OS detection (-O flag) allows Nmap to determine the operating system of a target device by analyzing factors such as TCP/IP stack responses, packet TTL values and specific fingerprinting techniques.

5. Discuss the significance of the -A option in Nmap scanning.

- The (-A) option enables aggressive scanning which includes OS detection, version detection, script scanning, and traceroute. It provides comprehensive information about the target network but is more intrusive.

6. How can Nmap be used for security auditing and penetration testing?

- It identifies open ports and running services, exposing potential vulnerabilities.
- It detects outdated software versions with known exploits.

→ Uncovers misconfigurations and weak security controls.

7. what are the limitations of Nmap in network Monitoring?

- It cannot detect all vulnerabilities - relies on known signatures.
- Slow performance on large network or when scanning with deep analysis options.
- limited effectiveness against encrypted traffic.

Section 2: Wireshark

1. What is Wireshark, and how it used in network analysis?

- Wireshark is powerful open-source packet analyser used for real-time network traffic analysis. It captures and inspects data packets travelling over a network, helping network administrators and security professionals, detect intrusions.

2. Explain the different components of a captured packet in Wireshark.

- Frame Header - contains metadata such as timestamp, frame length and source/destination MAC addresses.

- Ethernet Header: Includes source and destination MAC addresses and the Ethertype field indicating the next protocol layer.
- IP Header: Shows the source and destination IP addresses, protocol type (TCP, UDP, ICMP) and packet fragmentation details.
- Transport layer: Display port numbers, sequence/acknowledgment numbers and flags.

3. How does Wireshark help in troubleshooting network issues?

- It identifies latency and network congestion by analyzing packet delays.
- Detects dropped packets, retransmissions and high error rates.
- Analyzes DNS, HTTP traffic to pinpoint misconfigurations.

4. Discuss the importance of filtering packets in Wireshark?

- Capture filters: Limit the data collected during packet capture.
- Display filters: Narrow down displayed packets for easier analysis.
- Filtering enhances performance, reduces noise, speeds up troubleshooting.

5.

What security concerns should be considered when using Wireshark?

→ Unauthorized packet capture can expose sensitive data such as passwords or confidential communications.

→ Wireshark does not decrypt encrypted traffic but plaintext data remains visible.

→ Running Wireshark with administrative privileges can pose security risks.

6. How does Wireshark differentiate between various network protocols?

→ Wireshark uses protocol dissectors, which analyze packet headers and payloads to classify traffic based on predefined rules. It recognizes protocols by inspecting fields such as Ethernet, port numbers and protocol identifiers (IP, ARP).

7. Why is Wireshark widely used in cybersecurity investigations?

→ It detects suspicious network traffic, such as malware communication and unauthorized access attempts.

→ Monitors encrypted traffic patterns for anomalies without decrypting data.

→ It helps in analyse distributed attacks by identifying malicious packet floods.

Section 3: Comparative Analysis

1. Compare and contrast Nmap and Wireshark in terms of their functionalities and use cases.

→ Nmap

Wireshark

(i) Network scanning and discovery.	(ii) Packet capture and analysis.
(iii) It identifies open ports, services and hosts on a network.	(iv) It inspects the real-time network traffic at a granular level.
(v) It sends packets to gather information.	(vi) It listens to network traffic without sending packets.
(vi) It identifies network structure, weak points and misconfigurations.	(vii) It analyses detailed network communications and detecting anomalies.

- Q. Which tool would you use for detecting open ports and why? → Nmap is the best tool because:
- It actively probes target hosts and check port statuses (open, closed, filtered).
 - It provides detailed service.
 - It is designed specifically for network scanning and security auditing.

3.

How can both Nmap and Wireshark be used together for network security analysis?

- Step 1: Use Nmap to scan for vulnerabilities
- Identify active hosts.
 - Detect open ports and services.
 - Find OS details.

→ Step 2: Use Wireshark to analyze suspicious traffic

- Capture network traffic to examine packet behavior.
- Filter specific traffic.
- Detect anomalies such as excessive failed login attempts or unusual data transfers.

→ Step 3: Correlate findings

- If Nmap detects an open port running a vulnerable service, Wireshark can confirm if it's being exploited.
- If Wireshark detects suspicious traffic, Nmap can scan for unprotected devices that may be the source.

4. Discuss a real-world scenario where Nmap and Wireshark could help in identifying a network attack.

— Scenario - Detecting a Brute-force Attack on a SSH Server.

→ Step 1: Use Nmap to identify Open SSH Ports.

- Run `nmap -p 22 -sV Target-ip` to check if SSH is open.

- If detected, further analyse version details to check for vulnerabilities.

→ Step 2: Use Wireshark to Monitor SSH traffic.

- Apply a filter (`tcp port == 22`) to capture all SSH related packets.

- Detect multiple failed login attempts from the same IP, indicating a brute-force attack.

→ Step 3: Take Action.

- Block the attacker's IP using a firewall or intrusion prevention system (IPS).
- Enforce stronger authentication mechanisms.
- Monitor further traffic for other attack patterns.

Assignment-11

1. Basic Nmap Commands

nmap<target>

Basic scan of a target (IP or hostname)

```
Nmap Output Ports / Hosts Topology Host Details Scans
nmap 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:44 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.191.36.214
Host is up (0.00012s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3306/tcp   open  mysql
7070/tcp   open  realserver

Nmap done: 1 IP address (1 host up) scanned in 0.49 seconds
```

nmap – v <target>

Verbose output of the scan

```
nmap -v 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:44 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Initiating SYN Stealth Scan at 19:44
Scanning 172.191.36.214 [1000 ports]
Discovered open port 445/tcp on 172.191.36.214
Discovered open port 3306/tcp on 172.191.36.214
Discovered open port 139/tcp on 172.191.36.214
Discovered open port 135/tcp on 172.191.36.214
Discovered open port 7070/tcp on 172.191.36.214
Completed SYN Stealth Scan at 19:44, 0.05s elapsed (1000 total ports)
Nmap scan report for 172.191.36.214
Host is up (0.00056s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3306/tcp   open  mysql
7070/tcp   open  realserver

Read data files from: C:\Program Files (x86)\Nmap
Nmap done: 1 IP address (1 host up) scanned in 0.43 seconds
Raw packets sent: 1000 (44.000KB) | Rcvd: 2005 (84.220KB)
```

nmap -sS <target>

SYN (Stealth) scan

```
nmap -sS 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:54 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.191.36.214
Host is up (0.00012s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3306/tcp   open  mysql
7070/tcp   open  realserver

Nmap done: 1 IP address (1 host up) scanned in 0.42 seconds
```

nmap -sT <target>

TCP Connect scan (less stealthy)

```
nmap -sT 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:45 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.191.36.214
Host is up (0.0017s latency).
Not shown: 995 filtered tcp ports (no-response)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3306/tcp   open  mysql
7070/tcp   open  realserver

Nmap done: 1 IP address (1 host up) scanned in 5.64 seconds
```

nmap -sU <target>

UDP scan

```
nmap -sU 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:45 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.191.36.214
Host is up (0.00031s latency).
Not shown: 990 closed udp ports (port-unreach)
PORT      STATE      SERVICE
PORT      STATE      SERVICE
80/udp    open|filtered http
123/udp   open|filtered ntp
137/udp   open|filtered netbios-ns
138/udp   open|filtered netbios-dgm
443/udp   open|filtered https
1900/udp  open|filtered upnp
4500/udp  open|filtered nat-t-ike
5050/udp  open|filtered mmcc
5353/udp  open|filtered zeroconf
5355/udp  open|filtered llmnr

Nmap done: 1 IP address (1 host up) scanned in 181.33 seconds
```

nmap -p <target>

Scan all 65535 ports

```
nmap -p - 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:46 India Standard Time
Nmap scan report for 172.191.36.214
Host is up (0.00034s latency).
Not shown: 65516 closed tcp ports (reset)
PORT      STATE    SERVICE
135/tcp    open     msrpc
137/tcp    filtered netbios-ns
139/tcp    open     netbios-ssn
445/tcp    open     microsoft-ds
3306/tcp   open     mysql
5040/tcp   open     unknown
7070/tcp   open     realserver
7680/tcp   open     pando-pub
8885/tcp   open     unknown
8886/tcp   open     unknown
33060/tcp  open     mysqlx
49664/tcp  open     unknown
49665/tcp  open     unknown
49668/tcp  open     unknown
49669/tcp  open     unknown
49672/tcp  open     unknown
49688/tcp  open     unknown
57272/tcp  open     unknown
57621/tcp  open     unknown

Nmap done: 1 IP address (1 host up) scanned in 4.22 seconds
```

nmap -O <target>

Try to detect OS

```

nmap -O 172.191.36.214

Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:47 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.191.36.214
Host is up (0.0004s latency).

Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3306/tcp   open  mysql
7070/tcp   open  realserver

No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).

TCP/IP fingerprint:
OS:SCAN(V=7.95%E=D#4=13%OT=135%CT=1%CU=34769%PV=N%D$=0%D=L%G=Y%TM=67FBC7
OS:84%P=I686-pc-windows-windows)SEQ(SP=103%CD=1%ISR=109%T=I%CI=I%II=I%SS=
OS:$STS=A)SEQ(SP=106%CD=1%ISR=109%T=I%CI=I%II=I%SS=S%TS=A)SEQ(SP=106%CD=
OS:1%ISR=108%T=I%CI=I%II=I%SS=S%TS=A)SEQ(SP=2%CD=1%ISR=10C%T=I%CI=I%II=
OS:$STS=A)OPS(01=MFPD7NW8NT11%O3=MFPD7NW8NT11%O4=MFPD
7NW8NT11%O5=MFPD7NW8NT11%O6=MFPD7S11)WIN(W1=FFFFFW2=FFFFFW3=FFFFFW4=FFF
OS:F4W5=FFFFFW6=FFFFECN(R=YDF=Y%T=80%W=FFFFFO=MFMD7NW8NNSC=C%NQ%)T1(R=Y%
OS:D%F=Z%A=S%F=ASRD=0%Q%)T2(R=Y%DF=Y%T=80%W=0%S=Z%A=S%F=AR%O=%RD=0%Q%
OS:0%Q%)T3(R=Y%DF=Y%T=80%W=0%S=2%A=0%F=AR%O=%RD=0%Q%)T4(R=Y%DF=Y%T=80%W=0%S
OS:=%F=DF=Y%T=80%W=0%S=AA=A=0%F=R%O=%RD=0%Q%)T5(R=Y%DF=Y%T=80%W=0%S=Z%A=S%F=
OS:AR%O=%RD=0%Q%)UL(R=Y%DF=N%T=0%IPL=164%UN=0%IRPL=G%RID=G%RICK=G%RUCK=G%
OS:IRUD=G)IE(R=Y%DFI=N%T=80%CD=2)

Network Distance: 0 hops

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 10.70 seconds

```

nmap -A <target>

Aggressive scan (OS, version, script scanning, traceroute)

```

nmap -A 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:48 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.191.36.214
Host is up (0.00051s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE          VERSION
135/tcp    open  msrpc           Microsoft Windows RPC
139/tcp    open  netbios-ssn     Microsoft Windows netbios-ssn
445/tcp    open  microsoft-ds
3306/tcp   open  mysql           MySQL (unauthorized)
7070/tcp   open  ssl/realserver?
| ssl-cert: Subject: commonName=AnyDesk Client
| Not valid before: 2023-07-27T16:15:58
|_ Not valid after: 2023-07-14T16:15:58
_|_ssl-date: TLS randomness does not represent time
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).

TCP/IP fingerprint:
OS:SCAN(V=7.95%#D=4/13%OT=135%CT=1%CU=32230%PV=N%DS=0%DC=L%G=Y%TM=67FBC7
OS:CA#P=1606-%c-windows-windows) SEQ(SP=101%GCD=%ISR=10B#T=I%CI=I%II=I%SS=
OS:%TS=A) SEQ(SP=106%GCD=1%ISR=100%TI=%CI=I%II=I%SS=S%TS=A) SEQ(SP=FC%GCD=1
OS:%SR=F8#T=I%CI=I%II=I%TS=A) SEQ(SP=F8#GCD=1%ISR=10C#T=I%CI=I%II=I%SS=S%
OS:TS=A) SEQ(SP=PP#GCD=1%ISR=108#T=I%CI=I%II=I%SS=S%TS=A) OPS(O1=MFPD7NW8ST1
OS:1%O2=MFPD7NW8ST11%O3=MFPD7NW8NT11%O4=MFPD7NW8ST11%O5=MFPD7NW8ST11%O6=MP
OS:W1#F#W2=FFFF#W3=FFFF#W4=FFFF#W5=FFFF#W6=FFFF) ECN(R=Y#DF=Y
OS:4#T=80#W=FFFF#O=MFPD7NW8NNNS%CC=N5#O=T1 (R=Y#DF=Y#T=80#S=0%#A=8%#F=AS#RD=0%Q
OS:T2 (R=Y#DF=Y#T=80#W=0%#S=2%#F=AR#O=%RD=0%Q) T3 (R=Y#DF=Y#T=80#W=0%#S=2%
OS:A=0%#F=AR#O=%RD=0%Q) T4 (R=Y#DF=Y#T=80#W=0%#S=2%#F=AR#O=%RD=0%Q) T5 (R=Y#D
OS:F#Y#T=80#W=0%#S=2%#F=AR#O=%RD=0%Q) T6 (R=Y#DF=Y#T=80#W=0%#S=2%#F=AR#O=%RD=0%Q
OS:S#RD=0%Q) T7 (R=Y#DF=Y#T=80#W=0%#S=2%#F=AR#O=%RD=0%Q) U1 (R=Y#DF=NST=80
OS:IPL=164#UN=0#IRIPL=G#RID=G#RIPCR=G#RUCK=G#RUD=G) IE(R=Y#DFI=N#T=80#CD=Z)

Network Distance: 0 hops
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
| smb2-time:
|   date: 2025-04-13T14:18:39
|_ start_date: N/A
| smb2-security-mode:
|   3:1:::
|_ Message signing enabled but not required

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 36.82 seconds

```

nmap -sV <target>

Detect service versions

```

nmap -sV 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:49 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.191.36.214
Host is up (0.00091s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE          VERSION
135/tcp    open  msrpc           Microsoft Windows RPC
139/tcp    open  netbios-ssn     Microsoft Windows netbios-ssn
445/tcp    open  microsoft-ds
3306/tcp   open  mysql           MySQL (unauthorized)
7070/tcp   open  ssl/realserver?
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.96 seconds

```

nmap -T4 <target>

Faster scan timing (T0 to T5, T4 is common for speed)

```

nmap -T4 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:50 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.191.36.214
Host is up (0.000071s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3306/tcp   open  mysql
7070/tcp   open  realserver

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

```

nmap -Pn <target>

Skip host discovery (assume host is up)

```

nmap -Pn 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:50 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.191.36.214
Host is up (0.00016s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3306/tcp   open  mysql
7070/tcp   open  realserver

Nmap done: 1 IP address (1 host up) scanned in 0.44 seconds

```

nmap --script vuln <target>

Run common vulnerability scripts

```
nmap --script vuln 172.191.36.214
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-13 19:52 India Standard Time
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 172.191.36.214
Host is up (0.00034s latency).
Not shown: 995 closed tcp ports (reset)
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3306/tcp   open  mysql
7070/tcp   open  realserver

Host script results:
|_smb-vuln-ms10-054: false
|_smb-vuln-ms10-061: Could not negotiate a connection:SMB: Failed to receive bytes: ERROR
|_samba-vuln-cve-2012-1182: Could not negotiate a connection:SMB: Failed to receive bytes: ERROR

Nmap done: 1 IP address (1 host up) scanned in 29.05 seconds
```

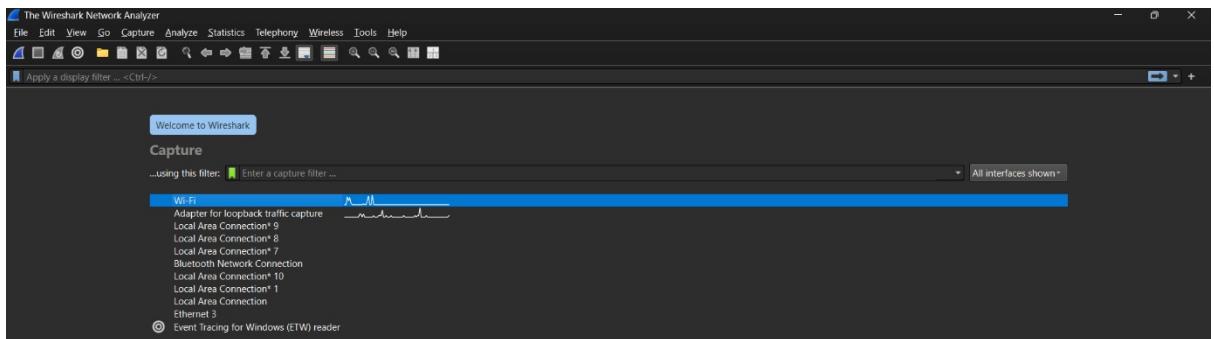
2. Capturing and Analyzing Network Traffic Using Wireshark

1. Download and Install Wireshark:

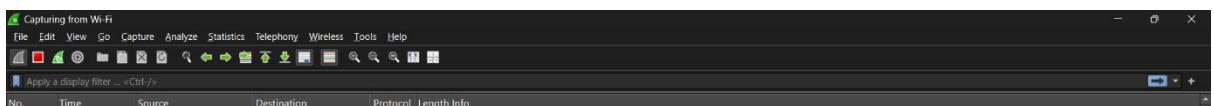
- o Visit <https://www.wireshark.org/>
- o Download and install the latest version for your operating system.

2. Start a Packet Capture Session:

- o Open Wireshark.
- o Select your active network interface (e.g., Wi-Fi or Ethernet).



- o Click the Start Capturing Packets button (shark fin icon).



3. Generate Some Network Traffic:

- o Open a web browser and visit a few websites.
- o You may also open apps that use the internet like email or messaging tools.

4. Stop the Capture:

- o After about 2–5 minutes, stop the capture using the red square icon in Wireshark.

A screenshot of the Wi-Fi Network Monitor application. The interface includes a menu bar with File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with icons for file operations, search, and other tools. A status bar at the bottom displays "Apply a display filter ... <Ctrl-/->". The main window shows a table of network traffic with columns for No., Time, Source, Destination, Protocol, Length, and Info.

5. Analyze the Traffic:

- o Use filters Simple filter HTTP, DNS, or TCP to view specific types of traffic.

HTTP :

File	Edit	View	Go	Capture	Analyze	Statistics	Telephony	Wireless	Tools	Help
No.	Time	Source	Destination	Protocol	Length	Info				
21966	58.020238	34.184.35.123	172.191.36.198	HTTP	605	HTTP/1.1 200 OK				
21970	58.117600	172.191.36.198	34.184.35.123	HTTP	387	GET /edge/relase2/chrome_component/glikajjhx6vvdhrlrvbclvcyfba_499/lmeqlgejhemejginpboagddgfbebpgm_p_499_-				
21999	58.140331	34.184.35.123	172.191.36.198	HTTP	349	HTTP/1.1 200 OK				
22217	62.356986	172.191.36.198	34.184.35.123	HTTP	314	HEAD /edge/defdiffgen-puffin/nikhgdajlphfehepbhblakbdgeejf/1678659f9ff65787dfdf5950febf64fc935fd32632b3419-				
22222	62.367491	34.184.35.123	172.191.36.198	HTTP	604	HTTP/1.1 200 OK				
22321	62.456043	172.191.36.198	34.184.35.123	HTTP	365	GET /edge/defdiffgen-puffin/nikhgdajlphfehepbhblakbdgeejf/1678659f9ff65787dfdf5950febf64fc935fd32632b3419-				
22326	62.483838	34.184.35.123	172.191.36.198	HTTP	812	HTTP/1.1 200 OK				
23075	66.767747	172.191.36.198	23.60.172.26	HTTP	411	HEAD /filestreamingservice/files/5d32607d-eea9-44fc-ac55-77800b9862a5?P1=17449545338&P2=404&P3=284+J10TNE8k-				
23077	66.806879	23.60.172.26	172.191.36.198	HTTP	666	HTTP/1.1 200 OK				
23082	66.859675	172.191.36.198	23.60.172.26	HTTP	482	GET /filestreamingservice/files/5d32607d-eea9-44fc-ac55-77800b9862a5?P1=17449545338&P2=404&P3=284+J10TNE8k-				
23088	66.874085	23.60.172.26	172.191.36.198	HTTP	204	HTTP/1.1 206 Partial Content (application/x-chrome-extension)				
23125	69.146520	172.191.36.198	23.60.172.26	HTTP	485	GET /filestreamingservice/files/5d32607d-eea9-44fc-ac55-77800b9862a5?P1=17449545338&P2=404&P3=284+J10TNE8k-				
23131	69.165384	23.60.172.26	172.191.36.198	HTTP	894	HTTP/1.1 206 Partial Content (application/x-chrome-extension)				
23162	70.352892	172.191.36.198	23.60.172.26	HTTP	487	GET /filestreamingservice/files/5d32607d-eea9-44fc-ac55-77800b9862a5?P1=17449545338&P2=404&P3=284+J10TNE8k-				
23165	70.367983	23.60.172.26	172.191.36.198	HTTP	277	HTTP/1.1 206 Partial Content (application/x-chrome-extension)				
23199	71.368818	172.191.36.198	23.60.172.26	HTTP	487	GET /filestreamingservice/files/5d32607d-eea9-44fc-ac55-77800b9862a5?P1=17449545338&P2=404&P3=284+J10TNE8k-				
23198	71.409982	23.60.172.26	172.191.36.198	HTTP	201	HTTP/1.1 206 Partial Content (application/x-chrome-extension)				
23213	72.384930	172.191.36.198	23.60.172.26	HTTP	488	GET /filestreamingservice/files/5d32607d-eea9-44fc-ac55-77800b9862a5?P1=17449545338&P2=404&P3=284+J10TNE8k-				

TCP :

DNS :

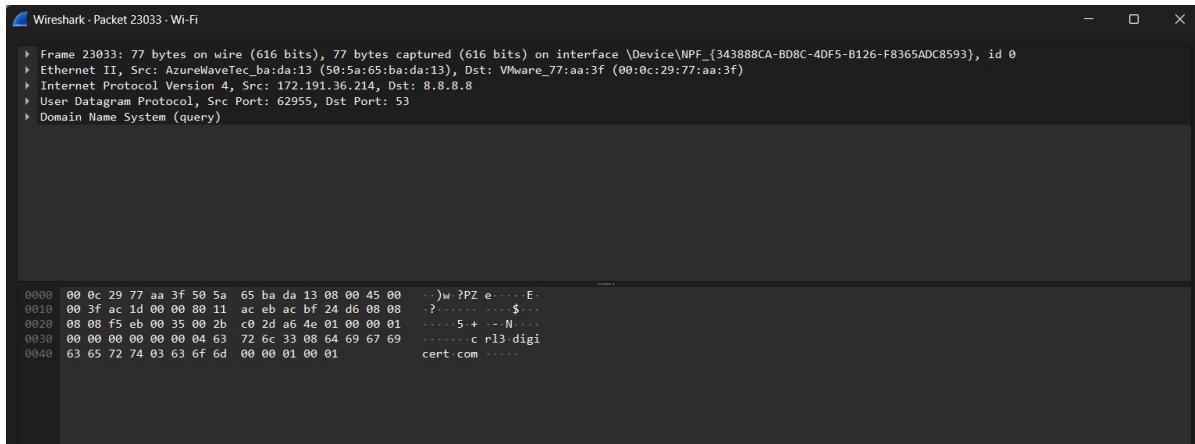
No.	Time	Source	Destination	Protocol	Length Info
187	16.9.72929	172.191.36.214	8.8.4.4	DNS	83 Standard query 0xc7ad A chatgpt.com
191	16.9.76768	172.191.36.214	8.8.4.4	DNS	83 Standard query 0x6dd9 HTTPS.chatgpt.com
194	16.9.79025	8.8.4.4	172.191.36.214	DNS	113 Standard query 0x7a09 A chatgpt.com A 172.64.155.209 A 104.18.32.47
198	16.9.91150	8.8.4.4	172.191.36.214	DNS	146 Standard query response 0x6d49 HTTPS.chatgpt.com SOA hassan.ns.cloudflare.com
213	18.464856	172.191.36.214	8.8.4.4	DNS	92 Standard query 0xd697 HTTPS.beacons.gcp.gvt2.com
255	18.465163	172.191.36.214	8.8.4.4	DNS	93 Standard query 0xb998 A beacons.gcp.gvt2.com
258	18.473190	8.8.4.4	172.191.36.214	DNS	181 Standard query response 0x6d97 HTTPS.beacons.gcp.gvt2.com CNAME beacons-handoff.gcp.gvt2.com SOA ns1.google.com
262	18.473880	8.8.4.4	172.191.36.214	DNS	140 Standard query response 0xb997 A beacons.gcp.gvt2.com CNAME beacons-handoff.gcp.gvt2.com A 142.250.194.195
329	19.519196	172.191.36.214	8.8.4.4	DNS	86 Standard query 0xfa07 HTTPS。www.google.com
333	19.52612	172.191.36.214	8.8.4.4	DNS	86 Standard query 0xfa07 A www.google.com
336	19.529990	8.8.4.4	172.191.36.214	DNS	113 Standard query response 0xfa07 HTTPS.www.google.com HTTPS
340	19.54303	8.8.4.4	172.191.36.214	DNS	104 Standard query response 0xfc67 A www.google.com A 216.58.200.196
435	19.85925	172.191.36.214	8.8.4.4	DNS	86 Standard query 0x762f A lh3.google.com
436	19.85996	8.8.4.4	172.191.36.214	DNS	85 Standard query 0x762f A lh3.google.com
439	19.859996	8.8.4.4	172.191.36.214	DNS	124 Standard query response 0x762f A lh3.google.com CNAME lh2.l.google.com A 142.250.193.206
447	19.861501	8.8.4.4	172.191.36.214	DNS	155 Standard query response 0x0205 HTTPS.lh3.google.com CNAME lh2.l.google.com SOA ns1.google.com
467	19.942140	172.191.36.214	8.8.4.4	DNS	100 Standard query 0x8331 HTTPS.opads-pa.clients6.google.com
469	19.942212	172.191.36.214	8.8.4.4	DNS	100 Standard query 0x7eab8 A opads-pa.clients6.google.com
481	19.964868	8.8.4.4	172.191.36.214	DNS	152 Standard query response 0x8331 HTTPS.opads-pa.clients6.google.com SOA ns1.google.com
482	19.964868	8.8.4.4	172.191.36.214	DNS	118 Standard query response 0x7a8b A opads-pa.clients6.google.com A 142.250.193.74
Frame 198: 146 bytes on wire (1168 bits), 146 bytes captured (1168 bits) on interface \Device\NPF_{34388...} (00:0c:29:77:a3:f3), Dst: AzureWaveNet_Bridge[13] (59:5a:65:ba:d1:13)					
Ethernet II, Src: VMware_77:a3:f3 (00:0c:29:77:a3:f3), Dst: AzureWaveNet_Bridge[13] (59:5a:65:ba:d1:13)					
Internet Protocol Version 4, Src: 8.8.4.4, Dst: 172.191.36.214					
Transmission Control Protocol, Src Port: 53, Dst Port: 61469, Seq: 1, Ack: 32, Len: 92					
Domain Name System (response)					

- o Select a few packets and inspect their details in the packet details pane.

```
Wireshark - Packet 23041 - Wi-Fi

Frame 23041: 63 bytes on wire (504 bits), 63 bytes captured (504 bits) on interface \Device\NPF_{343888CA-BD8C-4DF5-B126-F8365ADC8593}, id 0
Ethernet II, Src: VMware_77:aa:3f (00:0c:29:77:aa:3f), Dst: AzureWaveTec_bada13 (50:5a:65:bada:13)
Internet Protocol Version 4, Src: 162.247.243.29, Dst: 172.191.36.214
Transmission Control Protocol, Src Port: 443, Dst Port: 62827, Seq: 4249, Ack: 2029, Len: 9
[2 Reassembled TCP Segments (215 bytes): #23040(206), #23041(9)]
Transport Layer Security
```

```
0000  50 5a 65 ba da 13 00 0c  29 77 aa 3f 08 00 45 28  PZ.....)w?..-E(      .....
0010  00 31 1e 29 40 00 38 06  bc cb a2 f7 f3 1d ac bf  _1 )@ B.....-
0020  24 d6 01 bb f5 6b bc 22  59 69 35 31 53 69 50 18  $...k " Y151SiP
0030  01 1e 70 e1 00 00 7b d8  69 f3 ff 3d 69 c1 f2  ..p...{. 1 ..=1 ..
```



6. Export Your Capture: Save the capture as a .pcapng file for later review.

https://drive.google.com/drive/folders/1aHY89uyI8rcGuUdVozl8XzqsG5Se3KVf?usp=drive_link

Theory:-

Assignment-11

I) Basic Nmap Commands.

Command	Description
a) nmap <target>	Basic uscan of a target
b) nmap -v <target>	verbose output of the scan
c) nmap -sS <target>	SYN (stealth) scan
d) nmap -st <target>	TCP connect scan (less stealthy)
e) nmap -sU <target>	UDP scan
f) nmap -P2 <target>	Scan specific port(s)
g) nmap -Pf <target>	Scan all 65535 ports
h) nmap -O <target>	try to detect OS
i) nmap -A <target>	aggressive scan (OS, version, script scanning)
j) nmap -SV <target>	detect service versions
k) nmap -T4 <target>	faster scan timing (ToToTs, T4 is common for speed)
l) nmap -Pn <target>	skip host discovery (assume host is up)
m) nmap -L target.txt	scan multiple targets from a file.
n) nmap -ON <target>	output save output in normal format
o) nmap -script <script-name> <target>	run specific NSE script
p) nmap --script vuln <target>	Run common vulnerability scripts.

Q2 Capturing and analyzing network traffic using Wireshark.

Q1 What interface did you select for capturing packets?

Soln: For capturing packets, I selected Wi-Fi interface because my computer is connected to the internet via a wireless network. In Wireshark, all available network interfaces are listed when the application is launched, and selecting the correct one is essential to monitor the traffic effectively. The Wi-Fi interface typically shows ongoing data transmission in the form of spikes which indicates active network traffic. This interface allows the capture of all incoming and outgoing packets from the wireless connection in real-time.

Q2 List 3 protocols you observed during the capture session.

Soln: I observed the three commonly used network protocols: HTTP, TCP, and UDP.

1) HTTP (Hyper Text Transfer Protocol) is used for transferring web-page over the internet. I noticed several HTTP requests when I opened different websites.

- DATE
- 2) DNS (Domain name system)
this protocol translates domain names into IP addresses whenever I typed a URL. DNS packet generated to resolve the IP address.
- 3) TCP (Transmission control protocol)
A connection-oriented protocol that ensures reliable communication between devices. TCP packets were visible in every connection, especially during communications.
- Q3 Pick one HTTP packet. What was the host website being accessed?
⇒ I selected an HTTP GET request packet during the session. The host accessed in that particular packet was: www.example.com (Note: Replace this with actual domain name you saw during my session such as www.google.com, www.wikipedia.org etc.)
In the packet details, under the HTTP section the 'Host' field displays the exact domain being requested. The host field shows which server my browser or was trying to reach using the HTTP protocol.
- Q4 What is the IP address of your system and the destination IP for the selected packet?

1) Ethernet II: This is the datalink layer that includes the MAC addresses of the source and destination devices within the local network.

2) Internet Protocol version 4 (IPv4): This is the network layer, responsible for identifying source and destination IP addresses.

3) Transmission control protocol (TCP): This is the transport layer, ensuring reliable transmission and providing ports to identify applications.

4) Hypertext Transfer Protocol (HTTP): This is the application layer, where the actual web request (such as GET or POST) is visible. Each layer wraps the data from the layer above, which is called encapsulation when the packet is received from the layers in reverse order.

Q6: What did you learn from this activity about how data travels over the internet?

This activity offered a hands-on understanding of how data travels through the Internet using different protocol layers.

- Some Key takeaways:
- 1) Layer architecture - Data is encapsulated in layer (Ethernet, IP, HTTP, TCP) during transmission and de-encapsulated at the receiver's end.
 - 2) Protocol role - Each protocol serves a specific function - for example, DNS reserves domain names, IP handles addressing and TCP ensures reliable delivery.
 - 3) Traffic visibility - Wireshark allows us to inspect the raw data being transmitted, including requests to websites, responses from servers, and system interactions over the network.
 - 4) Security considerations - Observing how easily HTTP data can be read underscores the importance of using HTTPS for secure communication.
 - 5) Packet-level understanding - Viewing each packet's structure gives a deeper appreciation of the technical details involved in something as simple as opening a web page.
- Rid
Wu

Q1) What is the purpose of using filter in Wireshark?

Ans: Filters helps you focus on specific types of network traffic, making it easier to analyze and troubleshoot issue without being overwhelmed by all the data.

Q2) Which network interface did you use for capturing packets? Why?

Ans: You would typically use the network interface that is connected to the internet or the local network you want to analyze. This allows you to capture relevant traffic.

Q3) Name at least 3 different protocols you observed during the capture.

→ Common protocols you might observe include

- HTTP (for web traffic)
- DNS (for domain name resolution)
- TCP (for reliable data transmission)

Q4) What is the IP address for your system (Source IP)?

Ans: You can find your system's IP address in Wireshark by looking at the source IP of any particular packet capture. It usually starts with "192.168." or "10.x" for local network.

Q5) What is one destination IP you communicate with during the capture?

Looks for any packet where your system's IP is the source. The destination IP will be the address.

of the server or device you communicate with
(eg: a website's IP)

6) Find a DNS query in your capture. What domain name was being looked up?

Ans Look for ipackets with the DNS protocol. You can find a query that shows a domain name like "example.com" being looked up.

Q7) Pick one HTTP packets and describe its source . . .

Ans Select an HTTP packets and check the details. The source IP is your system's IP, the destination IP is the web server's IP, and most the most Host name is usually found in the HTTP header.
(eg: "Host : www.example.com"):

8) Which filter showed the most traffic? why do you think that is ?

Ans The HTTP filter often shows the most traffic because web browsing generates a lot of request and responses. Many user access website frequently leading to high HTTP traffic.

9) Did you observe any encrypted traffic . . . ?

Ans Yes, you can tell if traffic is encrypted by looking for packets with the "TLS" or "SSL" protocol. These packet usually indicate secure connections, like HTTPS website.

10) What's the difference between 'ip.addr == x.x.x.x' and 'ip.src == x.x.x.x'?

$ip\cdot addr == x\cdot x\cdot x\cdot x$: This filter shows packets where the IP address is e.g. either the source or destination.

$ip\cdot src == x\cdot x\cdot x\cdot x$: This filter shows only packets where the IP address is the source

11) FIND a TCP handshake (SYN, SYN-ACK, ACK)

→ A TCP handshake involves three steps:

1. SYN: Your system sends a SYN packet to initiate a connection.
2. SYN - ACK: The server responds with a SYN-ACK packet to acknowledge the request.
3. ACK: Your system sends an ACK packet to confirm the connection is established.

12) What's one interesting thing you noticed in your network traffic?

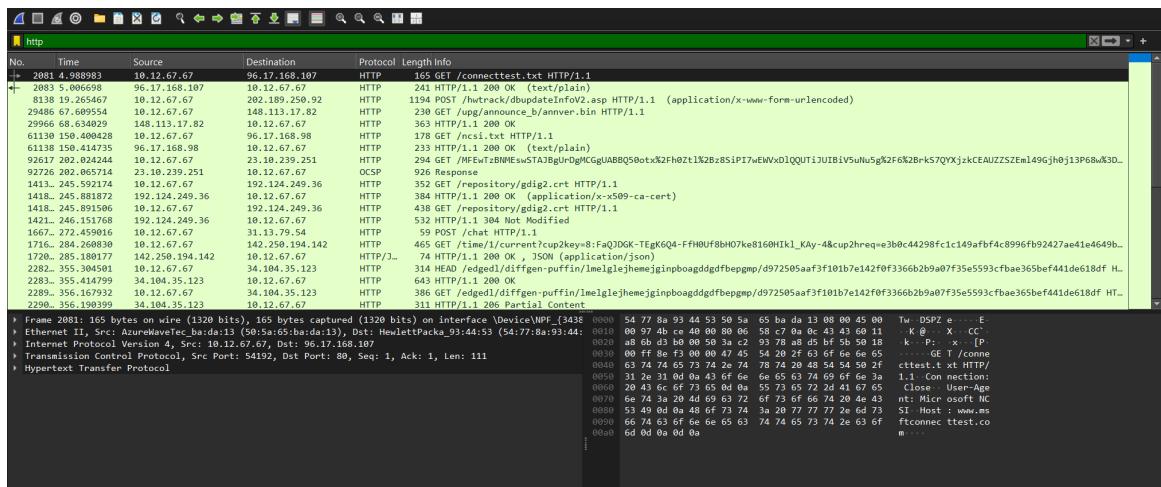
→ You might notice unusual traffic patterns such as a high number of DNS queries, unexpected IP addresses communicating with your system, or a lot of encrypted traffic, which could indicate secure browsing or potential security concerns.

Assignment-12

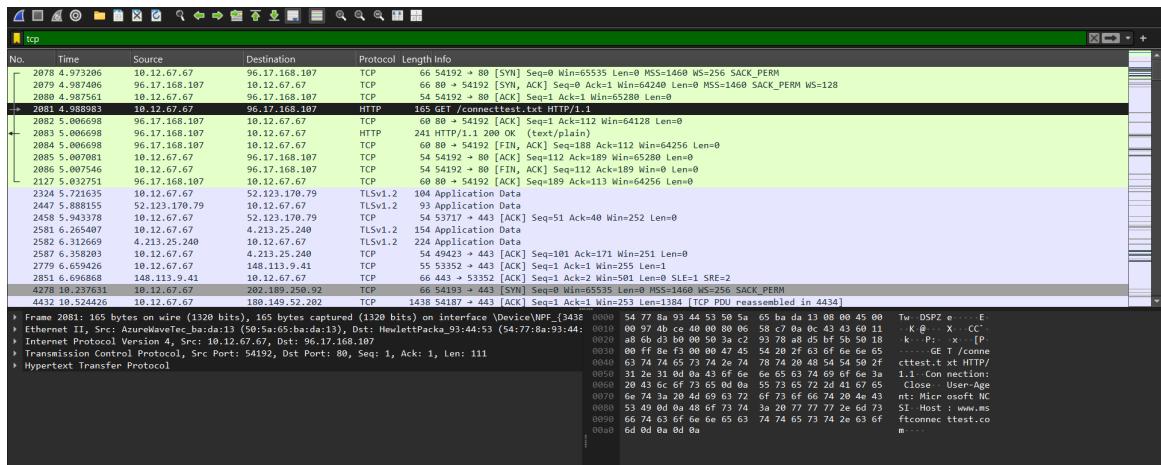
Wireshark display filters :-

1. Protocol-Based Filters

- http – Shows only HTTP traffic



- tcp – Displays only TCP packets



- udp – Displays only UDP packets

No.	Time	Source	Destination	Protocol	Length Info
2061 4.849059	fe80::1:50:9d1:3d8c::ff02::fb	MDNS	156 Standard query 0x0000 TXT Saranya's MacBook Air._companion-link._tcp.local, "QU" question TXT ANKIT's MacBook Air._companion-link._tcp.local, "QU" question TXT Mayank's MacBook Pro (2)._companion-link._tcp.local TXT TXT, cache flush AAAA, cache flush f...		
2062 4.849059	fe80::1:6cbf:8e10::ff02::fb	MDNS	451 Standard query response 0x0000 PTR Soumya's MacBook Pro (2)._companion-link._tcp.local TXT TXT, cache flush AAAA, cache flush f...		
2063 4.849059	10.12.1.66	MDNS	451 Standard query response 0x0000 PTR Soumya's MacBook Pro (2)._companion-link._tcp.local TXT TXT, cache flush AAAA, cache flush f...		
2064 4.849059	10.6.4.45	239.255.255.250	SSDP	489 NOTIFY + HTTP/1.1	
2065 4.849059	fe80::1:7b5::fb77:ddc::ff02::fb	SSDP	521 NOTIFY + HTTP/1.1		
2066 4.849059	10.12.1.66	10.12.1.66	DNS	160 Standard query 0x0000 TXT ANKIT's MacBook Air._companion-link._tcp.local, "QU" question	
2067 4.876747	10.12.1.67	10.2.1.60	DNS	83 Standard query 0x0000 A www.msftconnecttest.com	
2068 4.913626	fe80::1:419::7a00:9b4::ff02::fb	MDNS	223 Standard query 0x0000 PTR _companion-link._tcp.local, "QU" question PTR _rdlink._tcp.local, "QU" question PTR Ayush's MacBook A...		
2069 4.913626	fe80::1:45c::8500:27a::ff02::fb	MDNS	128 Standard query 0x0000 TXT ANKIT's MacBook Air._companion-link._tcp.local, "QU" question		
2070 4.913626	10.12.1.66	10.12.1.66	DNS	200 Standard query 0x0000 A www.msftncsi.com.edgesuite.net	
2093 5.019380	0.0.0.0	255.255.255.255	DHCP	362 DHCP Request - Transaction ID 0xbFB2328a	
2093 5.019380	10.6.13.1	224.0.0.252	LLMNR	75 Standard query 0x000f ANY LAPTOP-1093V2R	
2094 5.020397	10.12.1.66	239.255.255.250	MDNS	125 Standard query 0x0000 TXT Mayank's MacBook Air (4)._airplay._tcp.local, "QU" question TXT Mayank's Air._airplay._tcp.local, "QU" question	
2096 5.021043	fe80::0054:ffff:feaa::ff02::fb	MDNS	107 Standard query 0x0000 PTR _spotifyconnect._tcp.local, "QU" question		
2097 5.022083	10.6.11.62	224.0.0.251	MDNS	755 Standard query response 0x0000 PTR FABCE4E4B87@amrita's MacBook Pro._raop._tcp.local PTR Amrita's MacBook Pro._companion-link...	
2098 5.022083	fe80::c1:7197::ff02::fb	MDNS	145 Standard query 0x0000 TXT Aryan's MacBook Air (4)._airplay._tcp.local, "QU" question TXT Mayank's Air._airplay._tcp.local, "QU" question		
2099 5.022083	fe80::c1:7197::ff02::fb	MDNS	77 Standard query 0x0000 PTR _spotifyconnect._tcp.local, "QU" question PTR Amrita's MacBook Pro._companion-link...		
2101 5.022083	10.12.41.223	224.0.0.251	MDNS	108 Standard query 0x0000 TXT ANKIT's MacBook Air._companion-link._tcp.local, "QU" question	
2102 5.022083	fe80::1:ceb1bea::697::ff02::fb	MDNS	128 Standard query 0x0000 TXT ANKIT's MacBook Air._companion-link._tcp.local, "QU" question		
2103 5.022083	10.12.45.201	224.0.0.251	MDNS	893 Standard query 0x0000 TXT Mayank's MacBook Pro._raop._tcp.local, "QU" question PTR 468BD0468056565@raop.msnf...	

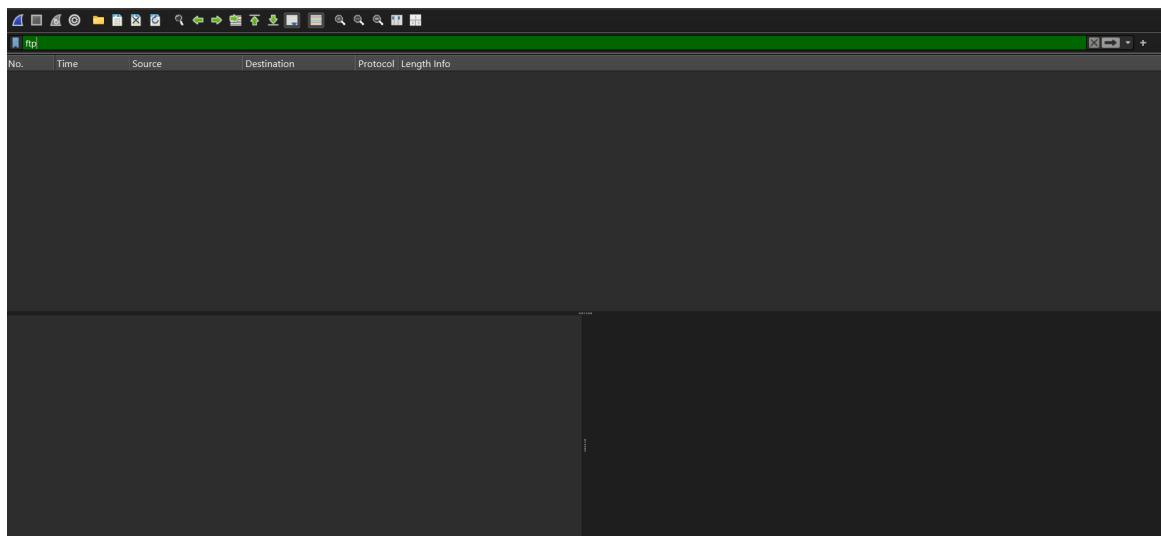
• dns – Shows only DNS traffic

No.	Time	Source	Destination	Protocol	Length Info
2067 4.870674	10.12.67.67	10.2.1.60	DNS	83 Standard query 0x87d3 A www.msftconnecttest.com	
2077 4.970666	10.2.1.68	10.12.67.67	DNS	227 Standard query response 0x87d3 A www.msftconnecttest.com CNAME www.msftncsi.com.edgesuite.net	
4276 10.186359	10.12.67.67	10.2.1.60	DNS	82 Standard query 0x533a A www.computerumbai.com	
4277 10.224247	10.2.1.68	10.12.67.67	DNS	98 Standard query response 0x533a A www.computerumbai.com A 202.189.250.92	
4911 11.868186	10.12.67.67	10.2.1.60	DNS	99 Standard query response 0x1669 A zhu.corpwebcontrol.com A 202.189.250.92	
11034 27.155919	10.12.67.67	10.2.1.60	DNS	91 Standard query 0x9f57 HTTPS teams.events-data.microsoft.com	
11034 27.157732	10.12.67.67	10.2.1.60	DNS	91 Standard query 0xc2f2 HTTPS teams.events-data.microsoft.com	
11034 27.158736	10.12.67.67	10.2.1.60	DNS	79 Standard query 0x9f69 A wpad.DDN.UPEs.AC.IN	
11037 27.201371	10.2.1.60	10.12.67.67	DNS	134 Standard query response 0x9f69 A No such name A wpad.DDN.UPEs.AC.IN SOA authdn.DDN.UPEs.AC.IN	
11039 27.218616	10.2.1.60	10.12.67.67	DNS	213 Standard query 0x0000 PTR _events-data.microsoft.com CNAME teams-events-data.microsoft.com	
11039 27.218616	10.12.67.67	10.2.1.60	DNS	91 Standard query response 0xc2f2 HTTPS teams.events-data.microsoft.com	
12443 31.209206	10.12.67.67	10.2.1.60	DNS	74 Standard query 0x60ee A w21.pnawnet	
12507 31.434489	10.2.1.60	10.12.67.67	DNS	90 Standard query response 0x609e A w21.pnawnet.in A 146.112.61.106	
12891 32.429626	10.12.67.67	10.2.1.60	DNS	71 Standard query 0x6a29 A w4.pnaw.net	
12976 32.496710	10.2.1.68	10.12.67.67	DNS	87 Standard query response 0x6a29 A w4.pnaw.net A 15.235.218.105	
13358 33.476552	10.12.67.67	10.2.1.60	DNS	82 Standard query 0xb085 A w11.corpwebcontrol.com	
14340 33.476552	10.2.1.60	10.12.67.67	DNS	98 Standard query response 0xb085 A w11.corpwebcontrol.com A 150.242.201.76	
15607 34.306955	10.12.67.67	10.2.1.60	DNS	71 Standard query 0x3798 A w5.pnaw.net	
17004 34.438345	10.2.1.60	10.12.67.67	DNS	87 Standard query response 0x3798 A w5.pnaw.net A 148.113.17.82	

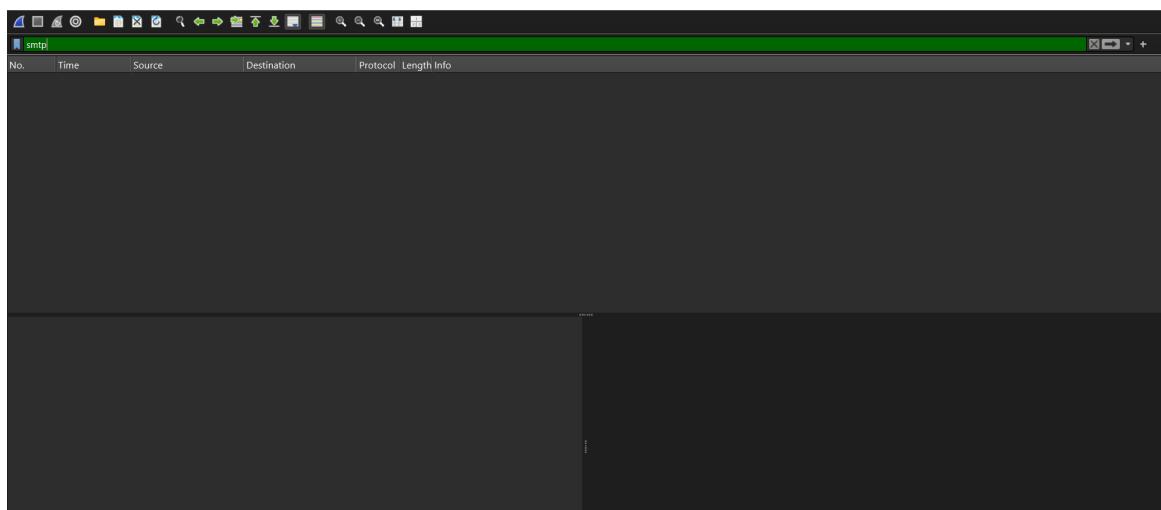
• icmp – Filters ICMP packets (e.g., ping)

No.	Time	Source	Destination	Protocol	Length Info
→ 3018 9.701165	10.12.67.67	142.250.194.196	ICMP	74 Echo (ping) request id=0x0001, seq=22/5632, ttl=128 (reply in 3066)	
← 3066 9.750933	142.250.194.196	10.12.67.67	ICMP	74 Echo (ping) reply id=0x0001, seq=22/5632, ttl=117 (request in 3018)	
→ 3593 10.876550	142.250.194.196	10.12.67.67	ICMP	74 Echo (ping) request id=0x0001, seq=23/5808, ttl=128 (request in 3592)	
3823 11.889435	10.12.67.67	142.250.194.196	ICMP	74 Echo (ping) reply id=0x0001, seq=24/6144, ttl=128 (reply in 3848)	
3848 11.889552	142.250.194.196	10.12.67.67	ICMP	74 Echo (ping) reply id=0x0001, seq=24/6144, ttl=128 (request in 3823)	
4136 12.902741	10.12.67.67	142.250.194.196	ICMP	74 Echo (ping) request id=0x0001, seq=25/6400, ttl=128 (reply in 4137)	
4137 12.982355	142.250.194.196	10.12.67.67	ICMP	74 Echo (ping) reply id=0x0001, seq=25/6400, ttl=117 (request in 4136)	

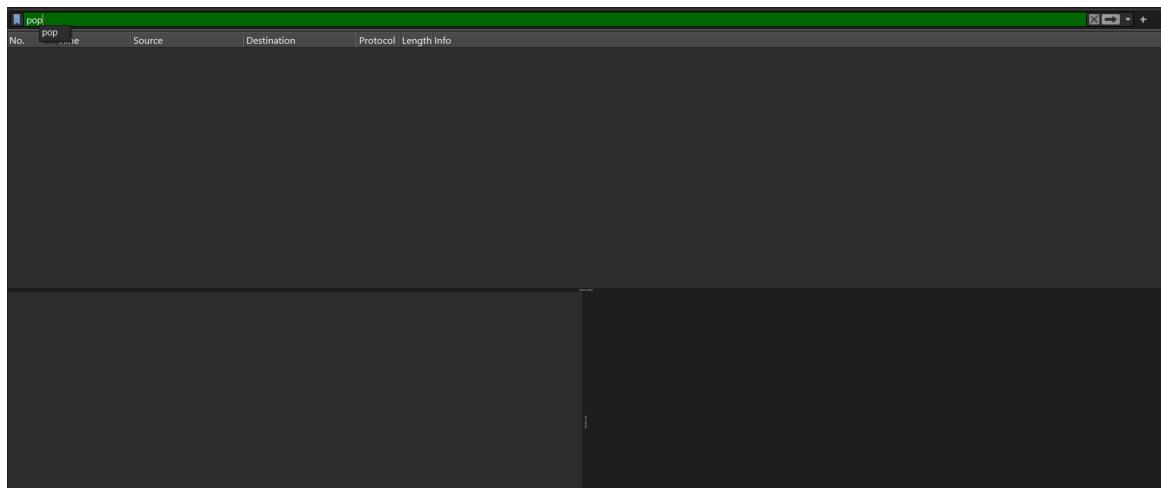
• ftp – Filters FTP traffic



- **smtp** – Filters SMTP (email sending)



- **pop** – Filters POP3 (email receiving)



- **tls or ssl** – Filters encrypted traffic (HTTPS)

No.	Time	Source	Destination	Protocol	Length	Info
2324	5.721639	10.12.67.67	52.123.170.79	TLSv1.2	104	Application Data
2447	5.888155	52.123.170.79	10.12.67.67	TLSv1.2	93	Application Data
2585	6.265407	10.12.67.67	4.213.25.240	TLSv1.2	154	Application Data
2582	6.312669	4.213.25.240	10.12.67.67	TLSv1.2	224	Application Data
4444	10.544246	10.12.67.67	180.149.52.202	TLSv1.2	1249	Application Data, Application Data, Application Data
4446	10.544247	180.149.52.202	10.12.67.67	TLSv1.2	93	Application Data
4462	10.692376	180.149.52.202	10.12.67.67	TLSv1.2	473	Application Data
4463	10.692376	180.149.52.202	10.12.67.67	TLSv1.2	85	Application Data
594	14.274791	10.12.67.67	52.123.169.122	TLSv1.2	111	Application Data
6065	14.462652	52.123.169.122	10.12.67.67	TLSv1.2	100	Application Data
11198	27.511937	10.12.67.67	164.208.16.89	TLSv1.3	450	Client Hello (SNI=teams.events.data.microsoft.com)
11200	27.511937	164.208.16.89	10.12.67.67	TLSv1.3	153	Server Hello, Change Cipher Spec, Client Hello (SNI=teams.events.data.microsoft.com)
11257	27.809168	10.12.67.67	164.208.16.89	TLSv1.3	153	Server Hello, (SNI=teams.events.data.microsoft.com)
11258	27.810895	10.12.67.67	164.208.16.89	TLSv1.3	736	Change Cipher Spec, Client Hello (SNI=teams.events.data.microsoft.com)
11360	28.138141	164.208.16.89	10.12.67.67	TLSv1.3	153	HelloRetryRequest, Change Cipher Spec
11362	28.138512	164.208.16.89	10.12.67.67	TLSv1.3	1514	ServerHello
11364	28.138764	164.208.16.89	10.12.67.67	TLSv1.3	620	Application Data
11367	28.142488	10.12.67.67	164.208.16.89	TLSv1.3	673	ChangeCipherSpec, ClientHello (SNI=teams.events.data.microsoft.com)
11368	28.143960	10.12.67.67	164.208.16.89	TLSv1.3	128	Application Data
11369	28.144439	10.12.67.67	164.208.16.89	TLSv1.3	146	Application Data

2. IP Address Filters

- **ip.addr == 192.168.1.1 – Shows all traffic to/from a specific IP**

No.	Time	Source	Destination	Protocol	Length	Info
2067	4.876874	10.12.67.67	10.2.1.68	DNS	83	Standard query 0x87d3 A www.msftconnecttest.com
2077	4.979806	10.2.1.68	10.12.67.67	DNS	27	Standard query response 0x87d3 A www.msftconnecttest.com CNAME www.msftncsi.com.edgesuite.net
2078	4.980285	10.12.67.67	96.17.168.107	TCP	66	54192 > 104 [SYN] Seq=0 Win=65535 MSS=1468 WS=256 SACK_PERM
2079	4.987406	96.17.168.107	10.12.67.67	TCP	54	54192 > 104 [SYN] Seq=0 Win=65535 MSS=1468 WS=256 SACK_PERM
2080	4.987561	10.12.67.67	96.17.168.107	TCP	54	54192 > 104 [ACK] Seq=1 Win=65280 Len=0
2081	4.988983	10.12.67.67	96.17.168.107	HTTP	165	GET /connecttest.txt HTTP/1.1
2082	5.006698	96.17.168.107	10.12.67.67	HTTP	60	80 > 54192 [ACK] Seq=112 Win=64218 Len=0
2083	5.006700	96.17.168.107	10.12.67.67	HTTP	243	404 (Not Found) [Text/Plain]
2084	5.006698	96.17.168.107	10.12.67.67	HTTP	60	80 > 54192 [FIN, ACK] Seq=119 Win=65280 Len=0
2085	5.007081	10.12.67.67	96.17.168.107	TCP	54	54192 > 80 [ACK] Seq=112 Ack=189 Win=65280 Len=0
2086	5.007546	10.12.67.67	96.17.168.107	TCP	54	54192 > 80 [FIN, ACK] Seq=112 Ack=189 Win=0 Len=0
2127	5.032751	96.17.168.107	10.12.67.67	TCP	60	80 > 54192 [ACK] Seq=189 Ack=113 Win=64256 Len=0
2324	5.721639	10.12.67.67	52.123.170.79	TLSv1.2	104	Application Data
2458	5.943378	10.12.67.67	52.123.170.79	TLSv1.2	104	Application Data
2581	6.265407	10.12.67.67	4.213.25.240	TLSv1.2	154	Application Data
2582	6.312669	4.213.25.240	10.12.67.67	TLSv1.2	224	Application Data
2587	6.358203	10.12.67.67	4.213.25.240	TCP	54	54192 > 443 [ACK] Seq=101 Ack=171 Win=251 Len=0
2779	6.659426	10.12.67.67	148.113.9.41	TCP	55	53352 > 443 [ACK] Seq=1 Ack=1 Win=255 Len=1
2781	6.696868	148.113.9.41	10.12.67.67	TCP	66	443 > 5352 [ACK] Seq=1 Ack=2 Win=255 Len=1 SIf=1 SRIf=2

- **ip.src == 192.168.1.1 – Traffic from a specific IP**

No.	Time	Source	Destination	Protocol	Length	Info
2067	4.876874	10.12.67.67	10.2.1.68	DNS	83	Standard query 0x87d3 A www.msftconnecttest.com
2978	4.973296	10.12.67.67	96.17.168.107	TCP	66	54192 > 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1468 WS=256 SACK_PERM
2080	4.987561	10.12.67.67	96.17.168.107	HTTP	165	GET /connecttest.txt HTTP/1.1
2081	4.988983	10.12.67.67	96.17.168.107	HTTP	60	80 > 54192 [ACK] Seq=112 Ack=189 Win=65280 Len=0
2086	5.007546	10.12.67.67	96.17.168.107	HTTP	54	54192 > 80 [FIN, ACK] Seq=112 Ack=189 Win=0 Len=0
2234	5.721639	10.12.67.67	52.123.170.79	TLSv1.2	104	Application Data
2458	5.943378	10.12.67.67	52.123.170.79	TLSv1.2	104	Application Data
2581	6.265407	10.12.67.67	4.213.25.240	TLSv1.2	154	Application Data
2587	6.358203	10.12.67.67	4.213.25.240	TCP	54	54192 > 443 [ACK] Seq=101 Ack=171 Win=251 Len=0
2779	6.659426	10.12.67.67	148.113.9.41	TCP	55	53352 > 443 [ACK] Seq=1 Ack=1 Win=255 Len=1
4432	10.524426	10.12.67.67	180.149.52.202	TCP	1438	54187 > 443 [ACK] Seq=Ack=1 Win=253 Len=1384 [TCP PDU reassembled in 4434]
4433	10.524426	10.12.67.67	180.149.52.202	TCP	1438	54187 > 443 [ACK] Seq=1385 Ack=1 Win=253 Len=1384 [TCP PDU reassembled in 4434]
4434	10.524426	10.12.67.67	180.149.52.202	TLSv1.2	1249	Application Data, Application Data, Application Data
4435	10.524426	10.12.67.67	180.149.52.202	TCP	54	54192 > 443 [ACK] Seq=Ack=1 Win=251 Len=0
4465	10.759815	10.12.67.67	180.149.52.202	TCP	54	54187 > 443 [ACK] Seq=1364 Ack=1 Win=253 Len=9
4466	10.739805	10.12.67.67	180.149.52.202	TCP	66	[TCP Dup ACK 4465:1] 54187 > 443 [ACK] Seq=3984 Ack=490 Win=252 Len=0 SLF=459 SRF=498
4663	11.248177	10.12.67.67	202.189.250.92	TCP	66	[TCP Retransmission] 54193 > 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1468 WS=256 SACK_PERM

- `ip.dst == 8.8.8.8` – Traffic to a specific IP

3. Port Filters

- `tcp.port == 80` – Shows HTTP traffic on port 80

No.	Time	Source	Destination	Protocol	Length Info
1	2078.4 9:37:206	10.12.67.67	96.17.168.107	TCP	66 54192 + 80 [SYN] Seq=0 Win=5535 Len=0 MSS=1460 WS=256 SACK_PERM
2079.4 9:37:206	96.17.168.107	10.12.67.67	TCP	66 80 + 54192 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128	
2080.4 9:37:206	10.12.67.67	96.17.168.107	TCP	54 54192 + 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0	
2081.4 9:38:093	10.12.67.67	96.17.168.107	HTTP	165 GET /connecttest.txt HTTP/1.1	
2082.4 9:38:093	10.12.67.67	96.17.168.107	TCP	66 80 + 54192 [FIN, ACK] Seq=188 Ack=112 Win=64128 Len=0	
2083.5 9:06:598	96.17.168.107	10.12.67.67	HTTP	241 HTTP/1.1 200 OK [text/plain]	
2084.5 9:06:598	96.17.168.107	10.12.67.67	TCP	68 80 + 54192 [FIN, ACK] Seq=188 Ack=112 Win=64256 Len=0	
2085.5 9:07:081	10.12.67.67	96.17.168.107	TCP	54 54192 + 80 [ACK] Seq=112 Ack=189 Win=65280 Len=0	
2086.5 9:07:546	10.12.67.67	96.17.168.107	TCP	54 54192 + 80 [FIN, ACK] Seq=112 Ack=189 Win=0 Len=0	
2127.5 9:08:551	96.17.168.107	10.12.67.67	TCP	66 80 + 54192 [ACK] Seq=114 Ack=113 Win=64256 Len=0	
2128.5 9:08:551	10.12.67.67	96.17.168.107	TCP	54 54192 + 80 [FIN, ACK] Seq=114 Ack=113 Win=64256 Len=0	
5312.12 8:44:47	10.12.67.67	202.189.250.92	TCP	66 [TCP Retransmission] 54194 + 80 [SYN] Seq=0 Win=5535 Len=0 MSS=1460 WS=256 SACK_PERM	
6344.12 8:44:513	10.12.67.67	202.189.250.92	TCP	66 [TCP Retransmission] 54194 + 80 [SYN] Seq=0 Win=5535 Len=0 MSS=1460 WS=256 SACK_PERM	
7999.12 8:49:414	10.12.67.67	202.189.250.92	TCP	66 [TCP Retransmission] 54194 + 80 [SYN] Seq=0 Win=5535 Len=0 MSS=1460 WS=256 SACK_PERM	
8102.12 8:51:183	10.12.67.67	202.189.250.92	TCP	66 80 + 54194 [SYN, ACK] Seq=0 Ack=1 Win=6535 Len=0 MSS=1460 WS=256 SACK_PERM	
8103.12 8:51:183	10.12.67.67	202.189.250.92	TCP	64 54194 + 80 [ACK] Seq=1 Ack=1 Win=6535 Len=0 MSS=1460 WS=256 SACK_PERM	
8108.12 8:54:467	10.12.67.67	202.189.250.92	HTTP	1100 POST "/hercules/submitDataInfo.cgi" HTTP/1.1 [application/x-www-form-urlencoded]	
8148.12 8:59:56	202.189.250.92	10.12.67.67	TCP	66 80 + 54194 [ACK] Seq=1141 Win=65280 Len=0	
22799.51 344:346	10.12.67.67	202.189.250.92	TCP	54 54194 + 80 [RST, ACK] Seq=1141 Ack=1 Win=0 Len=0	
23374.52 9:08:516	10.12.67.67	103.239.171.37	TCP	66 54230 + 80 [SYN] Seq=0 Win=5535 Len=0 MSS=1460 WS=256 SACK_PERM	
Frame 2078: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{34388C00-0000-0000-0000-000000000000					
Ethernet II, Src: AzureWind_bada:13 (50:50:65:bada:13), Dst: Hewlett-Packard_93:44:53 (54:77:8a:93:44:53)					
Internet Protocol Version 4, Src: 10.12.67.67, Dst: 96.17.168.107					
Transmission Control Protocol, Src Port: 54192, Dst Port: 80, Seq: 0, Len: 0					

- `udp.port == 53` – Shows DNS traffic over UDP

No.	Time	Source	Destination	Protocol	Length Info
2067	10.12.67.67	10.2.1.68	10.12.67.67	DNS	22 Standard query 0x97d3 A www.msftconnecttest.com
2077	10.12.67.66	10.2.1.68	10.12.67.67	DNS	22 Standard query 0x97d3 A www.msftconnecttest.com CNAME www.msftncsi.com.edgesuite.net
4276	10.18.6359	10.2.1.68	10.2.1.68	DNS	82 Standard query 0x533a A www.computermumbai.com
4277	10.2.1.68	10.2.1.68	10.12.67.67	DNS	98 Standard query response 0x933a A www.computermumbai.com A 202.189.250.92
4911	11.812453	10.2.1.68	10.2.1.68	DNS	83 Standard query 0x1669 A zvh.corpwebcontrol.com
4912	11.868186	10.2.1.68	10.12.67.67	DNS	99 Standard query response 0x1669 A zvh.corpwebcontrol.com A 202.189.250.92
11834	11.159519	10.2.1.68	10.2.1.68	DNS	91 Standard query 0x9f69 A teams.events.data.microsoft.com
11835	11.159519	10.2.1.68	10.2.1.68	DNS	92 Standard query response 0x9f69 HTTPS teams.events.data.microsoft.com
11836	27.158736	10.2.1.68	10.2.1.68	DNS	79 Standard query 0x9f69 A wpad.DDN.UPES.AC.IN
11037	27.201371	10.2.1.68	10.12.67.67	DNS	134 Standard query response 0x9f69 No such name A wpad.DDN.UPES.AC.IN SOA authddn.DDN.UPES.AC.IN
11038	27.219813	10.2.1.68	10.12.67.67	DNS	213 Standard query response 0x9f57 A teams.events.data.microsoft.com CNAME teams-events-data.trafficmanager.net CNAME onedscolprdcu...
11039	27.219816	10.2.1.68	10.12.67.67	DNS	91 Standard query response 0x9f57 HTTPS teams.events.data.microsoft.com
12443	10.12.67.67	10.2.1.68	10.2.1.68	DNS	73 Standard query 0x9e90 A w2l.corpwebcontrol.com
12507	31.434489	10.2.1.68	10.12.67.67	DNS	98 Standard query response 0x9e90a A w2l.npavnet.in A 146.112.61.106
12891	32.429629	10.2.1.68	10.2.1.68	DNS	71 Standard query 0xe2a9 A w4.npav.net
12976	32.496719	10.2.1.68	10.12.67.67	DNS	87 Standard query response 0xe2a9 A w4.npav.net A 15.235.218.105
13358	33.476552	10.2.1.68	10.12.67.67	DNS	82 Standard query 0xe0b85 A w11.corpwebcontrol.com
13430	34.476552	10.2.1.68	10.12.67.67	DNS	98 Standard query response 0xe0b85 A w11.corpwebcontrol.com A 150.242.201.76
13493	34.369865	10.2.1.68	10.2.1.68	DNS	71 Standard query 0x2798 A w5.npav.net A 148.113.17.82
13704	34.438345	10.2.1.68	10.12.67.67	DNS	87 Standard query response 0x2798 A w5.npav.net A 148.113.17.82

• tcp.srcport == 443 – HTTPS traffic from server

No.	Time	Source	Destination	Protocol	Length Info
2447	5.888155	4.213.170.79	10.12.67.67	TLSv1.2	93 Application Data
2582	6.312669	4.213.25.240	10.12.67.67	TLSv1.2	224 Application Data
2851	6.696866	148.113.9.41	10.12.67.67	TCP	66 443 > 53352 [ACK] Seq=1 Ack=2 Win=501 Len=0 SLE=1 SRE=2
4444	10.549137	180.149.52.202	10.12.67.67	TCP	66 [TCP Keep-Alive] 443 > 54189 [ACK] Seq=1 Ack=3964 Win=1105 Len=0
4445	10.549137	180.149.52.202	10.12.67.67	TCP	66 443 > 54189 [ACK] Seq=1 Ack=3964 Win=1105 Len=0
4446	10.549137	180.149.52.202	10.12.67.67	TLSv1.2	473 Application Data
4462	10.692376	180.149.52.202	10.12.67.67	TLSv1.2	473 Application Data
4463	10.692376	180.149.52.202	10.12.67.67	TLSv1.2	473 Application Data
4464	10.739803	180.149.52.202	10.12.67.67	TCP	85 [TCP Retransmission] 443 > 54187 [PSH, ACK] Seq=459 Ack=3964 Win=1105 Len=31
6063	14.270452	52.123.170.79	10.12.67.67	TLSv1.2	100 Application Data
7150	14.270452	148.113.9.41	10.12.67.67	TCP	66 [TCP Keep-Alive] 443 > 53352 [ACK] Seq=1 Ack=2 Win=501 Len=0 SLE=1 SRE=2
10753	26.783831	148.113.9.41	10.12.67.67	TCP	66 [TCP Keep-Alive] 443 > 53352 [ACK] Seq=1 Ack=2 Win=501 Len=0 SLE=1 SRE=2
10951	27.064594	180.149.52.218	10.12.67.67	TCP	66 443 > 54189 [ACK] Seq=1 Ack=2 Win=501 Len=0 SLE=1 SRE=2
11088	27.308232	20.189.173.1	10.12.67.67	TCP	66 443 > 54188 [ACK] Seq=1 Ack=2 Win=16384 Len=0 SLE=1 SRE=2
11195	27.508163	180.208.16.89	10.12.67.67	TCP	66 443 > 54189 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 WS=256 SACK_PERM
13300	27.508163	180.208.16.89	10.12.67.67	TCP	66 443 > 54189 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 WS=256 SACK_PERM
11254	27.800402	104.208.16.89	10.12.67.67	TLSv1.3	153 HelloRetryRequest, ChangeCipherSpec
11309	27.995642	104.208.16.89	10.12.67.67	TCP	60 443 > 54195 [ACK] Seq=109 Ack=2519 Win=4194560 Len=0
11368	28.138141	104.208.16.89	10.12.67.67	TLSv1.3	153 HelloRetryRequest, ChangeCipherSpec
13161	28.138141	104.208.16.89	10.12.67.67	TLSv1.3	1514 Server Hello

• tcp.dstport == 443 – HTTPS traffic to server

No.	Time	Source	Destination	Protocol	Length Info
2350	5.943038	10.12.67.67	52.123.170.79	TLSv1.2	38 Application Data
2459	5.943038	10.12.67.67	52.123.170.79	TCP	54 53373 > 443 [ACK] Seq=51 Ack=40 Win=25 Len=0
2581	6.265407	10.12.67.67	4.213.25.240	TLSv1.2	154 Application Data
2587	6.358203	10.12.67.67	4.213.25.240	TCP	54 49423 > 443 [ACK] Seq=101 Ack=171 Win=251 Len=0
2779	6.659426	10.12.67.67	148.113.9.41	TCP	55 53352 > 443 [ACK] Seq=1 Ack=1 Win=255 Len=1
4278	10.235702	10.12.67.67	20.189.173.1	TCP	66 443 > 54189 [ACK] Seq=103 Ack=1 Win=13400 Len=0
4432	10.235702	10.12.67.67	180.149.52.202	TCP	1438 54189 > 443 [ACK] Seq=103 Ack=1 Win=13400 Len=0
4443	10.235702	10.12.67.67	180.149.52.202	TCP	1438 54187 > 443 [ACK] Seq=1385 Ack=2535 Win=1384 [TCP PDU reassembled in 4434]
4444	10.235702	10.12.67.67	180.149.52.202	TLSv1.2	1249 Application Data, Application Data, Application Data
4457	10.596864	10.12.67.67	180.149.52.202	TCP	54 54187 > 443 [ACK] Seq=3964 Ack=40 Win=253 Len=0
4465	10.739815	10.12.67.67	180.149.52.202	TCP	54 54187 > 443 [ACK] Seq=3964 Ack=490 Win=252 Len=0
4466	10.739865	10.12.67.67	180.149.52.202	TCP	66 [TCP Dup ACK 4465#1] 54187 > 443 [ACK] Seq=3964 Ack=490 Win=252 Len=0
4683	10.739815	10.12.67.67	20.189.173.1	TCP	66 [TCP Retransmission] 54187 > 443 [ACK] Seq=3964 Ack=490 Win=252 Len=0
5353	13.250206	10.12.67.67	20.189.173.1	TCP	54 54193 > 443 [SYN, ACK] Seq=0 Ack=0 Win=65535 Len=0
5945	14.274791	10.12.67.67	52.123.168.122	TLSv1.2	311 Application Data
6064	14.518798	10.12.67.67	52.123.168.122	TCP	54 53720 > 443 [ACK] Seq=47 Win=254 Len=0
7147	14.701978	10.12.67.67	148.113.9.41	TCP	55 [TCP Keep-Alive] 53352 > 443 [ACK] Seq=1 Ack=1 Win=255 Len=1
7482	17.258647	10.12.67.67	202.189.250.93	TCP	66 [TCP Retransmission] 54187 > 443 [SYN, ACK] Seq=0 Ack=0 Win=65535 Len=0
7500	17.258647	10.12.67.67	202.189.250.93	TCP	66 [TCP Retransmission] 54187 > 443 [SYN, ACK] Seq=0 Ack=0 Win=65535 Len=0
10741	26.738182	10.12.67.67	148.113.9.41	TCP	55 [TCP Keep-Alive] 53352 > 443 [ACK] Seq=1 Ack=1 Win=255 Len=1

4. MAC Address Filters

• eth.addr == 00:11:22:33:44:55 – Traffic to/from a specific MAC address

No.	Time	Source	Destination	Protocol	Length Info
2067	4.570674	10.12.67.67	10.2.1.68	DNS	83 Standard query 0x87d3 A www.msftconnecttest.com
2077	4.570666	10.2.1.68	10.12.67.67	DNS	227 Standard query response 0x87d3 A www.msftconnecttest.com CNAME ncsi-geo.trafficmanager.net CNAME www.msftncsi.com.edgesuite.net
2078	4.573026	10.12.67.67	96.17.168.107	TCP	66 54192 > 80 [SYN] Seq=0 Win=5535 Len=0 MSS=1460 WS=256 SACK_PERM
2079	4.574046	96.17.168.107	10.12.67.67	TCP	66 80 > 54192 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
2080	4.574051	10.12.67.67	96.17.168.107	TCP	54 54192 > 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0
2081	4.588983	10.12.67.67	96.17.168.107	HTTP	165 GET /Connecttest.txt HTTP/1.1
2082	5.066698	96.17.168.107	10.12.67.67	TCP	60 80 > 54192 [ACK] Seq=1 Ack=112 Win=64128 Len=0
2083	5.066698	96.17.168.107	10.12.67.67	HTTP	241 HTTP/1.1 200 OK (text/plain)
2084	5.066698	96.17.168.107	10.12.67.67	TCP	60 54192 > 80 [FIN, ACK] Seq=122 Ack=112 Win=64256 Len=0
2085	5.067081	10.12.67.67	96.17.168.107	TCP	54 54192 > 80 [ACK] Seq=112 Ack=119 Win=65280 Len=0
2086	5.067546	10.12.67.67	96.17.168.107	TCP	54 54192 > 80 [FIN, ACK] Seq=122 Ack=119 Win=0 Len=0
2127	5.932751	96.17.168.107	10.12.67.67	TCP	60 80 > 54192 [ACK] Seq=189 Ack=113 Win=64256 Len=0
2324	5.721635	10.12.67.67	52.123.170.79	TLSv1.2	104 Application Data
2447	5.843003	52.123.170.79	10.12.67.67	TLSv1.2	104 Application Data
2454	5.843078	10.12.67.67	52.123.170.79	TLSv1.2	104 Application Data Seq=51 Ack=40 Win=252 Len=0
2581	6.265487	10.12.67.67	4.213.25.240	TLSv1.2	154 Application Data
2582	6.312669	4.213.25.240	10.12.67.67	TLSv1.2	224 Application Data
2587	6.358203	10.12.67.67	4.213.25.240	TCP	54 49423 > 443 [ACK] Seq=101 Ack=171 Win=251 Len=0
2779	6.659426	10.12.67.67	148.113.9.41	TCP	55 53552 > 443 [ACK] Seq=102 Ack=171 Win=251 Len=0
2801	6.659426	148.113.9.41	10.12.67.67	TCP	66 645 > 53552 [ACK] Seq=103 Ack=171 Win=501 Len=1 STE=1 SRE=2
F	From: 10.12.67.67	To: 148.113.9.41	(664 bits)	83 bytes captured	83 bytes received (664 bits) on interface Device#0 IEEE802.11 wireless
E	Ethernet II, Src: Arduinokw [00:5e:65:b2:d1:13], Dst: HewlettPacke_93:44:53 (54:77:8a:93:44:53)	Internet Protocol Version 4, Src: 10.12.67.67, Dst: 10.2.1.60	0010 00 00 01 45 60 00 00 08 11 f9 ba 0a 9c 43 d3 03 02 E .. CC ..		
E	User Datagram Protocol, Src Port: 59534, Dst Port: 53	Domain Name System (query)	0020 00 01 c3 e8 8e 00 35 00 31 25 4f 87 d3 01 00 00 01 <---- 5 30 ..		
E			0030 00 00 00 00 00 00 03 77 77 0f 6d 73 66 74 63 <---- www.msftc ..		
E			0040 00 00 00 00 00 00 00 75 74 65 65 74 74 63 <---- www.msftc ..		
E			0050 01 00 01 <---- www.msftc ..		

- **eth.src == 00:11:22:33:44:55** – Traffic from MAC

No.	Time	Source	Destination	Protocol	Length Info
2067	10.12.67.67	10.12.2.1.68	DNS	83 Standard query 0x87d A www.msftconnecttest.com	
2068	10.12.67.67	96.17.168.187	TCP	54 54192 + 80 [ACK] Seq=112 Win=65280 Len=0	SACK_PERM
2069	10.12.67.67	96.17.168.187	TCP	54 54192 + 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0	
2081	10.12.67.67	96.17.168.187	HTTP	165 GET /connectstt.txt HTTP/1.1	
2085	10.12.67.67	96.17.168.187	TCP	54 54192 + 80 [ACK] Seq=112 Ack=189 Win=65280 Len=0	
2086	10.12.67.67	96.17.168.187	TCP	54 54192 + 80 [ACK] Seq=112 Ack=189 Win=65280 Len=0	
2324	10.12.67.67	52.123.170.179	TLSv1.2	104 Application Data	
2458	10.12.67.67	52.123.170.179	TCP	53 53717 + 443 [ACK] Seq=51 Ack=40 Win=252 Len=0	
6188	10.12.67.67	4.213.25.248	TLSv1.2	154 Application Data	
2309	10.12.67.67	4.213.25.248	TCP	53 53352 + 443 [ACK] Seq=101 Ack=171 Win=251 Len=0	
2779	10.12.67.67	148.113.9.41	TCP	53 53352 + 443 [ACK] Seq=1 Ack=1 Win=255 Len=1	
4276	10.12.67.67	10.2.1.68	DNS	82 Standard query 0x533a A www.computerumbali.com	
4280	10.12.67.67	10.2.1.68	DNS	80 Standard query 0x533a A www.computerumbali.com	SACK_PERM
4432	10.12.67.67	188.149.52.202	TCP	1438 54187 + 443 [ACK] Seq=1 Ack=1 Win=253 Len=1384 [TCP PDU reassembled in 4434]	
4433	10.12.67.67	188.149.52.202	TCP	1438 54187 + 443 [ACK] Seq=1385 Ack=1 Win=253 Len=1384 [TCP PDU reassembled in 4434]	
4434	10.12.67.67	188.149.52.202	TLSv1.2	1429 Application Data Application Data Application Data	
4465	10.12.67.67	188.149.52.202	TCP	54 54187 + 443 [ACK] Seq=3964 Ack=499 Win=252 Len=0	
4466	10.12.67.67	188.149.52.202	TCP	66 [TCP Dup ACK 4465@51487 + 443 [ACK] Seq=3964 Ack=499 Win=252 Len=0 SLE=459 SRE=409	
4663	10.24.67.67	202.189.250.1	HTTP	66 [TCP Retransmission] 51487 + 443 [SEQ] Seq=0 Win=65535 Len=0 NSP=1460 WS=256 SACK_PERM	
Ethernet II, Src: Arunreddy (00:0c:29:b1:d1:19), Dst: HewlettPacke_93:44:53 (54:77:08:93:44:53) [ethernet header length: 14 bytes]					
Internet Protocol Version 4, Src: 10.12.67.67, Dst: 10.2.1.68					
User Datagram Protocol, Src Port: 59534, Dst Port: 53					
Domain Name System (query)					
0050 01 00 01					

- `eth.dst == 00:11:22:33:44:55` – Traffic to MAC

5. Network Filters

- `ip.addr == 192.168.0.0/24` – Traffic within a subnet

No.	Time	Source	Destination	Protocol	Length Info
→ 2067 4.378674	10.12.67.67	96.17.168.107	10.2.1.68	DNS	83 Standard query 0x87d3 A www.msftncsi.com test.com
→ 2068 4.378206	10.12.67.67	96.17.168.107	10.2.1.68	DNS	66 54192 + 80 [SYN] Seq=0 Win=6535 Len=80 MSS=1460 WS=256 SACK_PERM
2079 4.397406	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=80 MSS=1460 WS=128
2080 4.398561	10.12.67.67	96.17.168.107	10.2.1.68	TCP	54 54192 + 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0
2081 4.398983	10.12.67.67	96.17.168.107	10.2.1.68	HTTP	165 GET /connecttest.txt HTTP/1.1
2082 5.006698	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [ACK] Seq=1 Ack=12 Win=64128 Len=0
2083 5.006698	10.12.67.67	96.17.168.107	10.2.1.68	TCP	284 54192 + 80 [ACK] Seq=12 Ack=12 Win=64128 Len=0
2084 5.006698	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [FIN, ACK] Seq=188 Ack=112 Win=64256 Len=0
2085 5.007081	10.12.67.67	96.17.168.107	10.2.1.68	TCP	54 54192 + 80 [ACK] Seq=112 Ack=189 Win=65280 Len=0
2086 5.007546	10.12.67.67	96.17.168.107	10.2.1.68	TCP	54 54192 + 80 [FIN, ACK] Seq=112 Ack=189 Win=8 Len=0
2127 5.932751	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [ACK] Seq=189 Ack=113 Win=64256 Len=0
2324 5.721655	10.12.67.67	96.17.168.107	10.2.1.68	TLSv1.2	52.123 170.79 227 bytes on wire (1816 bits), 227 bytes captured (1816 bits) on interface <i>DeviceWMP_34388C</i> (0x0000:00:00:00:00:00)
2447 5.932751	96.17.168.107	10.12.67.67	10.2.1.68	TLSv1.2	52.123 170.79 227 bytes on wire (1816 bits), 227 bytes captured (1816 bits) on interface <i>DeviceWMP_34388C</i> (0x0000:00:00:00:00:00)
2458 5.943378	10.12.67.67	96.17.168.107	10.2.1.68	TCP	52.123 170.79 227 bytes on wire (1816 bits), 227 bytes captured (1816 bits) on interface <i>DeviceWMP_34388C</i> (0x0000:00:00:00:00:00)
2581 6.265407	10.12.67.67	4.213.25.249	10.2.1.68	TCP	54 5317 + 443 [ACK] Seq=51 Ack=40 Win=252 Len=0
2582 6.312669	4.213.25.249	10.12.67.67	10.2.1.68	TLSv1.2	224 Application Data
2587 6.358203	10.12.67.67	4.213.25.249	10.2.1.68	TCP	54 49423 + 443 [ACK] Seq=101 Ack=171 Win=251 Len=0
2779 6.659426	10.12.67.67	148.113.9.41	10.2.1.68	TCP	55 5352 + 443 [ACK] Seq=1 Ack=1 Win=255 Len=1
2780 6.659426	10.12.67.67	148.113.9.41	10.2.1.68	TCP	66 443 + 53352 [ACK] Seq=1 Ack=2 Win=550 Len=0
2781 6.659426	10.12.67.67	148.113.9.41	10.2.1.68	TCP	66 443 + 53352 [ACK] Seq=1 Ack=2 Win=550 Len=0

6. TCP Flags

- **tcp.flags.syn == 1 – SYN packets (start of TCP handshake)**

No.	Time	Source	Destination	Protocol	Length Info
→ 2078 4.973206	10.12.67.67	96.17.168.107	10.2.1.68	TCP	66 54192 + 80 [SYN] Seq=0 Win=6535 Len=80 MSS=1460 WS=256 SACK_PERM
2079 4.987406	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=80 MSS=1460 WS=128
→ 2078 4.973206	10.12.67.67	96.17.168.107	10.2.1.68	TCP	66 54192 + 80 [SYN] Seq=0 Win=6535 Len=80 MSS=1460 WS=256 SACK_PERM
2080 4.987561	10.12.67.67	96.17.168.107	10.2.1.68	TCP	54 54192 + 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0
2081 4.988983	10.12.67.67	96.17.168.107	10.2.1.68	TCP	165 GET /connecttest.txt HTTP/1.1
2082 5.006698	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [ACK] Seq=1 Ack=12 Win=64128 Len=0
2083 5.006698	10.12.67.67	96.17.168.107	10.2.1.68	TCP	284 54192 + 80 [ACK] Seq=12 Ack=12 Win=64128 Len=0
2084 5.006698	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [FIN, ACK] Seq=188 Ack=112 Win=64256 Len=0
2085 5.007081	10.12.67.67	96.17.168.107	10.2.1.68	TCP	54 54192 + 80 [ACK] Seq=112 Ack=189 Win=65280 Len=0
2086 5.007546	10.12.67.67	96.17.168.107	10.2.1.68	TCP	54 54192 + 80 [FIN, ACK] Seq=112 Ack=189 Win=8 Len=0
2127 5.932751	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [ACK] Seq=189 Ack=113 Win=64256 Len=0
2324 5.721655	10.12.67.67	96.17.168.107	10.2.1.68	TLSv1.2	52.123 170.79 227 bytes on wire (1816 bits), 227 bytes captured (1816 bits) on interface <i>DeviceWMP_34388C</i> (0x0000:00:00:00:00:00)
2447 5.932751	96.17.168.107	10.12.67.67	10.2.1.68	TLSv1.2	52.123 170.79 227 bytes on wire (1816 bits), 227 bytes captured (1816 bits) on interface <i>DeviceWMP_34388C</i> (0x0000:00:00:00:00:00)
2458 5.943378	10.12.67.67	96.17.168.107	10.2.1.68	TCP	52.123 170.79 227 bytes on wire (1816 bits), 227 bytes captured (1816 bits) on interface <i>DeviceWMP_34388C</i> (0x0000:00:00:00:00:00)
2581 6.265407	10.12.67.67	4.213.25.249	10.2.1.68	TCP	54 5317 + 443 [ACK] Seq=51 Ack=40 Win=252 Len=0
2582 6.312669	4.213.25.249	10.12.67.67	10.2.1.68	TLSv1.2	224 Application Data
2587 6.358203	10.12.67.67	4.213.25.249	10.2.1.68	TCP	54 49423 + 443 [ACK] Seq=101 Ack=171 Win=251 Len=0
2779 6.659426	10.12.67.67	148.113.9.41	10.2.1.68	TCP	55 5352 + 443 [ACK] Seq=1 Ack=1 Win=255 Len=1
2780 6.659426	10.12.67.67	148.113.9.41	10.2.1.68	TCP	66 443 + 53352 [ACK] Seq=1 Ack=2 Win=550 Len=0
2781 6.659426	10.12.67.67	148.113.9.41	10.2.1.68	TCP	66 443 + 53352 [ACK] Seq=1 Ack=2 Win=550 Len=0

- **tcp.flags.fin == 1 – FIN packets (end of session)**

No.	Time	Source	Destination	Protocol	Length Info
→ 2078 4.973206	10.12.67.67	96.17.168.107	10.2.1.68	TCP	66 54192 + 80 [SYN] Seq=0 Win=6535 Len=80 MSS=1460 WS=256 SACK_PERM
2079 4.987406	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=80 MSS=1460 WS=128
→ 2078 4.973206	10.12.67.67	96.17.168.107	10.2.1.68	TCP	66 54192 + 80 [SYN] Seq=0 Win=6535 Len=80 MSS=1460 WS=256 SACK_PERM
2080 4.987561	10.12.67.67	96.17.168.107	10.2.1.68	TCP	54 54192 + 80 [ACK] Seq=1 Ack=1 Win=65280 Len=0
2081 4.988983	10.12.67.67	96.17.168.107	10.2.1.68	TCP	165 GET /connecttest.txt HTTP/1.1
2082 5.006698	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [ACK] Seq=1 Ack=12 Win=64128 Len=0
2083 5.006698	10.12.67.67	96.17.168.107	10.2.1.68	TCP	284 54192 + 80 [ACK] Seq=12 Ack=12 Win=64128 Len=0
2084 5.006698	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [FIN, ACK] Seq=188 Ack=112 Win=64256 Len=0
2085 5.007081	10.12.67.67	96.17.168.107	10.2.1.68	TCP	54 54192 + 80 [ACK] Seq=112 Ack=189 Win=65280 Len=0
2086 5.007546	10.12.67.67	96.17.168.107	10.2.1.68	TCP	54 54192 + 80 [FIN, ACK] Seq=112 Ack=189 Win=8 Len=0
2127 5.932751	96.17.168.107	10.12.67.67	10.2.1.68	TCP	66 80 + 54192 [ACK] Seq=189 Ack=113 Win=64256 Len=0
2324 5.721655	10.12.67.67	96.17.168.107	10.2.1.68	TLSv1.2	52.123 170.79 227 bytes on wire (1816 bits), 227 bytes captured (1816 bits) on interface <i>DeviceWMP_34388C</i> (0x0000:00:00:00:00:00)
2447 5.932751	96.17.168.107	10.12.67.67	10.2.1.68	TLSv1.2	52.123 170.79 227 bytes on wire (1816 bits), 227 bytes captured (1816 bits) on interface <i>DeviceWMP_34388C</i> (0x0000:00:00:00:00:00)
2458 5.943378	10.12.67.67	96.17.168.107	10.2.1.68	TCP	52.123 170.79 227 bytes on wire (1816 bits), 227 bytes captured (1816 bits) on interface <i>DeviceWMP_34388C</i> (0x0000:00:00:00:00:00)
2581 6.265407	10.12.67.67	4.213.25.249	10.2.1.68	TCP	54 5317 + 443 [ACK] Seq=51 Ack=40 Win=252 Len=0
2582 6.312669	4.213.25.249	10.12.67.67	10.2.1.68	TCP	165 GET /connecttest.txt HTTP/1.1
2587 6.358203	10.12.67.67	4.213.25.249	10.2.1.68	TCP	54 49423 + 443 [ACK] Seq=101 Ack=171 Win=251 Len=0
2779 6.659426	10.12.67.67	148.113.9.41	10.2.1.68	TCP	55 5352 + 443 [ACK] Seq=1 Ack=1 Win=255 Len=1
2780 6.659426	10.12.67.67	148.113.9.41	10.2.1.68	TCP	66 443 + 53352 [ACK] Seq=1 Ack=2 Win=550 Len=0
2781 6.659426	10.12.67.67	148.113.9.41	10.2.1.68	TCP	66 443 + 53352 [ACK] Seq=1 Ack=2 Win=550 Len=0

- **tcp.flags.reset == 1 – RST packets (reset connection)**

No.	Time	Source	Destination	Protocol	Length Info
→ 16397 3.511699	10.12.67.67	96.17.168.107	10.2.1.68	TCP	60 80 + 54192 [FIN, ACK] Seq=188 Ack=112 Win=64256 Len=0
2086 5.007561	96.17.168.107	10.12.67.67	10.2.1.68	TCP	54 54192 + 80 [SYN] Seq=0 Win=6535 Len=80 MSS=1460 WS=256 SACK_PERM
13357 33.468372	10.12.67.67	15.235.218.105	10.2.1.68	TCP	66 54198 + 443 [FIN, ACK] Seq=189 Ack=113 Win=64240 Len=80 MSS=1460 WS=256 SACK_PERM
13358 33.468372	15.235.218.105	10.12.67.67	10.2.1.68	TCP	66 80 + 54198 [SYN] Seq=0 Win=6535 Len=80 MSS=1460 WS=256 SACK_PERM
13668 34.351049	10.12.67.67	159.242.67.76	10.2.1.68	TCP	66 54199 + 443 [FIN, ACK] Seq=98 Ack=975 Win=64240 Len=0
13706 34.508472	150.242.261.76	10.12.67.67	10.2.1.68	TCP	66 443 + 54199 [FIN, ACK] Seq=8755 Ack=995 Win=261888 Len=0
16232 39.391741	10.12.67.67	189.173.79	10.2.1.68	TCP	54 54213 + 443 [FIN, ACK] Seq=2511 Ack=106 Win=65280 Len=0
16233 39.391741	189.173.79	10.12.67.67	10.2.1.68	TCP	66 443 + 54213 [FIN, ACK] Seq=2511 Ack=106 Win=65280 Len=0
16234 39.391741	10.12.67.67	189.173.79	10.2.1.68	TCP	54 54213 + 443 [FIN, ACK] Seq=2511 Ack=106 Win=65280 Len=0
16235 39.391741	189.173.79	10.12.67.67	10.2.1.68	TCP	66 443 + 54213 [FIN, ACK] Seq=2511 Ack=106 Win=65280 Len=0
16236 39.391741	10.12.67.67	189.173.79	10.2.1.68	TCP	54 54213 + 443 [FIN, ACK] Seq=2511 Ack=106 Win=65280 Len=0
16237 39.391741	189.173.79	10.12.67.67	10.2.1.68	TCP	66 443 + 54213 [FIN, ACK] Seq=2511 Ack=106 Win=65280 Len=0
19226 43.268768	10.12.67.67	148.113.17.82	10.2.1.68	TCP	54 54225 + 443 [FIN, ACK] Seq=1865 Ack=1048 Win=24246 Len=0
19113 43.511367	148.113.17.82	10.12.67.67	10.2.1.68	TCP	66 443 + 54225 [FIN, ACK] Seq=1865 Ack=1048 Win=24246 Len=0
21543 48.882111	189.169.52.292	10.12.67.67	10.2.1.68	TCP	66 443 + 54209 [FIN, ACK] Seq=599 Win=64128 Len=0
21544 48.882111	10.12.67.67	189.169.52.292	10.2.1.68	TCP	66 443 + 54209 [FIN, ACK] Seq=599 Win=64128 Len=0
21586 49.011381	10.12.67.67	65.21.154.164	10.2.1.68	TCP	66 443 + 54228 [FIN, ACK] Seq=973 Ack=13710 Win=262144 Len=0
21873 49.647240	65.21.154.164	10.12.67.67	10.2.1.68	TCP	66 443 + 54228 [FIN, ACK] Seq=973 Ack=13710 Win=261632 Len=0
24676 56.306740	10.12.67.67	10.12.67.67	10.2.1.68	TCP	66 443 + 54209 [FIN, ACK] Seq=1000 Win=0 Len=0
50549 132.772332	208.79.197.222	10.12.67.67	10.2.1.68	TCP	54 54243 + 443 [FIN, ACK] Seq=5499 Ack=3268 Win=0 Len=0
50550 132.772332	208.79.197.222	10.12.67.67	10.2.1.68	TCP	66 443 +

7. Other Useful Filters

- frame contains "text" – Packets containing specific text

```
Frame 25130: 82 bytes captured (656 bits), 82 bytes transmitted (656 bits) on interface [DeviceNPF_34888BC] (monitored) at 10:21:59.000000000 UTC
  Internet Protocol Version 4, Src: 10.12.67.67, Dst: 10.2.1.60
  User Datagram Protocol, Src Port: 61828, Dst Port: 53
  > Domain Name System (query)
```

- `tcp.analysis.retransmission` – Shows retransmitted packets

- `tcp.analysis.flags` – Flags abnormal TCP behavior

- arp – Filters ARP requests/replies

No.	Time	Source	Destination	Protocol	Length Info
24439	55.766640	b4:c1:6d:57:4a:86	Broadcast	ARP	56 who has 10.12.1.1? Tell 10.12.13.124
24440	55.766648	9e:26:3b:98:b3:54	Broadcast	ARP	56 who has 10.12.3.26? Tell 10.12.14.12
24441	55.707209	3a:f4:ff:73:30:c1:e5	Broadcast	ARP	56 who has 10.12.1.1? Tell 10.12.20.82
24442	55.707209	b4:cc:e8:8b:58:d8	Broadcast	ARP	56 who has 10.12.30.165? Tell 10.12.41.46
24443	55.707209	b4:cc:e8:8b:58:d8	Broadcast	ARP	56 who has 10.12.35.31? Tell 10.12.41.46
24458	55.808614	5a:c1:ed:5b:09:73	Broadcast	ARP	56 who has 10.12.23.161? Tell 10.12.44.51
24459	55.808614	AzneWaveTec_e5:a1:..	Broadcast	ARP	56 who has 10.6.21.153? Tell 10.6.1.46
24460	55.808614	5a:c1:ed:5b:09:73	Broadcast	ARP	56 who has 10.6.21.153? Tell 10.6.1.46
24462	55.811198	16:1b:3:69:2c:1d:f3	Broadcast	ARP	56 who has 10.12.23.161? Tell 10.12.43.84
24482	55.908919	Intel_6:6:64:d7	Broadcast	ARP	56 who has 10.6.21.153? Tell 10.6.29.3
24557	56.032573	Apple_4:9:9c:07	Broadcast	ARP	56 who has 10.12.1.1? Tell 10.12.38.217
24563	56.113356	Intel_e:7:3e:16	Broadcast	ARP	56 who has 10.6.1.1? Tell 10.6.4.43
24564	56.113356	Intel_e:7:3e:16	Broadcast	ARP	56 who has 10.6.1.1? Tell 10.6.4.43
24567	56.115122	Intel_e:7:3e:16	Broadcast	ARP	56 who has 10.6.1.1? Tell 10.6.4.43
24568	56.115122	86:87:24:5f:ff:ef	Broadcast	ARP	56 who has 10.6.18.114? Tell 10.6.17.209
24569	56.115122	aa:61:29:16:44:25	Broadcast	ARP	56 who has 10.6.1.1? Tell 10.6.4.242
24571	56.115853	4e:a9:65:13:3a:f8	Broadcast	ARP	56 who has 10.12.1.1? Tell 10.12.29.46
24650	56.213869	2e:0:1c:1f:ff:ff	Broadcast	ARP	56 ARP Broadcast for 10.6.1.1
24650	56.213869	b4:c1:ed:5b:09:73	Broadcast	ARP	56 who has 10.12.1.1? Tell 10.12.41.46
24653	56.213869	4e:a9:65:13:3a:f8	Broadcast	ARP	56 ARP Announcements for 10.12.29.46
Frame 24482, 56 bytes on wire (448 bits), 56 bytes captured (448 bits) on interface 'Device\NPF_{343808C...}					
Ethernet II, Src: Intel_6:6:64:d7 (08:59:7a:6:64:d7), Dst: Broadcast (ff:ff:ff:ff:ff:ff)					
Address Resolution Protocol (request)					
00:00 08 00 06 04 00 01 98 59 7a e5 64 d7 08 06 00 01 .. Y z d ..					
00:00 08 00 06 04 00 01 98 59 7a e5 64 d7 0a 06 1d 03 .. Y z d ..					
00:00 00 00 00 00 00 00 0a 06 15 99 00 00 00 00 00 00 .. Y z d ..					
00:00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .. Y z d ..					

~~Set~~

Set-B

Q1 What is purpose of CRC in Error Detection?
Explain with

Q2 Write a program to calculate CRC for
the BitStream 100 100
with Poly 1011

Q1 \Rightarrow CRC (cyclic Redundancy check) is
a Method Designed to transmit Data over the
Internet. It does so by making the
BitStream Exactly Divisible by the Generator
Polynomial by padding the remainders of
the Division to the BitStream.

\Rightarrow The Layout is BitStream \rightarrow CRC

\Rightarrow The length of the CRC is Equal to the
Degree of the Polynomial

\Rightarrow It can only Detect Single Bit Errors.
for Ex. The CRC of 100100 with Generator
Poly 1011 is

$$\text{Generator Poly} = x^3 + x + 1$$

\therefore length of CRC = 3

Padded Message = 100100000

Binary Division

$$\begin{array}{r}
 1011 \quad | \quad 100100.000 \\
 \underline{1011} \\
 001000 \\
 \underline{0011} \\
 001100 \\
 \underline{1011} \\
 01110 \\
 \underline{1011} \\
 0101
 \end{array}$$

Remainder is 101

Transmitted Message = 100100.101

Consider this Message is Received with an Error (Single Bit)

Let Received Message = 100101.101

To check if error is present

Erroneous B.F.

we do XOR division with the Generator polynomial
If the Remainder is Non-zero then the Data is Erroneous.

$$\begin{array}{l}
 1 + x + x^3 = 1011 \text{ (Generator Polynomial)} \\
 100101.101 \\
 \underline{1011} \\
 001000 \\
 \underline{0011} \\
 001100 \\
 \underline{1011} \\
 01110 \\
 \underline{1011} \\
 0101
 \end{array}$$

XOR Division

$$1011 \mid 10010010$$

$$\overline{001000}$$

$$\overline{001010}$$

$$\overline{00011}$$

← Non-zero Remainder

∴ The Received B.T is Errored.

Code

```

def xor(a, b):
    result = []
    for i in range(len(a)):
        if a[i] == b[i]:
            result.append("0")
        else:
            result.append("1")
    return ''.join(result)

```

```

def div(divd, divs):
    c = len(divs)
    temp = divd[0:c]

```

```
while (c < len(divd)):
```

```
    if temp[0] == '1':
```

```
        temp = xor(temp, divs) + divd[c]
```

Else:

$$temp = \text{xor}('0' * c, temp) + \text{div}[c:]$$

$c += 1$

if $\text{temp}[0] == '1'$:
 $\text{temp} = \text{xor}(\text{temp}, \text{div})$

Else

$$\text{temp} = \text{xor}('0' * c, \text{div})$$

return temp

def Encoding(data, poly):

$$\text{Padded_Data} = \text{data} + '0'^{(\text{len}(\text{poly}) - 1)}$$

$$\text{remainders} = \text{div}(\text{data}, \text{poly})$$

$$\text{Encoded} = \text{data} + \text{remainders}$$

Print("Data is ", Data)

Print("CRC is ", Remainder)

Print("Encoded is ", Encoded)

(1) data = "100.100"
 Poly = "1011"

Encoding(data, Poly)

$$((\text{div})_{\text{nd}} >=) \text{ div}$$

$$(\text{div})_{\text{nd}} = \text{div}$$

$$(\text{div}_{\text{nd}})_{\text{rem}} = \text{rem}$$