Thadomal Shahani Engineering College

Bandra (W.), Mumbai- 400 050.

CERTIFICATE

Certify that Mr./Miss Parth Sandeep Dabholkar of Computer Department, Semester V with Roll No. 2103032 has completed a course of the necessary experiments in the subject Computer Networks under my supervision in the **Thadomal Shahani Engineering College** Laboratory in the year 2023 - 2024

19/10/23.

Teacher In-Charge

Head of the Department

Date 19/10/23

Principal

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

CONTENTS

SR. NO.	EXPERIMENTS	PAGE NO.	DATE	TEACHERS SIGN.
1.	Study of RJ45 and CAT6 cabling and connection using crimping.	1 - 4	19/07/23	
2.	Implementation of Hamming Code for Error Detection and Correction.	5 - 11	26/07/23	
3.	Implementation of CRC for Error detection	12 - 18	02/08/23	
4.	Simulation of Go-Back N Algorithm	19 - 23	09/08/23	
5.	Build simple n/w topology and configure it for static routing using packet tracer. Configure IP, subnetting and masking.	24 - 30	23/08/23	
6.	Design VPN and configure RIP using packet tracer.	31 - 40	06/09/23	
7.	WAP to implement IPv4 addressing along with subnet masking.	41 - 46	13/09/23	
8.	Use basic networking commands in Linux. (ping, tracert, nslookup, ip, ifconfig, dig)	47 - 55	27/09/23	
9.	Use wireshark to understand op of TCP/IP (Ethernet, Data-Link, Network, Transport and Application Layer)	56 - 61	27/09/23	
10.	Socket programming using TCP or UDP	62 - 65	04/10/23	
11.	Assignment 1	66 - 77	17/10/23	
12.	Assignment 2	78 - 83	17/10/23	

EXPERIMENT : 1

AIM: Study of RJ45 and CAT6 cabling and connection using crimping tool.

THEORY:

Tools and materials required are:

i) Ethernet cable
ii) RJ45 connectors
iii) Wire stripper
iv) Cable Tester
v) Crimping tool

Steps involved in crimping RJ45 are:

1) Prepare the Cable: Start by cutting the ethernet cable to the desired length, making sure it's long enough for your needs but not too long. Use the wire stripper to carefully remove about 1.5 inches of the outer jacket from the end of the cable. Be cautious not to cut into the inner wires while stripping.

2) Untwist and arrange wire: Inside the cable, you'll find four twisted pairs of wires. Untwist them gently without overstretching. Arrange the wires in the correct order for your chosen wiring standard. For a standard Ethernet cable using the T568B wiring scheme, the order from left to right should be: white-orange, orange, white-green, blue, white-blue, green, white-brown, brown.

3) Trim and align wires: Trim the wires so they are all even and flush with the front of the RJ45 connect

Use a wire cutter or scissors for this. Make sure the wires are in the correct order and properly align.

4) Insert wires into connector: Carefully insert the trimmed wires into the RJ45, ensuring they go all the way to the end of the connector and make contact with the metal pins inside.

5) Crimp the Connector: Place the RJ45 connector, with the wires inserted, into the crimping tool. Ensure the connector is seated properly in the tool's crimping tool's crimping chamber. Squeeze the crimping tool's chamber firmly and evenly.

6) Inspect the connection: Visually inspect the RJ45 connector to ensure all wires are still in the correct order and properly secured.

7) Repeat for the other end: If you are completing a ethernet cable, repeat the above steps on the other end of the cable.

8) Test the Cable: To ensure your cable is properly crimped and functional, you can use a cable tester.

* Cat 6 Cable: Category 6 is a standardized cable and connector type used in networking to transmit data at higher speeds and with better performance compared to its predecessors, such as Cat5.

Features of CAT6 :

1) Higher Data Transfer Rates
2) Enhanced Performance
3) Backward compatibility
4) Quality and shielding
5) Connector Compatibility

* The T568A and T568B are two different wiring schemes used for Ethernet cables, specifying the arrangement of wires within an RJ45 connector:

T568A :	T568B :
1) White/Green	1) White/orange
2) Green	2) Orange
3) White/orange	3) White/green
4) Blue	4) Blue
5) White/Blue	5) White/Blue
6) orange	6) Green
7) White/Brown	7) White/Brown
8) Brown	8) Brown.

* Standard Ethernet Connection :

1) Standard Ethernet cables are used for connecting different types of network devices, such as computers to switches or routers to switches
2) In a standard connection, both the ends of the cable use the same wiring scheme, either T568A or T568B.
3) The pin configuration on one end matches the pin configuration on other end.
4) These are used for regular data transmission.

* Crossover Ethernet connection:

1) Crossover cables are used for connecting similar types of networks devices directly to each other.
2) In a crossover Ethernet connection, the wiring scheme on one end is different from wiring scheme on the other end.
3) The pin configuration is crossover i.e. pin 1 on one end connects to pin 3 on the other end and vice versa.
4) These cables are used to establish a direct network connection between two devices of the same type.

CN-C12-32

EXPERIMENT : 2

AIM: Implementation of Hamming code for detection and correction.

THEORY : Hamming codes are a type of error correcting code used in digital communication systems to detect and correct transmitted errors in transmitted data. They are particularly useful for single bit error correction and error detection. Here's an overview of the Hamming code detection and correction method.

Hamming codes add redundant bits (parity bits) to the original data to create a codeword.
The number of parity bits is determined by the formula $2^p \geq p+m+1$, where m is the number of data bits, p is the number of parity bits.

1) Data encoding : For input message of 'k' bits, 'r' parity bits are added to the positions that are powers of 2 (1, 2, 4, 8, etc). The positions of these parity bits are calculated by using hamming representation. Each parity bit is responsible for checking results of data bit.

2) Parity calculation : Each parity bit calculates its value based on data bits that it carries. For eg: parity bit at position 1 checks all data bits that have a 1 in their least significant bits and so on. Parity bits are calculated using even parity, which means total numbers of 1s in group of bits should be even.

3) Error detection - During transmission if a bit gets flipped

due to noise, the parity bits will no longer match their calculated values. This discrepancy indicates an error.

4) Error correction: If an error is detected, the erroneous bits position can be determined by looking at parity bits that cover that position. By using the pattern of incorrect parity bits, the erroneous bits can be identified and corrected.

5) Decoding: The recieved data is compared to parity bits. If any discrepancies are found, error detection and correction are performed based on parity bit positions.

Its limitations, is it cannot perform correction for error in more than two bits or detect error in parity bits.

Problem: 7 bit hamming code is recieved as 1011011. Assume even parity and state whether the recieved code is correct or wrong, if wrong locate the bit in error.

Soln:- Recieved hc: D7 D6 D5 P4 D3 P2 P1
1 0 1 1 0 1 1

Detecting error: Analyzing bits 1, 3, 5, 7
we have P1 D3 D5 D7 ⇒ 1011 ∴ odd parity, error exists.
we put P1 = 1

Step 2: Analyzing 2, 3, 6, 7
P2 D3 D6 D7
1 0 0 1 ∴ even parity, no error exists.
⇒ P2 = 0

Step 3: Analyzing bits 4, 5, 6, 7

We have P_4, D_5, D_6, D_7

1 1 0 1 ← odd parity error exists

we put $P_4 = 1$

Correcting error: error word $E = P_4\ P_2\ P_1$

1 0 1

decimal value $\varepsilon = 5$ which shows that 5th bit is in error.

We write the correct word by simply inverting the 5th bit:-

$\therefore$ correct-word = 1001011

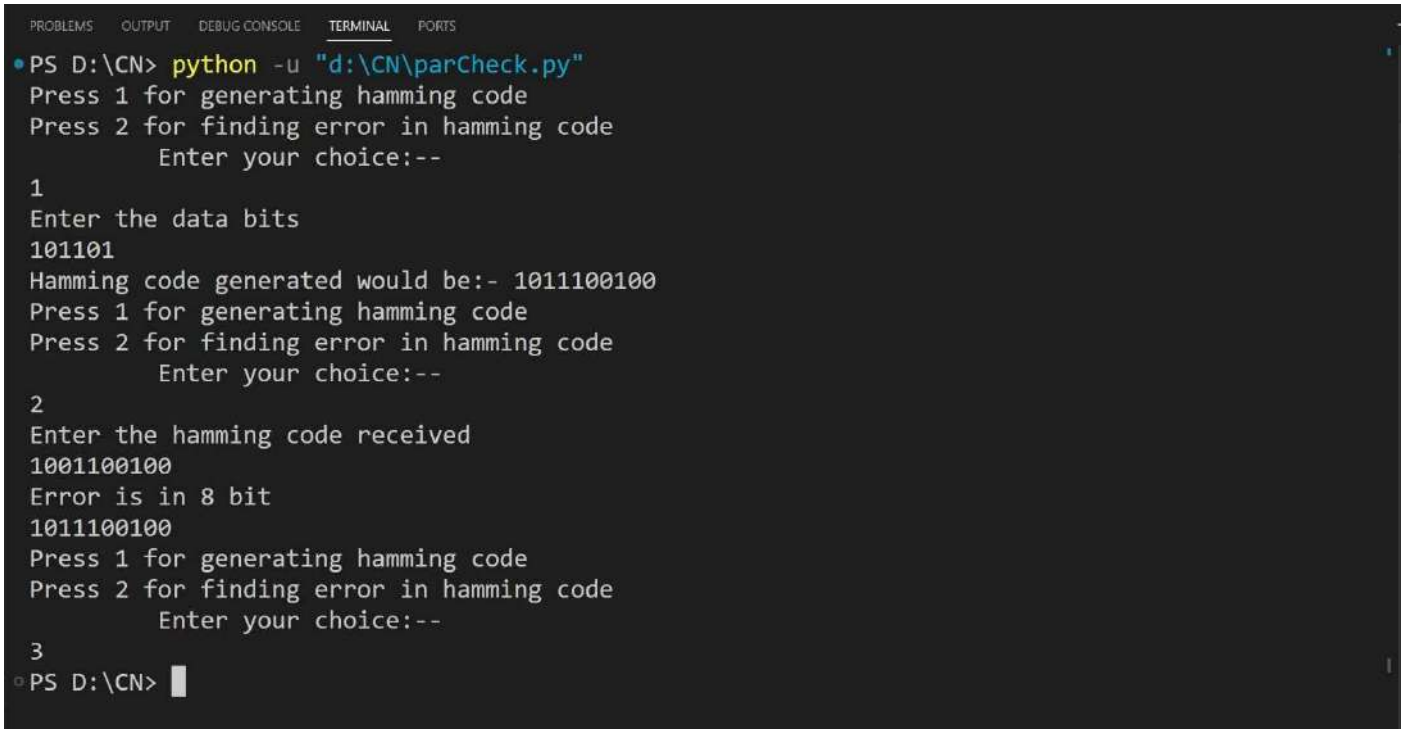
```
option=int(input('Press 1 for generating hamming code  \nPress 2 for finding error in hamming code\n\t Enter your choice:--\n'))
while(option>0 and option<3):  # GENERATE HAMMING CODE

   if(option == 1):
      print('Enter the data bits')
      d=input()
      data=list(d)
      data.reverse()
      c,ch,j,r,h=0,0,0,0,[]

      while ((len(d)+r+1)>(pow(2,r))):
         r=r+1

      for i in range(0,(r+len(data))):
         p=(2**c)

         if(p==(i+1)):
            h.append(0)
            c=c+1

         else:
            h.append(int(data[j]))
            j=j+1

      for parity in range(0,(len(h))):
         ph=(2**ch)
         if(ph==(parity+1)):
            startIndex=ph-1
            i=startIndex
            toXor=[]

            while(i<len(h)):
```

```
            block=h[i:i+ph]
            toXor.extend(block)
            i+=2*ph

        for z in range(1,len(toXor)):
            h[startIndex]=h[startIndex]^toXor[z]
        ch+=1

    h.reverse()
    print('Hamming code generated would be:- ', end="")
    print(int(''.join(map(str, h))))


elif(option==2): # DETECT ERROR IN RECEIVED HAMMING CODE
    print('Enter the hamming code received')
    d=input()
    data=list(d)
    data.reverse()
    c,ch,j,r,error,h,parity_list,h_copy=0,0,0,0,0,[],[],[]

    for k in range(0,len(data)):
        p=(2**c)
        h.append(int(data[k]))
        h_copy.append(data[k])
        if(p==(k+1)):
            c=c+1

    for parity in range(0,(len(h))):
        ph=(2**ch)
        if(ph==(parity+1)):

            startIndex=ph-1
```

```
        i=startIndex
        toXor=[]

        while(i<len(h)):
            block=h[i:i+ph]
            toXor.extend(block)
            i+=2*ph

        for z in range(1,len(toXor)):
            h[startIndex]=h[startIndex]^toXor[z]
        parity_list.append(h[parity])
        ch+=1
parity_list.reverse()
error=sum(int(parity_list) * (2 ** i) for i, parity_list in enumerate(parity_list[::-1]))

if((error)==0):
    print('There is no error in the hamming code received')

elif((error)>=len(h_copy)):
    print('Error cannot be detected')

else:
    print('Error is in',error,'bit')

    if(h_copy[error-1]=='0'):
        h_copy[error-1]='1'

    elif(h_copy[error-1]=='1'):
        h_copy[error-1]='0'
        print('After correction hamming code is:- ')
    h_copy.reverse()
    print(int(''.join(map(str, h_copy))))
```

```
    else:
        print('Option entered does not exist')


    option=int(input('Press 1 for generating hamming code  \nPress 2 for finding error in hamming code\n\t Enter your choice:--\n'))
```

OUTPUT



EXPERIMENT : 3

AIM: Implementation of CRC (Cyclic Redundancy Check)

THEORY: CRC is method of detecting error in communication channel. Given k bit frame or message, the transmitter generates a n-bit sequence known as frame check sequence (FCS), so that resulting frame, consisting of (k+n) bits.

- Bit sequences can be written as polynomials with coefficient 0 and 1.
- Frame with k bits is considered as polynomial of degree k-1.
- The most significant bit is coefficient of x^{k-1}. The next bit is coefficient of x^{k-2}. Eg: The bit sequence 10011010 corresponds to the polynomial
- Sending and receiving message can be imagined as exchange of polynomials.

$M(x) = 1 * x^7 + 0 * x^6 + x^5 + 1 * x^4 + 1 * x^3 + 0 * x^2 + x^1 + 0 x^0$

$= x^7 + x^4 + x^3 + x^1$

- The Data Link Layer protocol specifies a generator polynomial C(x). Generator polynomial is available for both sender and receiver side.

C(x) is a polynomial of degree k

Eg: $C(x) = x^3 + x^2 + x^0$

$= 1101$, then $k = 3$

The degree of generator polynomial is degree 3.
The degree of generator polynomial is equal to number of bits minus one.

If for a frame, the CRC need to be calculated, no. of bits are appended to the frame. 'n' corresponds to degree of generator polynomial.

Generator polynomial 10011 0

Generator polynomial has 4 digits. There 3 zero bits are appended.

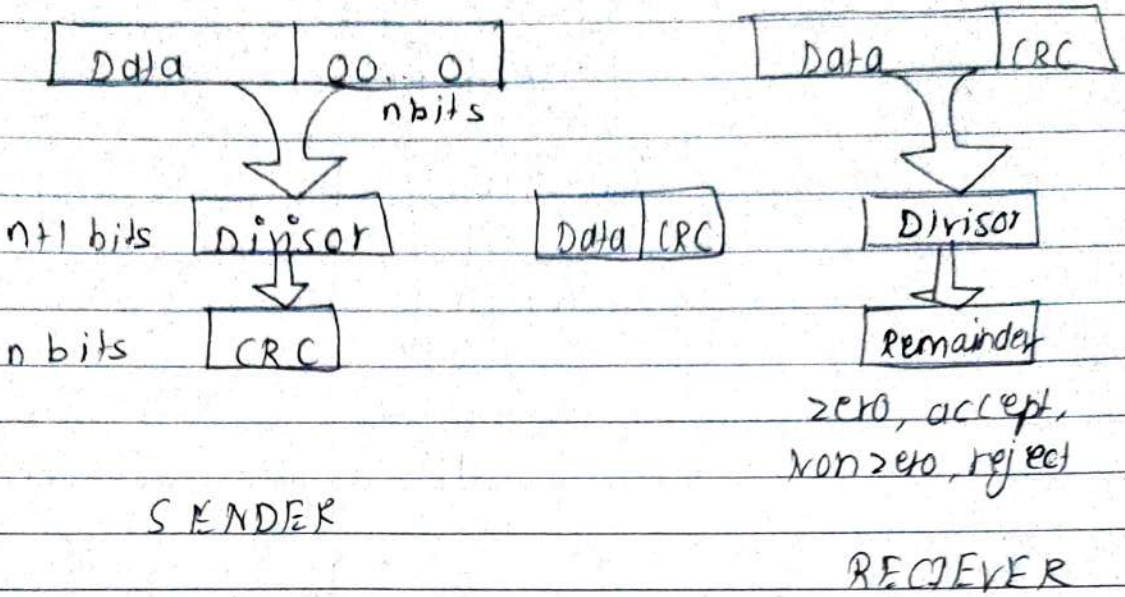
Frame with appended 0 bit 1010100000

* Sender Side (Generation of encoded Data from Data and Generator Polynomial (or Key))
- Binary data is first arrange/argumented by adding n zeroes in end of the data (n degree of generator polynomial).
- Use modulo-2 binary division to divide binary data by key and store remainder of division.
- Append remainder at end of the data to form encoded data and send the same.

* Reciever Side (check if there is any error introduced in transmission).

- Perform modulo -> division again if the remainder is 0, then there are no errors.

Diagram:



Eg:-

```
          1 1 1 1 0 1
      ---------------
1 1 0 1 ) 1 0 0 1 0 0 0 0 0
          1 1 0 1 ↓
          ---------
          0 1 0 0 0
            1 1 0 1 ↓
            ---------
            0 1 0 1 0
              1 1 0 1 ↓
              ---------
              0 1 1 1 0
                1 1 0 1 ↓ ↓
                ---------
                0 0 1 1 0 0
                  1 1 0 1
                  -------
                  0 0 0 1   → CRC
```

∴ Transmitted code: 100100001

Consider error in bit 5.

∴ Recieved code: 100110001

```
          1 1 1 1 1 1
      ┌──────────────
1101 ) 1 0 0 1 1 0 0 0 1
       1 1 0 1
       ---------
       0 1 0 0 1
         1 1 0 1
         ---------
         0 1 0 0 1
           1 1 0 1
           ---------
           0 1 0 0 0
             1 1 0 1
             ---------
             0 1 0 1 0
               1 1 0 1
               ---------
               0 1 1 1 1
                 1 1 0 1
                 ---------
                 0 0 1 0   → Remainder
                             not zero.
```

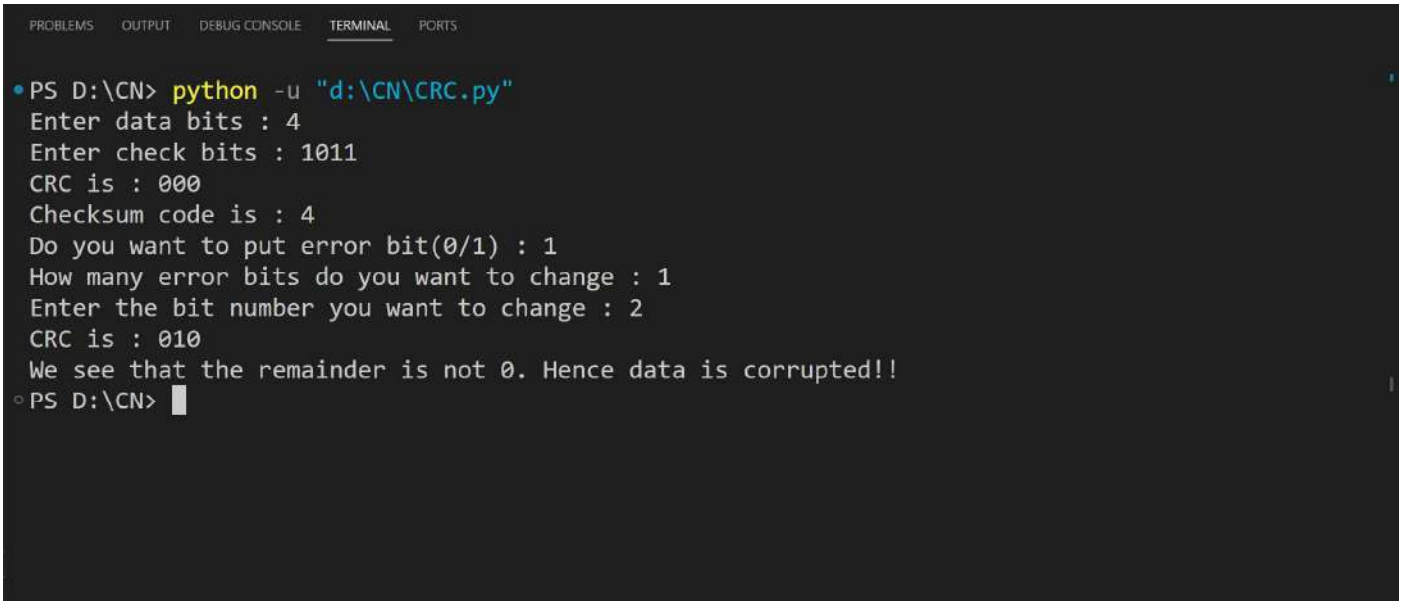
∴ Error has occured in recieved bits.

```
def XOR(x, y):
    if x == y:
        return 0
    return 1

def flip(x):
    if x == 0:
        return 1
    return 0

def moduloDivision(data, dividend, divisor):
    for i in range(len(data)):
        if dividend[i] == 1:
            for j in range(len(divisor)):
                dividend[i + j] = XOR(dividend[i + j], divisor[j])
    return dividend

def displayCRC(data, dividend):
    print("CRC is :", end=" ")
    for i in range(len(data), len(dividend)):
        print(dividend[i], end="")
    print()

def displayCheckSum(data, dividend):
    print("Checksum code is :", end=" ")
    for i in range(len(data)):
        dividend[i] = int(data[i])
        print(dividend[i], end="")
    print()

def main():
    print("Enter data bits : ", end="")
```

```
data = input()
print("Enter check bits : ", end="")
check = input()
dividend = [0] * (len(data) + len(check) - 1)
divisor = [0] * len(check)

for i in range(len(data)):
    dividend[i] = int(data[i])

for i in range(len(check)):
    divisor[i] = int(check[i])

# Calculating remainder (CRC)
dividend = moduloDivision(data, dividend, divisor)

# Display remainder
displayCRC(data, dividend)

# Display checksum
displayCheckSum(data, dividend)

# Asking for a change in checksum
print("Do you want to put error bit(0/1) : ", end="")
choice = int(input())

if choice == 1:
    print("How many error bits do you want to change : ", end="")
    select = int(input())
    print("Enter the bit number you want to change : ", end="")
    change = input()
    for i in range(select):
        dividend[int(change[i])] = flip(dividend[int(change[i])])
```

```
        dividend = moduloDivision(data, dividend, divisor)
        displayCRC(data, dividend)
        print("We see that the remainder is not 0. Hence data is corrupted!!")
    else:
        print("CRC obtained at the receiver side is zero")
        print("Data sent without corruption")

if __name__ == "__main__":
    main()
```

OUTPUT



EXPERIMENT : 4

AIM: Simulation of Go-Back-N Flow control algorithm

THEORY: The Go-Back-N protocol is a network protocol used in the data link layer of the OSI model to ensure reliable communication over a network. It is a sliding window protocol. Go-Back-N is designed for use in situations where packet loss is expected, such as in wireless or unreliable network environments.

1. **Sliding Window:** Go-Back-N employs a sliding window mechanism. The sender can transmit multiple packets before waiting for acknowledgements, while the reciever can accept and process packets out of order.

2. **Sender Window:** The sender maintains a fixed-size window that determines the maximum number of unacknowledge packets it can send.

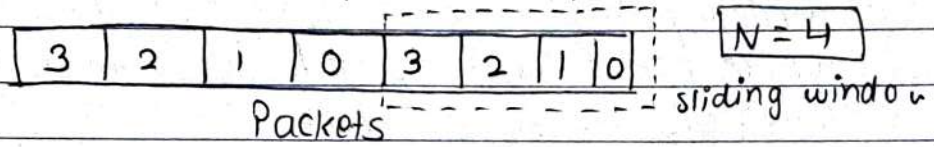
3. **Reciever Window:** The reciever also maintains a window, indicating the expected sequence number of incoming packets.

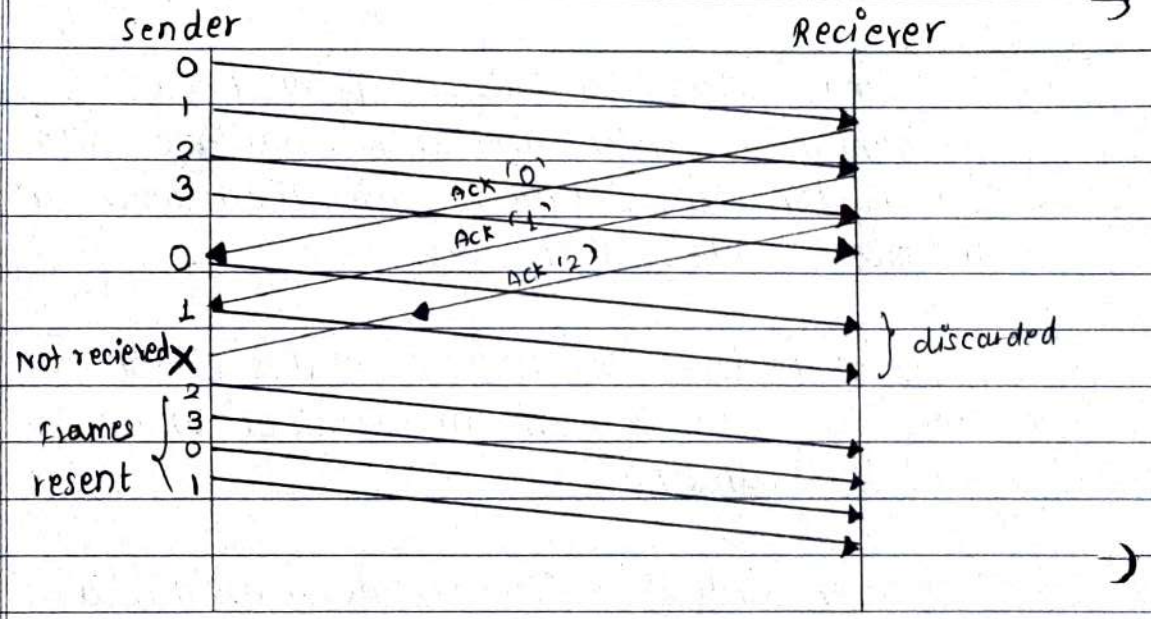
4. **Sequence Numbers:** Each packet is assigned a sequence number, which helps in tracking and ordering packets.

5. **Timeouts:** The sender sets timeout for each transmitted packet. If an acknowledgement is known not recieved within the timeout period, the sender assumes the packet was lost and retransmits it.

Example: Consider the Go Back N protocol where sender~~sliding~~ window size is given as 4, which means sender can send up to 4 packets unacknowledged at a time.

Since the window size is 4, the sequence number given to the packets will be 0,1,2,3,0,1,2,3 and so on. Consider there are 8 packets represented below:





In the above example, we can see that the sender sends 4 unacknowledged packets (0, 1, 2, 3). The sliding window shifts only if the sender recieves an acknowledgement from reciever. We can see that sender recieves acknowledgemet for frame 0. window slides further by sending next packet to reciever. Now, sender recieves another acknowledgement for packet 1. window slides further. This times sender does not recieves any acknowledgement. Therefore, the packets present in the sliding window are retransmitted again. whereas, the previously sent packets are discarded Go back to Packet 2. This is Go-Back-N protocol.

```
def main():
    print("Enter window size: ")
    window_size = int(input())
    print("Enter total frames to be sent: ")
    total_frames = int(input())

    # Initializing array with data frames
    sender_frames = [i for i in range(total_frames)]

    # Displaying data frames
    for frame in sender_frames:
        print(frame, "| ", end="")
    print()

    # Displaying sender window
    print("Do you want to start sending frames (0/1): ")
    choice = int(input())
    print()

    if choice == 1:
        ptr_on_window_left_sender = 0
        ptr_on_window_left_receiver = 0
        total_sent_frames = 0

        while ptr_on_window_left_sender < total_frames:
            # Sender side
            count = 0
            print("At Sender End:")
            for i in range(ptr_on_window_left_sender, total_frames):
                if count < window_size:
                    print("Sent frame[", i + 1, "]")
                    ptr_on_window_left_sender += 1
                    total_sent_frames += 1
                    count += 1
```

```
            else:
                break

        print()

        # Receiver side
        print("At Receiver end:")
        j = 0
        count = 0
        for i in range(ptr_on_window_left_receiver, total_frames):
            if count < window_size:
                print("Did you receive frame[", i + 1, "] (y/n) : ")
                y_n = input()
                if y_n == 'n':
                    print("Frames will again be sent from frame no.", i + 1)
                    print()
                    ptr_on_window_left_sender = i
                    break
                else:
                    j += 1
                    ptr_on_window_left_receiver += 1
                count += 1
            else:
                break

        if j == window_size:
            print("All Frames from this window sent without errors. Sending next frames...")
            print()

    print()
    print("All frames are sent.")
    print("Total no. of frames sent including retransmission is", total_sent_frames)

if __name__ == "__main__":
```

main()

OUTPUT

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS D:\CN> python -u "d:\CN\GoBackN.py"
Enter window size:
4
Enter total frames to be sent:
8
0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
Do you want to start sending frames (0/1):
1

At Sender End:
Sent frame[ 1 ]
Sent frame[ 2 ]
Sent frame[ 3 ]
Sent frame[ 4 ]

At Receiver end:
Did you receive frame[ 1 ] (y/n) :
y
Did you receive frame[ 2 ] (y/n) :
y
Did you receive frame[ 3 ] (y/n) :
y
Did you receive frame[ 4 ] (y/n) :
y
All Frames from this window sent without errors. Sending next frames...
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

At Sender End:
Sent frame[ 5 ]
Sent frame[ 6 ]
Sent frame[ 7 ]
Sent frame[ 8 ]

At Receiver end:
Did you receive frame[ 5 ] (y/n) :
y
Did you receive frame[ 6 ] (y/n) :
y
Did you receive frame[ 7 ] (y/n) :
n
Frames will again be sent from frame no. 7

At Sender End:
Sent frame[ 7 ]
Sent frame[ 8 ]

At Receiver end:
Did you receive frame[ 7 ] (y/n) :
y
Did you receive frame[ 8 ] (y/n) :
y

All frames are sent.
Total no. of frames sent including retransmission is 10
PS D:\CN>
```

EXPERIMENT : 5

AIM: Build a simple network topology and configure it for static routing protocol using packet tracer. Setup a network and configure IP addressing, subnetting, masking.

THEORY: Network topology refers to the physical or logical layout of devices and connections within a network. Different networks topologies are suited for various applications and have unique advantages and disadvantages. Here are a few different network topologies along with small examples of their applications.

Star Topology: In a star topology, all devices are connected to a central hub or switch. Devices do not connect directly to each other.
Star topology is commonly used in home networks where all devices (computers, printers, smart devices) connect to a central router or switch. This topology is straightforward to set up and manage.

Bus Topology : In a bus topology, all devices share a single communication line (Bus). Data is transmitted down the bus, and devices can "tap-in" to access the data.
Bus topologies were historically used in early ethernet networks. While they are less common today, they can still be found in applications like industrial control system.

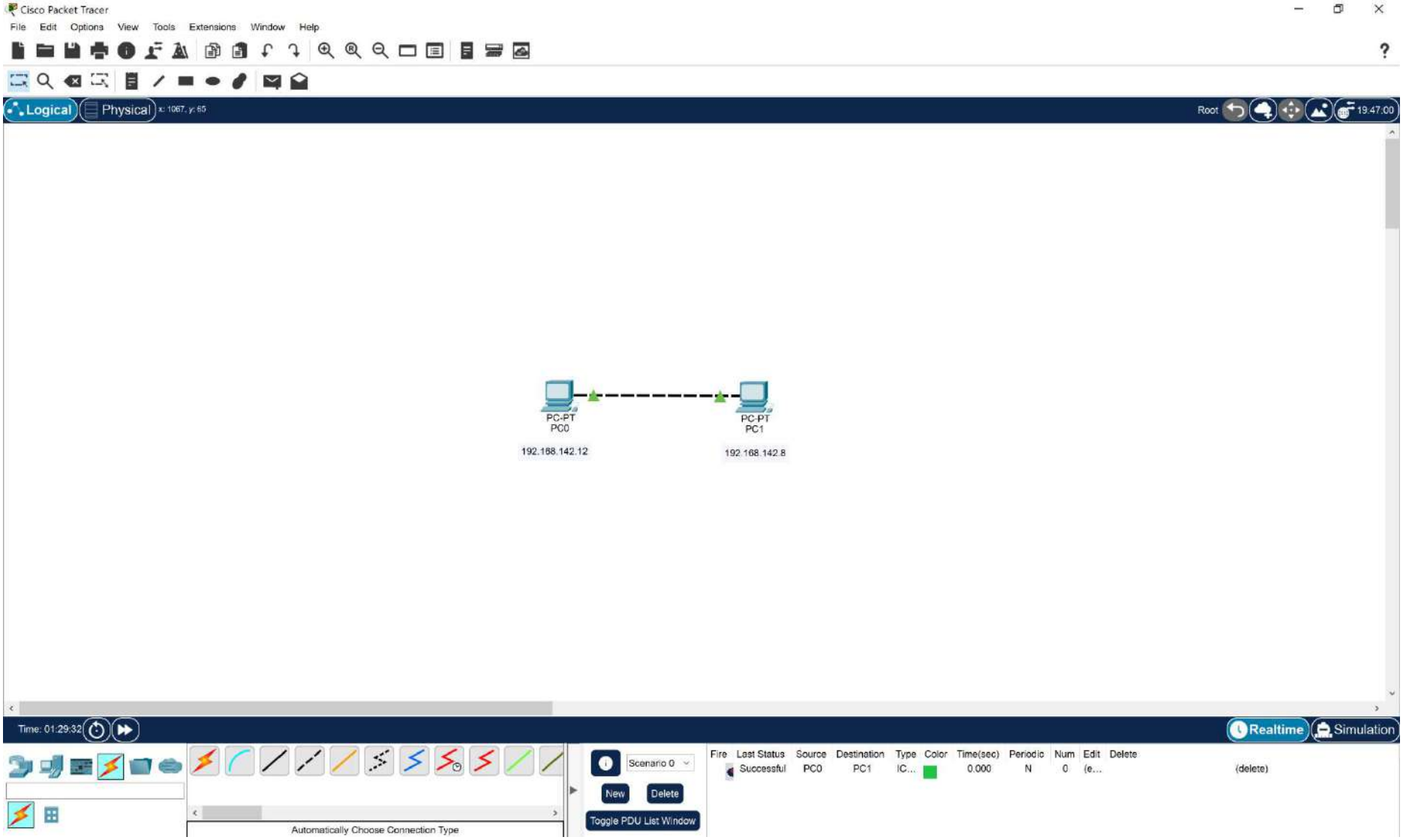
Ring Topology : In a ring topology, each device is connected to exactly two other devices, forming a closed loop.

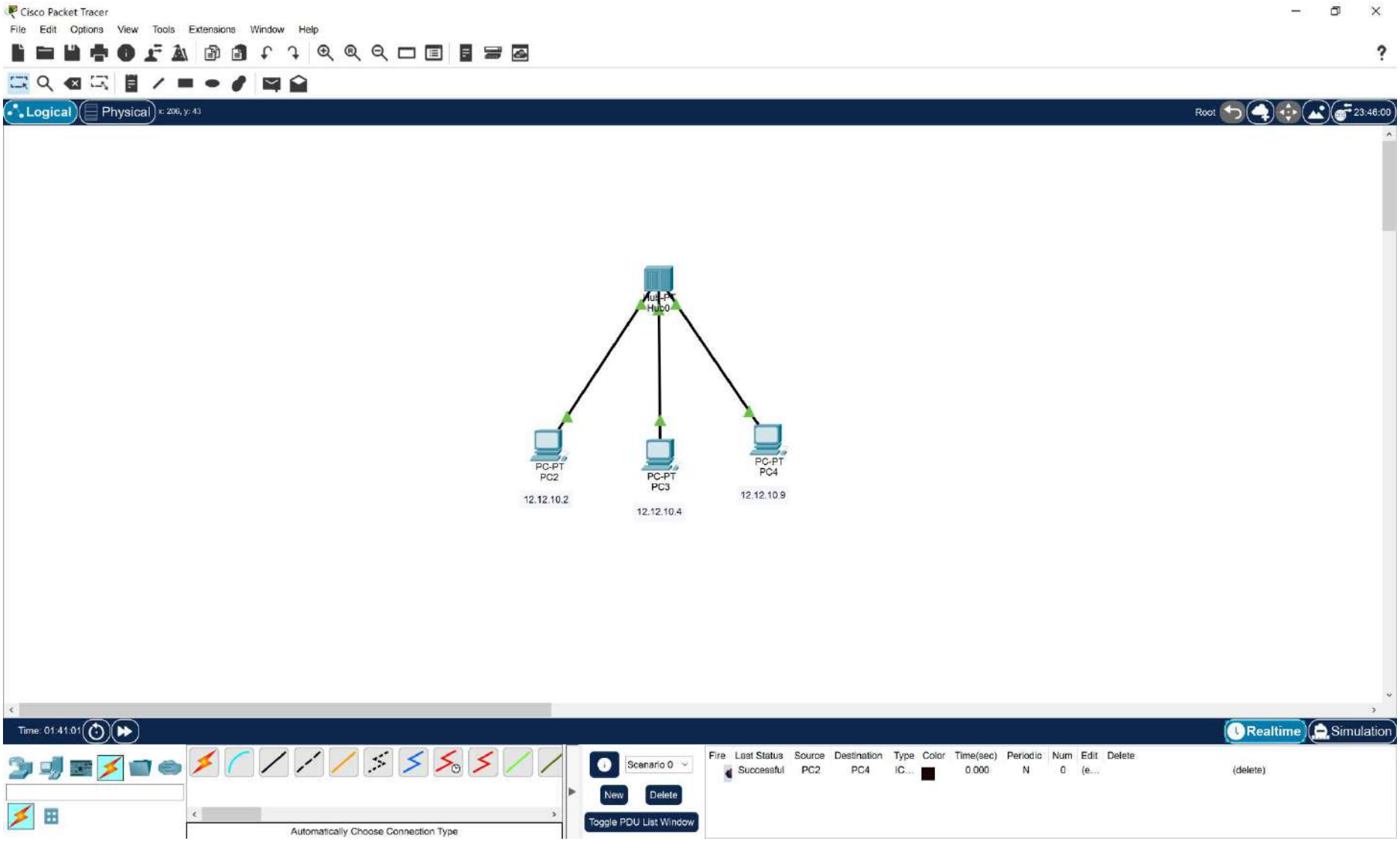
Data travels in one direction around the ring.
Ring topologies are less common in modern LANs but are used in specific applications like fibre optic networks. Token ring networks used this topology in the past.

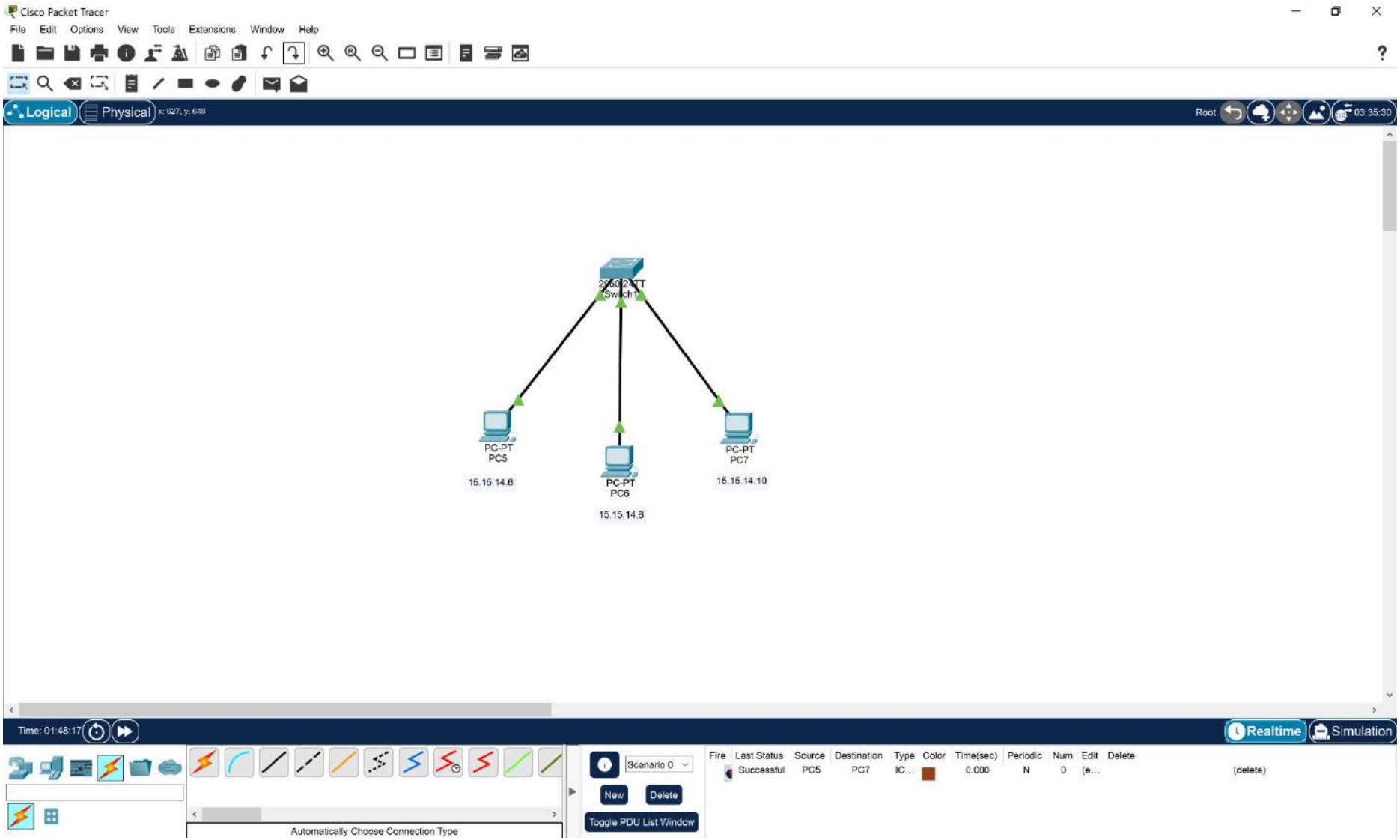
Mesh Topology: In a mesh topology, every device is connected to every other device, creating multiple plans/ paths for data to travel.
Mesh topologies are often used in critical applications, like data centres, where redundancy and fault tolerance are crucial. Each device can communicate directly with any other, enhancing reliability.

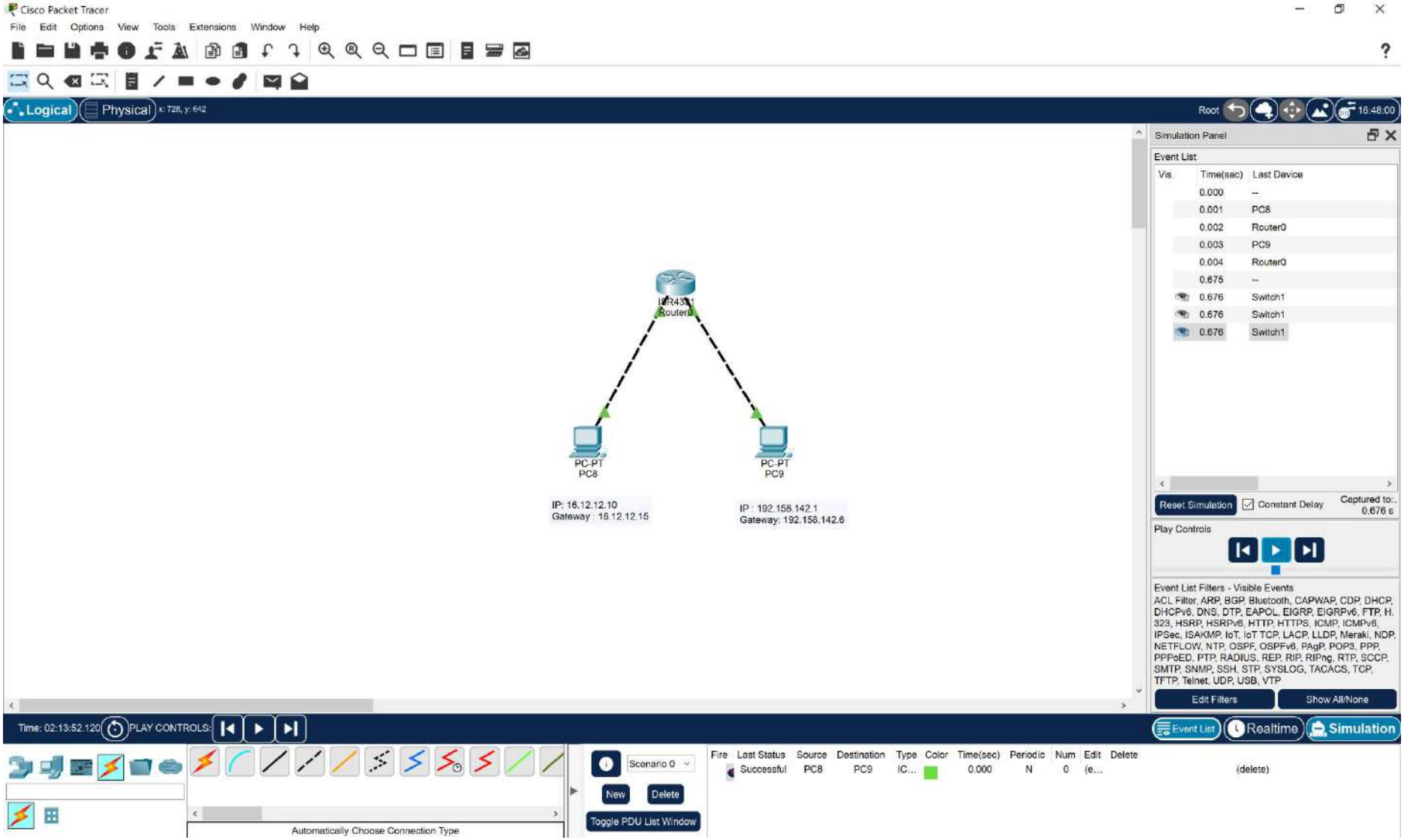
Hybrid Topology: A hybrid topology combines two or more different topologies into a single network. For example, a network might have a star topology in one office and a ring topology in another office connected through a router. Hybrid topologies allow organizations to tailor their network to specific needs. They are commonly found in large enterprises with diverse networking requirements.

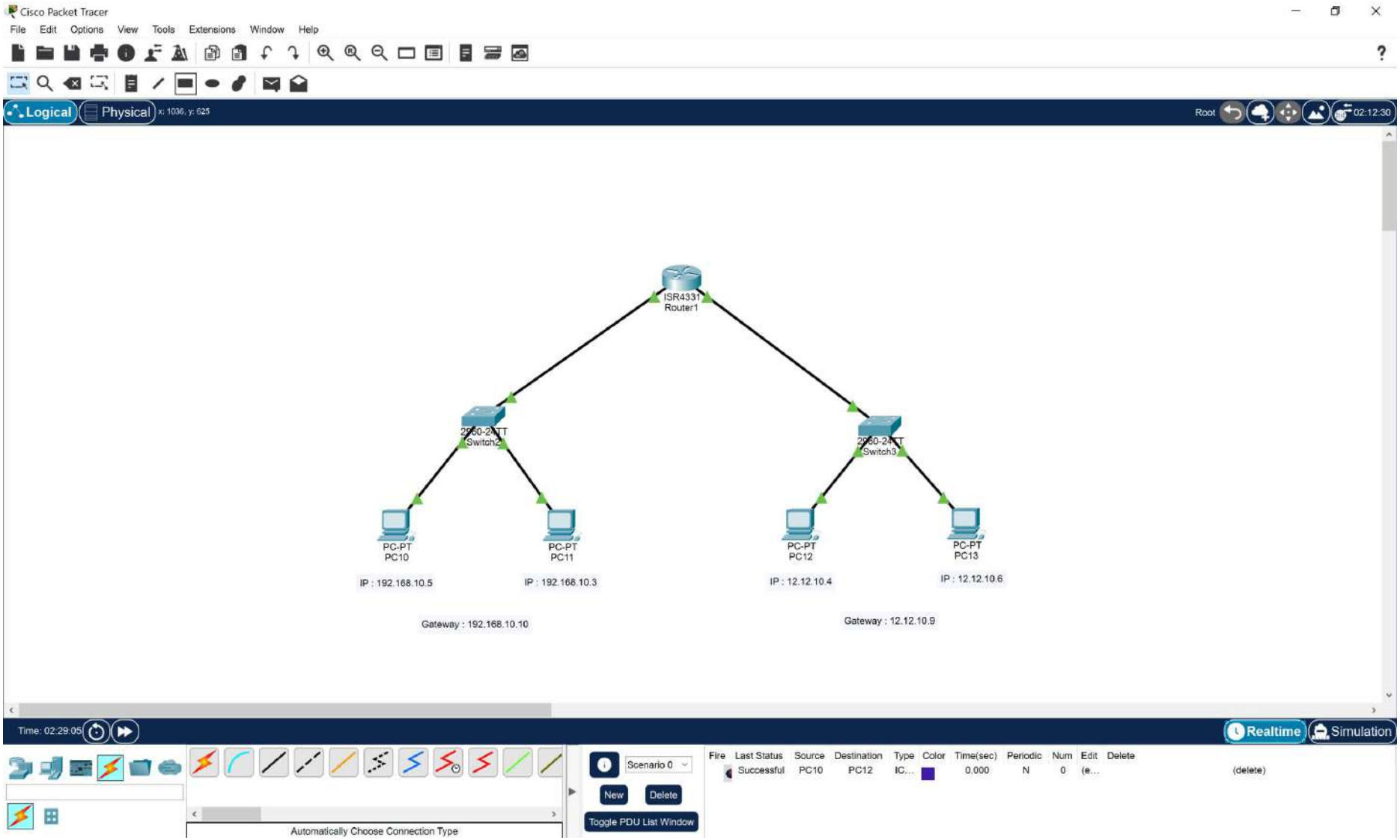
Tree (Hierarchical) Topology: In a tree topology, multiple stars are connected. A central hub from each star connects to a main central hub. Tree topologies are often used in large-scale networks, such as corporate networks with multiple branches. Each branch office may have a star topology and they all connect to the main office using a hierarchical structure.


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
19:47:00
PC-PT
PC0
192.168.142.12
PC-PT
PC1
192.168.142.8
Time: 01:29:32
Realtime
Simulation
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color Time(sec) Periodic Num Edit Delete
Successful PC0 PC1 IC... 0.000 N 0 (e... (delete)


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
23:46:00
Multi-PT Hub0
PC-PT PC2
12.12.10.2
PC-PT PC3
12.12.10.4
PC-PT PC4
12.12.10.9
Time: 01:41:01
Realtime
Simulation
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color Time(sec) Periodic Num Edit Delete
Successful PC2 PC4 IC... 0.000 N 0 (e... (delete)


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
03:35:30
PC-PT
PC5
15.15.14.6
PC-PT
PC6
15.15.14.8
PC-PT
PC7
15.15.14.10
Time: 01:48:17
Realtime
Simulation
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color Time(sec) Periodic Num Edit Delete
Successful PC5 PC7 IC... 0.000 N 0 (e... (delete)
Automatically Choose Connection Type


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
16:48:00
Router0
PC-PT
PC8
PC-PT
PC9
IP: 16.12.12.10
Gateway : 16.12.12.15
IP : 192.158.142.1
Gateway: 192.158.142.6
Simulation Panel
Event List
Vis. Time(sec) Last Device
0.000 --
0.001 PC8
0.002 Router0
0.003 PC9
0.004 Router0
0.675 --
0.676 Switch1
0.676 Switch1
0.676 Switch1
Reset Simulation
Constant Delay
Captured to:
0.676 s
Play Controls
Event List Filters - Visible Events
ACL Filter, ARP, BGP, Bluetooth, CAPWAP, CDP, DHCP, DHCPv6, DNS, DTP, EAPOL, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, IoT, IoT TCP, LACP, LLDP, Meraki, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, PPP, PPPoED, PTP, RADIUS, REP, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, USB, VTP
Edit Filters
Show All/None
Time: 02:13:52.120
PLAY CONTROLS:
Event List
Realtime
Simulation
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color Time(sec) Periodic Num Edit Delete
Successful PC8 PC9 IC... 0.000 N 0 (e... (delete)


Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
02:12:30
ISR4331
Router1
2960-24TT
Switch2
2960-24TT
Switch3
PC-PT
PC10
PC-PT
PC11
PC-PT
PC12
PC-PT
PC13
IP : 192.168.10.5
IP : 192.168.10.3
IP : 12.12.10.4
IP : 12.12.10.6
Gateway : 192.168.10.10
Gateway : 12.12.10.9
Time: 02:29:05
Realtime
Simulation
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Num
Edit
Delete
Successful
PC10
PC12
0.000
N
0
(delete)
Automatically Choose Connection Type

EXPERIMENT NO: 6

AIM: Design VPN and configure RIP/OSPF using Packet Tracer.

THEORY:

Routing Information Protocol: (RIP) is one of the oldest distance-vector routing protocols used in computer networks. It's primary purpose is to determine the best path for data packets to travel from one network segment to another within an IP based network.

Routing: Refers to the process of forwarding data packets from one network to another. Routers are devices responsible for this task.

Distance-Vector-Protocol: RIP uses a distance-vector algorithm to determine the best path for packet transmission. Each router in the network maintains a routing table, which contains information about the distance (number of hops) and direction to reach various network destinations.

Hop Count: RIP uses hop count as a metric to determine the best path. A hop is a passage through one router to another. RIP routers exchange routing information periodically, updating their routing tables based on the number of hops to reach different networks.

Routing Updates: RIP routers send routing updates at regular intervals to share information about the networks they know. These updates contain the network addresses and associated hop counts.

Use case: RIP is typically used in small to medium-size networks where simplicity is more important than rapid convergence.

Virtual Local Area Networks (VLANs):

VLANs are a method of segmenting a physical network into multiple logical networks, improving networks efficiency and security.

Logical Segmentation: VLAN's allow you to logically divide a physically LAN into multiple isolated networks. These networks behave as if they are on seperate physical switches, even if they share the same physical infrastructure.

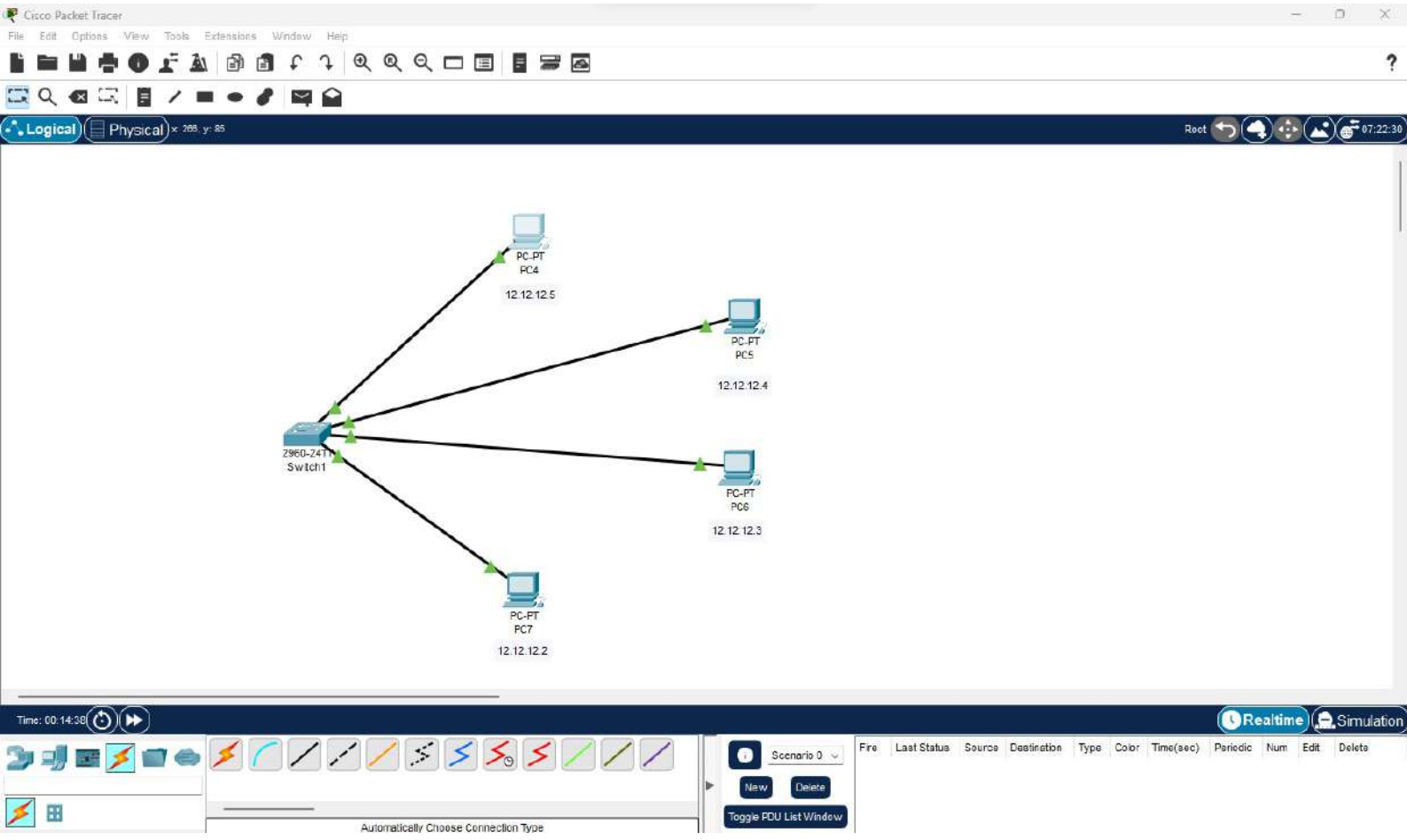
Flexibility: VLAN's are flexible and can be configured based on different criteria, such as department, function, or security requirements. VLAN configuration is typically done at the switch level.

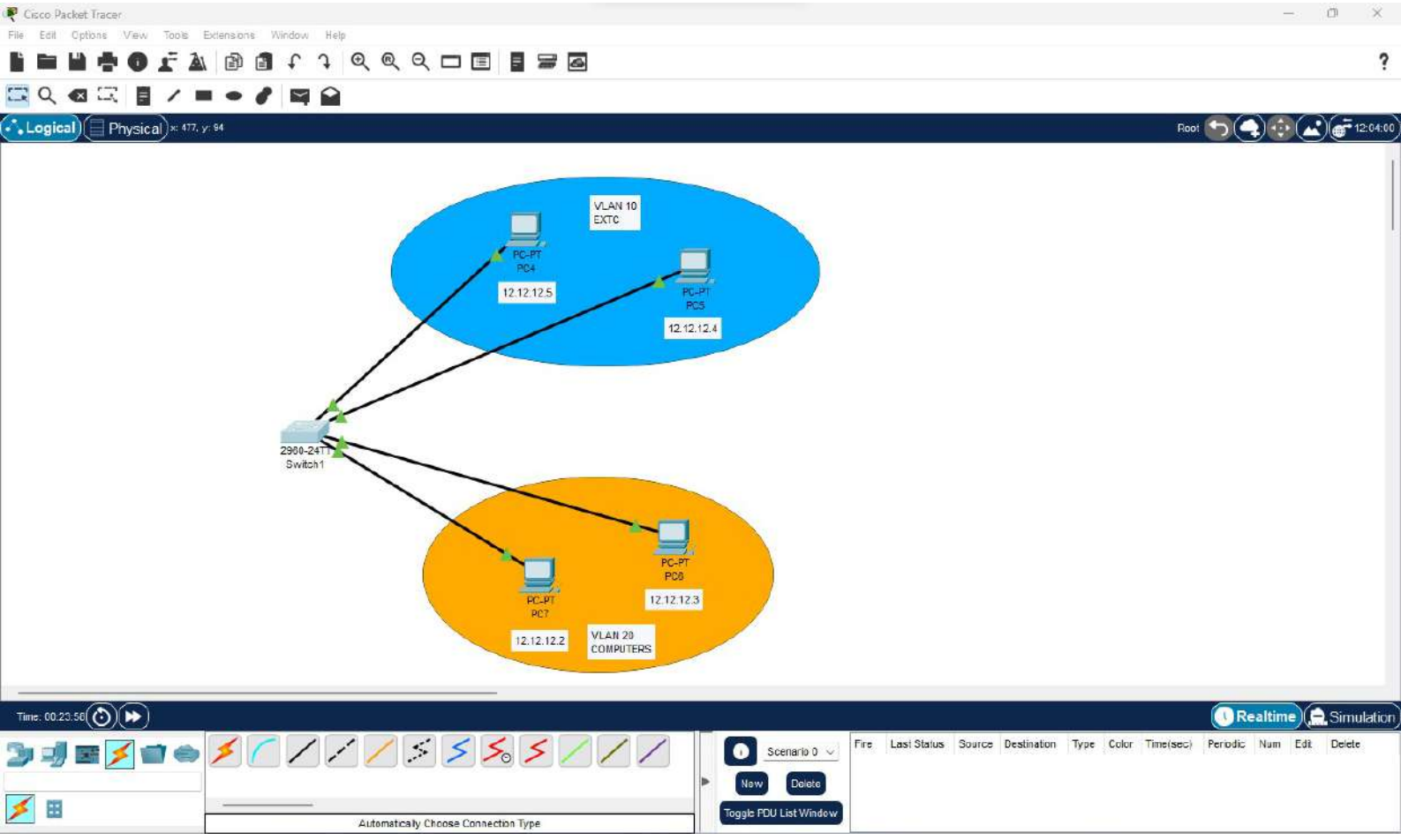
Security: VLANs enhance network security by isolating sensitive data or services. For example, a company might create seperate VLANs for guest WIFI, employee workstations, and servers to isolate traffics and enhance security.

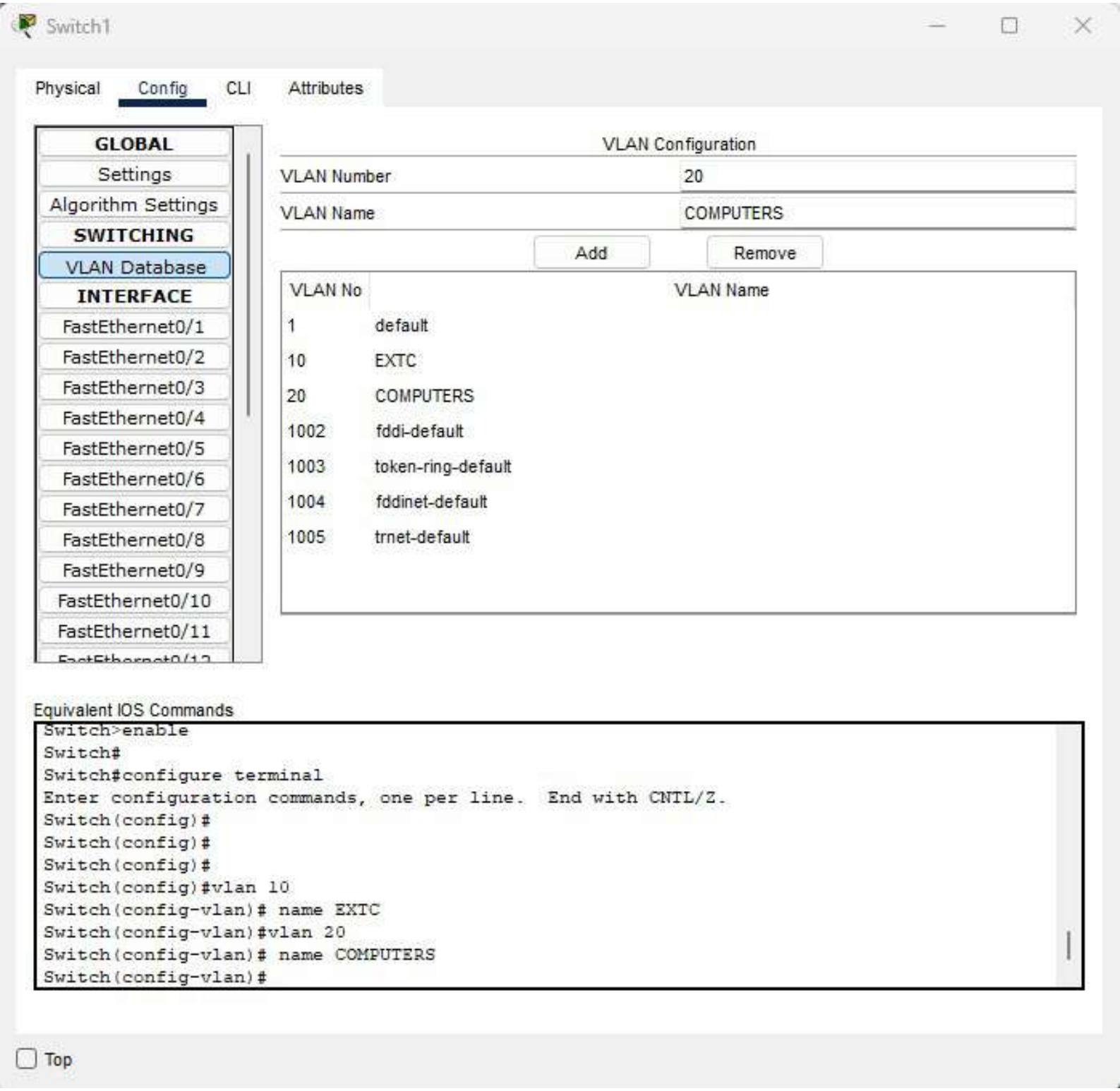
Scalability: VLANs are highly scalable, making them suitable for large and complex network environments.

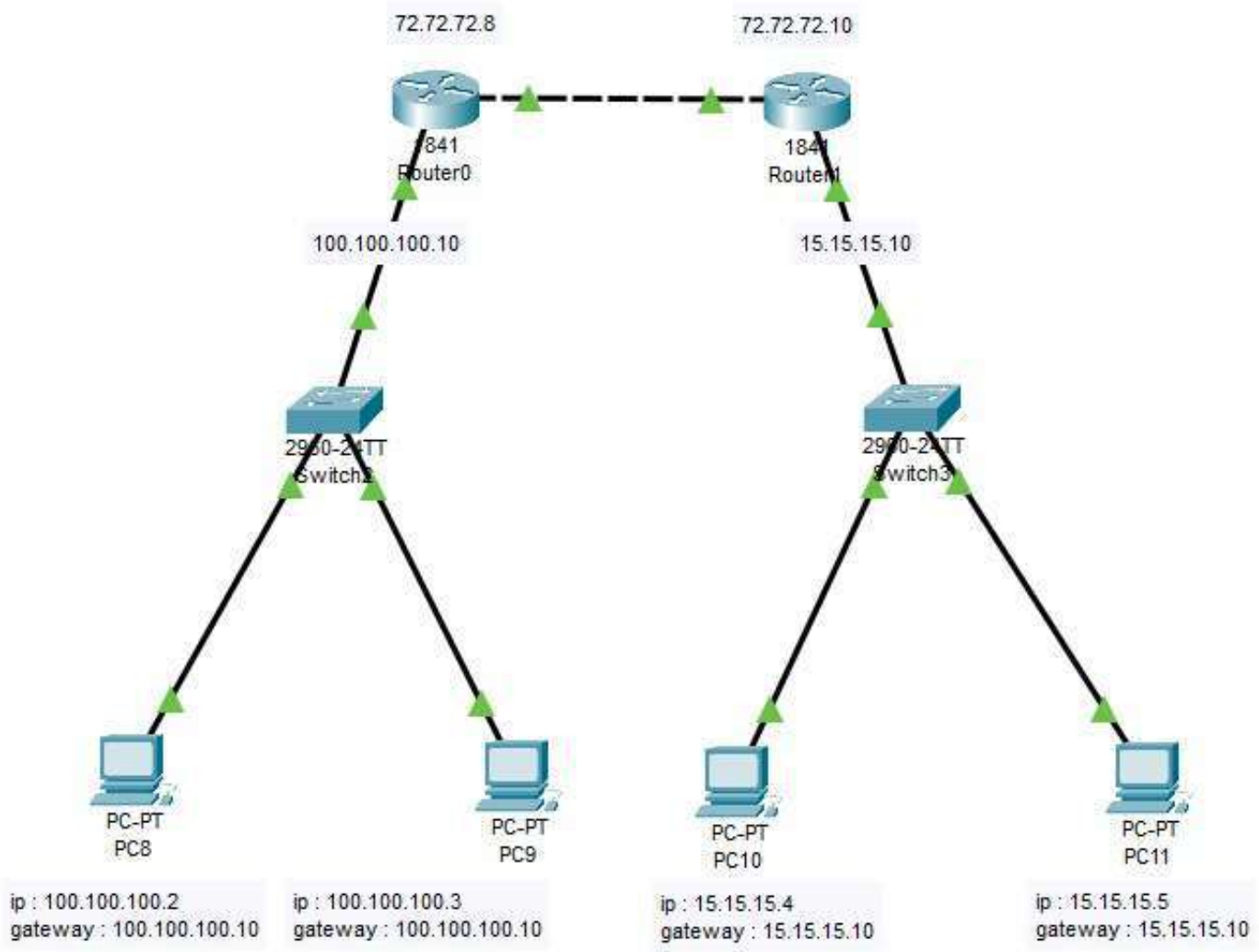
Use Cases: VLANs are commonly used in enterprise networks, data centers, and environments where network segmentation and essential.

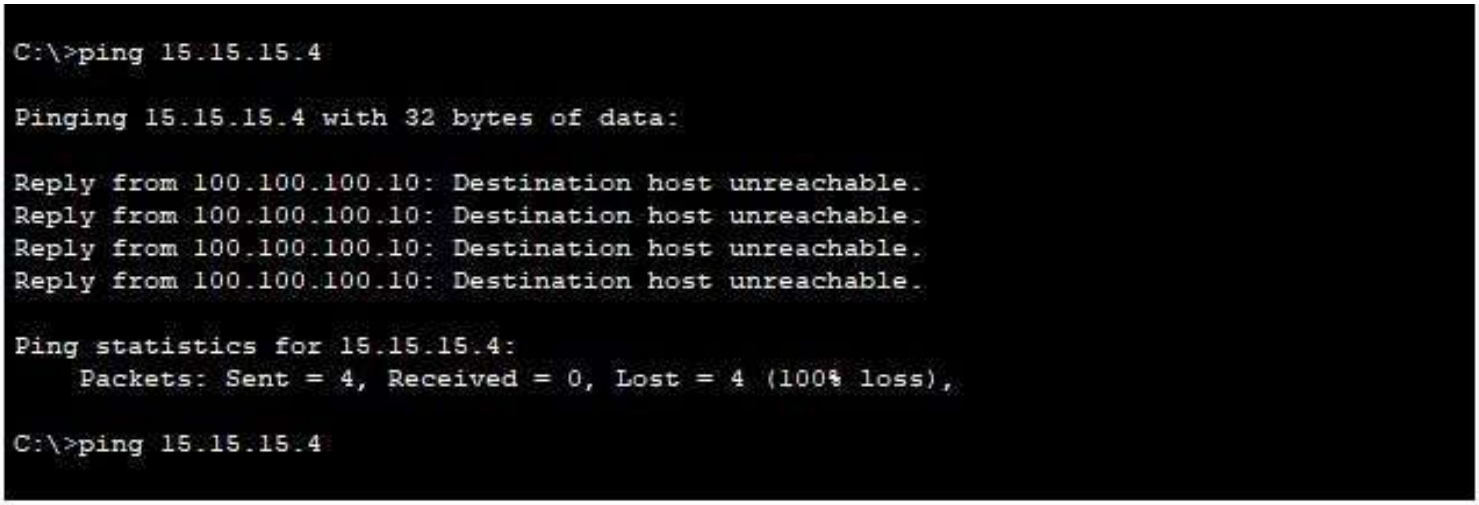
Inter-VLAN Routing: For communication between VLANs a router or Layer 3 switch is required. This allows controlled communication between different VLANs while maintaining isolation.

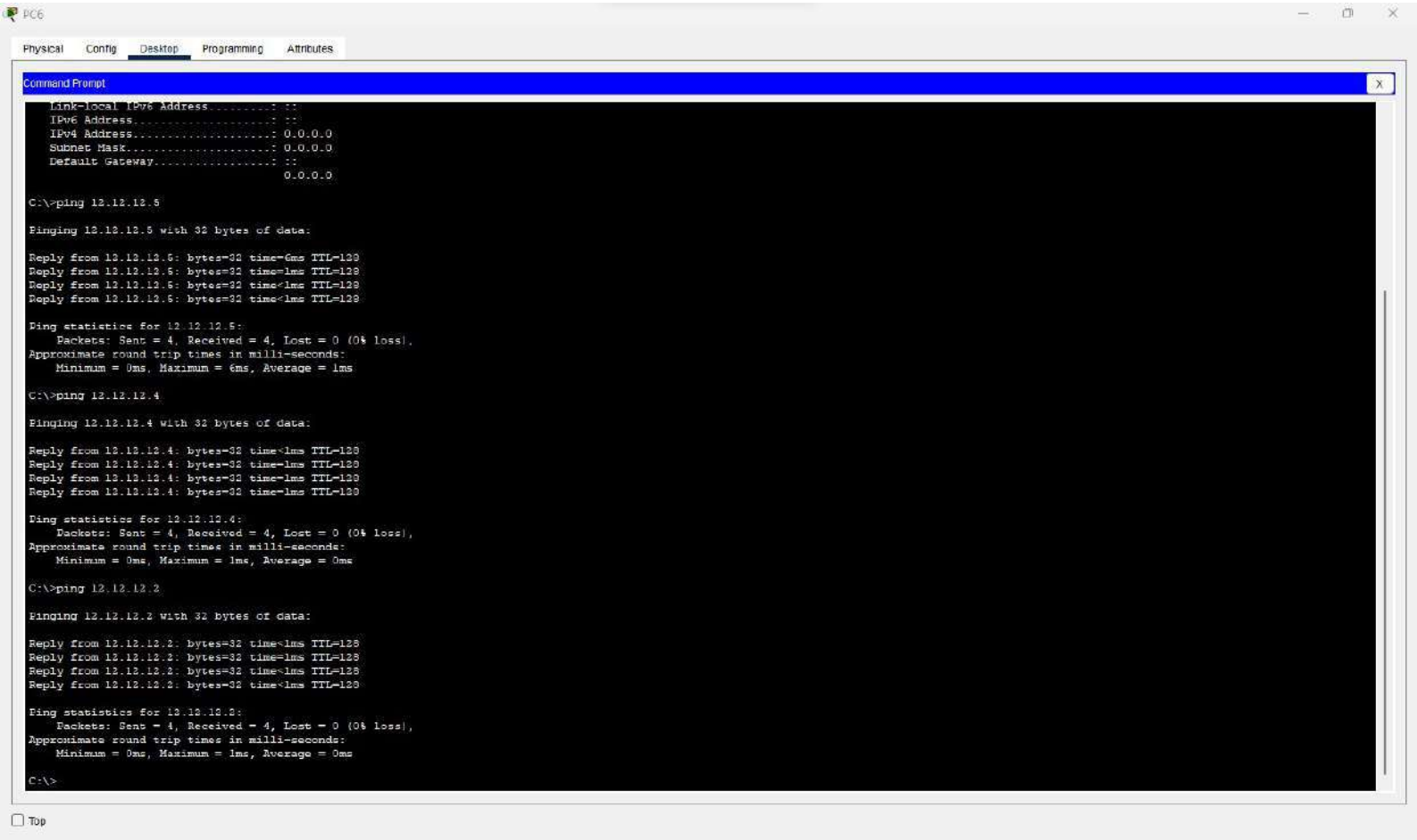

Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
Root
07:22:30
PC-PT
PC4
12.12.12.5
PC-PT
PC5
12.12.12.4
2960-24TT
Switch1
PC-PT
PC6
12.12.12.3
PC-PT
PC7
12.12.12.2
Time: 00:14:38
Realtime
Simulation
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Num
Edit
Delete
Automatically Choose Connection Type

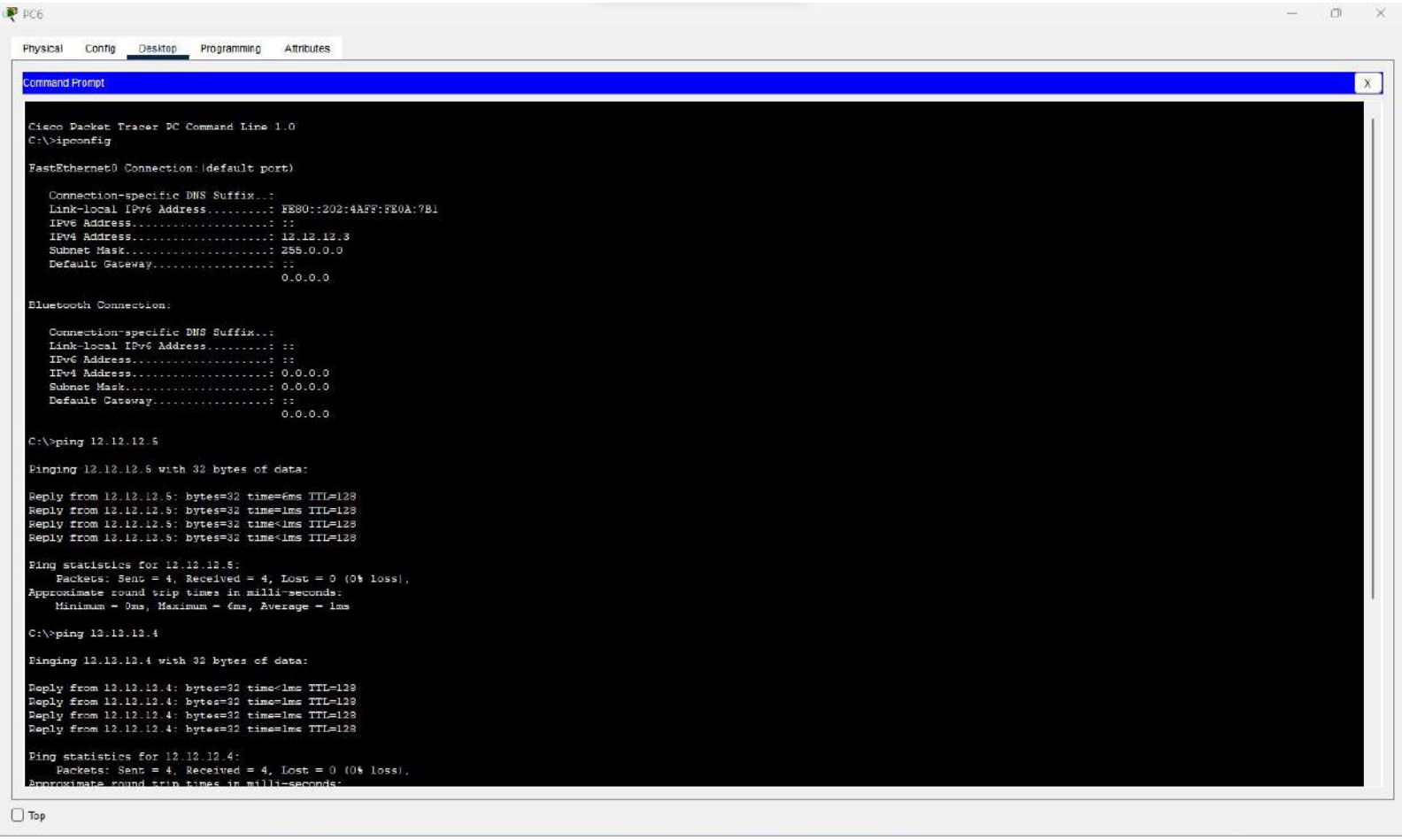

Cisco Packet Tracer
File Edit Options View Tools Extensions Window Help
Logical
Physical
x: 477, y: 94
Root
12:04:00
VLAN 10
EXTC
PC-PT
PC4
12.12.12.5
PC-PT
PC5
12.12.12.4
2960-24TT
Switch1
PC-PT
PC8
12.12.12.3
PC-PT
PC7
12.12.12.2
VLAN 20
COMPUTERS
Time: 00:23:58
Realtime
Simulation
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Num
Edit
Delete

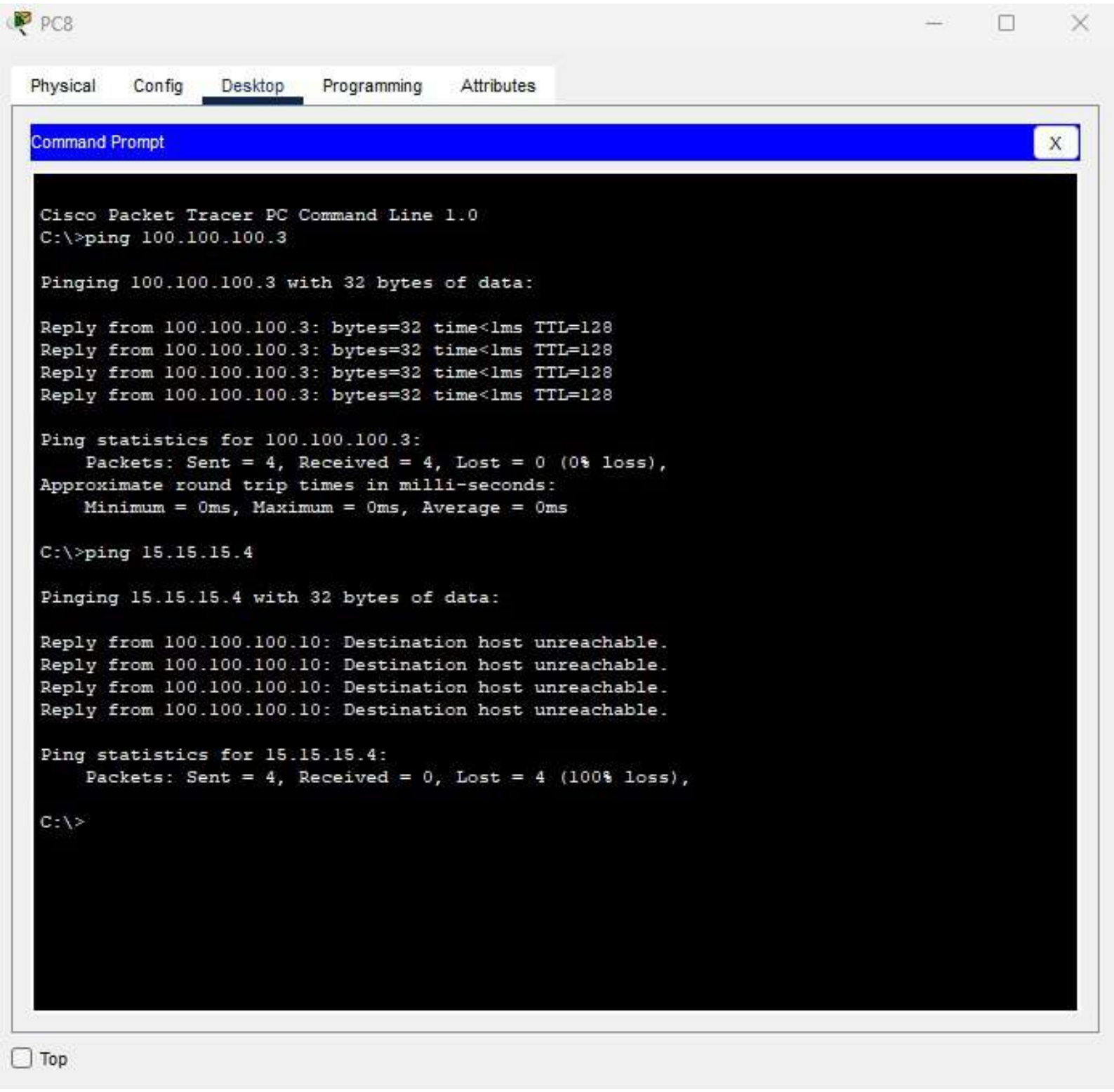

Switch1
Physical
Config
CLI
Attributes
GLOBAL
Settings
Algorithm Settings
SWITCHING
VLAN Database
INTERFACE
FastEthernet0/1
FastEthernet0/2
FastEthernet0/3
FastEthernet0/4
FastEthernet0/5
FastEthernet0/6
FastEthernet0/7
FastEthernet0/8
FastEthernet0/9
FastEthernet0/10
FastEthernet0/11
VLAN Configuration
VLAN Number
20
VLAN Name
COMPUTERS
Add
Remove
VLAN No
VLAN Name
1 default
10 EXTC
20 COMPUTERS
1002 fddi-default
1003 token-ring-default
1004 fddinet-default
1005 trnet-default
Equivalent IOS Commands
Switch>enable
Switch#
Switch#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Switch(config)#
Switch(config)#
Switch(config)#
Switch(config)#vlan 10
Switch(config-vlan)# name EXTC
Switch(config-vlan)#vlan 20
Switch(config-vlan)# name COMPUTERS
Switch(config-vlan)#
Top


72.72.72.8
72.72.72.10
1841
Router0
1841
Router1
100.100.100.10
15.15.15.10
2960-24TT
Switch2
2960-24TT
Switch3
PC-PT
PC8
PC-PT
PC9
PC-PT
PC10
PC-PT
PC11
ip : 100.100.100.2
gateway : 100.100.100.10
ip : 100.100.100.3
gateway : 100.100.100.10
ip : 15.15.15.4
gateway : 15.15.15.10
ip : 15.15.15.5
gateway : 15.15.15.10


C:\>ping 15.15.15.4
Pinging 15.15.15.4 with 32 bytes of data:
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Ping statistics for 15.15.15.4:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
C:\>ping 15.15.15.4


PC6
Physical
Config
Desktop
Programming
Attributes
Command Prompt
X
Link-local IPv6 Address.....: ::
IPv6 Address.....: ::
IPv4 Address.....: 0.0.0.0
Subnet Mask.....: 0.0.0.0
Default Gateway.....: ::
0.0.0.0
C:\>ping 12.12.12.5
Pinging 12.12.12.5 with 32 bytes of data:
Reply from 12.12.12.5: bytes=32 time=6ms TTL=128
Reply from 12.12.12.5: bytes=32 time=1ms TTL=128
Reply from 12.12.12.5: bytes=32 time<1ms TTL=128
Reply from 12.12.12.5: bytes=32 time<1ms TTL=128
Ping statistics for 12.12.12.5:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 6ms, Average = 1ms
C:\>ping 12.12.12.4
Pinging 12.12.12.4 with 32 bytes of data:
Reply from 12.12.12.4: bytes=32 time<1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Ping statistics for 12.12.12.4:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 1ms, Average = 0ms
C:\>ping 12.12.12.2
Pinging 12.12.12.2 with 32 bytes of data:
Reply from 12.12.12.2: bytes=32 time<1ms TTL=128
Reply from 12.12.12.2: bytes=32 time=1ms TTL=128
Reply from 12.12.12.2: bytes=32 time<1ms TTL=128
Reply from 12.12.12.2: bytes=32 time<1ms TTL=128
Ping statistics for 12.12.12.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 1ms, Average = 0ms
C:\>
Top


PC6
Physical
Config
Desktop
Programming
Attributes
Command Prompt
X
Cisco Packet Tracer PC Command Line 1.0
C:\>ipconfig
FastEthernet0 Connection:(default port)
Connection-specific DNS Suffix..:
Link-local IPv6 Address.....: FE80::202:4AFF:FE0A:7B1
IPv6 Address.....: ::
IPv4 Address.....: 12.12.12.3
Subnet Mask.....: 255.0.0.0
Default Gateway.....: ::
0.0.0.0
Bluetooth Connection:
Connection-specific DNS Suffix..:
Link-local IPv6 Address.....: ::
IPv6 Address.....: ::
IPv4 Address.....: 0.0.0.0
Subnet Mask.....: 0.0.0.0
Default Gateway.....: ::
0.0.0.0
C:\>ping 12.12.12.5
Pinging 12.12.12.5 with 32 bytes of data:
Reply from 12.12.12.5: bytes=32 time=6ms TTL=128
Reply from 12.12.12.5: bytes=32 time=1ms TTL=128
Reply from 12.12.12.5: bytes=32 time<1ms TTL=128
Reply from 12.12.12.5: bytes=32 time<1ms TTL=128
Ping statistics for 12.12.12.5:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 6ms, Average = 1ms
C:\>ping 12.12.12.4
Pinging 12.12.12.4 with 32 bytes of data:
Reply from 12.12.12.4: bytes=32 time<1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Reply from 12.12.12.4: bytes=32 time=1ms TTL=128
Ping statistics for 12.12.12.4:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Top


PC8
Physical
Config
Desktop
Programming
Attributes
Command Prompt
X
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 100.100.100.3
Pinging 100.100.100.3 with 32 bytes of data:
Reply from 100.100.100.3: bytes=32 time<1ms TTL=128
Reply from 100.100.100.3: bytes=32 time<1ms TTL=128
Reply from 100.100.100.3: bytes=32 time<1ms TTL=128
Reply from 100.100.100.3: bytes=32 time<1ms TTL=128
Ping statistics for 100.100.100.3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms
C:\>ping 15.15.15.4
Pinging 15.15.15.4 with 32 bytes of data:
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Reply from 100.100.100.10: Destination host unreachable.
Ping statistics for 15.15.15.4:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
C:\>
Top

EXPERIMENT : 7

AIM : Write a program to implement IPv4 addressing concept along with subnet masking.

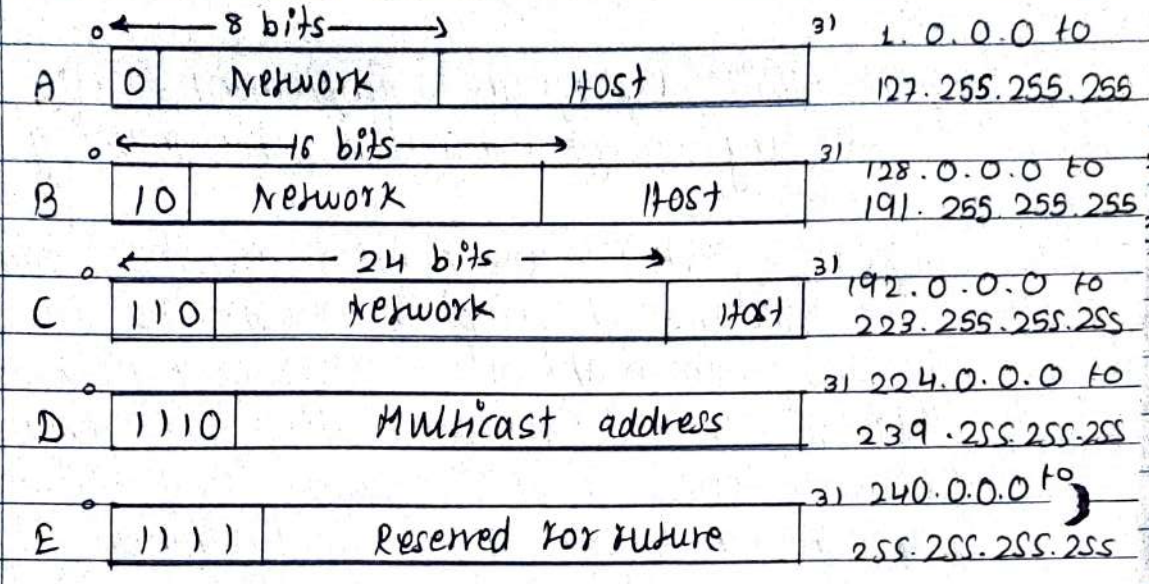
THEORY : IPv4 (Internet Protocol version 4) addressing is a fundamental concept in network that involves assigning numerical labels to devices on a network. These addresses are used for identification and communication between devices in the IP network. IPv4 addresses are represented as a series of 4 decimal numbers, each ranging from 0 to 255, seperated by periods (.). Each of these numbers is an octet, representing 8 bits of the 32 bit IPv4 address.

IP address of a system is a 32 bit unique address. It is divided into two parts: The network portion and the host portion. The boundary between these two portions is determined by a subnet mask.

Subnet Mask: The subnet mask is a 32-bit number that defines the boundary between the network and host portions of IPv4 address. It consists of a series of contiguous 1s followed by a series of contiguous 0s. The 1s indicate the network bits and the 0s indicate the host bits.

Network ID and Host ID: The network ID portion of the address identifies the network to which a device belongs, while the host portion identifies the specific device within that network. Devices within the same network share the same Net ID.

Classes of IPv4 address:



Example:- Give the class as well as subnet mask for IP: 200.1.2.10 divided into 2 subnets.

Given: 200.1.2.10 ∴ Network address: 200.1.2.0
The given IP is Class C Ip address.

Network ID: 200.1.2.0
Host ID: 10.

To get the subnet mask for above IP, make all bits of network address as 1 and make all bits of Host Id as 0.

∴ Subnet mask : 255.255.255.0

1st Subnet: Address: 200.1.2.0
Range: 200.1.2.0 - 200.1.2.127

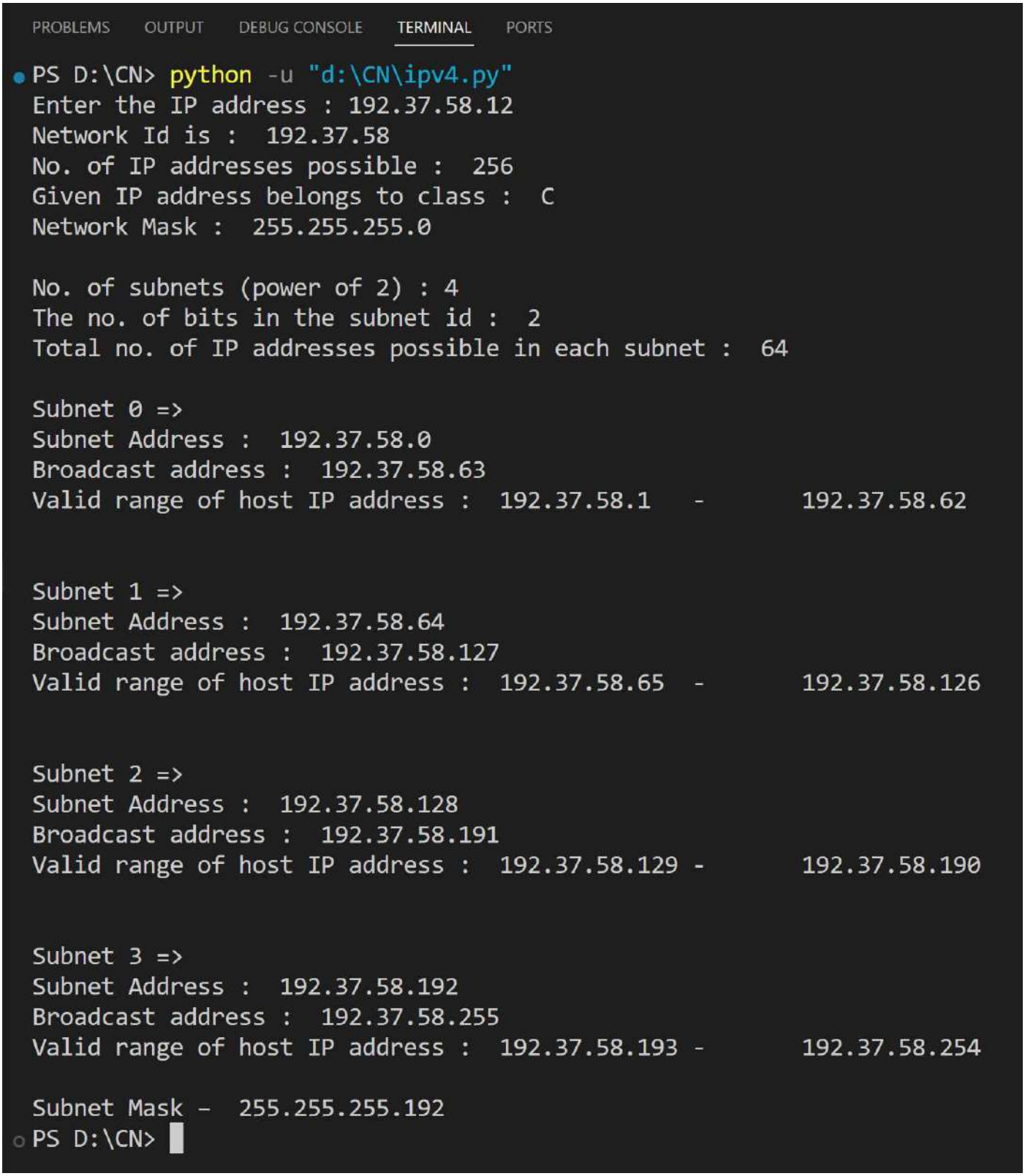
2nd Subnet: address: 200.1.2.128
Range: 200.1.2.128 - 200.1.2.255

```
import math


def findClass(ip):
    if 0 <= ip[0] <= 127:
        print("Network Address is : ", ip[0])
        print('No. of IP addresses possible : ', 2 ** 24)
        return "A", '255.0.0.0'
    elif 128 <= ip[0] <= 191:
        ip = [str(i) for i in ip]
        print("Network Address is : ", ".".join(ip[0:2]))
        print('No. of IP addresses possible : ', 2 ** 16)
        return "B", '255.255.0.0'
    elif 192 <= ip[0] <= 223:
        ip = [str(i) for i in ip]
        print("Network Id is : ", ".".join(ip[0:3]))
        print('No. of IP addresses possible : ', 2 ** 8)
        return "C", '255.255.255.0'
    elif 224 <= ip[0] <= 239:
        print("In this Class, IP address is not divided into Network and Host ID")
        return "D"
    else:
        print("In this Class, IP address is not divided into Network and Host ID")
        return "E"


def Subnetting(ip, num, className, ip_addresses):
    temp = 0
    if className == "A":
        place2 = ip_addresses / (256 ** 2)
        for i in range(num):
```

```
        print(f"Subnet {i} => ")
        print(temp)
        print("Subnet Address : ", ip[0] + '.' + str(temp) + '.0' + '.0')
        temp += int(place2)
        print("Broadcast address : ", ip[0] + '.' + str(temp - 1) + '.255' + '.255')
        print("Valid range of host IP address : ", ip[0] + '.' + str(temp - int(place2)) + '.' + '0' +
'.1' + '\t-\t' + ip[0] + '.' + str(temp - 1) + '.254' + '.254')
        print()
    elif className == "B":
        place2 = ip_addresses / 256
        for i in range(num):
            print(f"\nSubnet {i} => ")
            print("Subnet Address : ", ".".join(ip[0:2]) + '.' + str(temp) + '.0')
            temp += int(place2)
            print("Broadcast address : ", ".".join(ip[0:2]) + '.' + str(temp - 1) + '.255')
            print("Valid range of host IP address : ",".".join(ip[0:2]) + '.' + str(temp - int(place2)) +
'.1\t-\t' + ".".join(ip[0:2]) + '.' + str(temp - 1) + '.254')
            print()
    elif className == "C":
        for i in range(num):
            print(f"\nSubnet {i} => ")
            print("Subnet Address : ", ".".join(ip[0:3]) + '.' + str(temp))
            temp += int(ip_addresses)
            print("Broadcast address : ", ".".join(ip[0:3]) + '.' + str(temp - 1))
            print("Valid range of host IP address : ",".".join(ip[0:3]) + '.' + str(temp -
int(ip_addresses) + 1) + '\t-\t' + ".".join(ip[0:3]) + '.' + str(temp - 2))
            print()
    else:
        print("In this Class, IP address is not divided into Network and Host ID")
```

```
def subnetmask(num, network_mask):
    var = '1' * int(math.log(num, 2))
    var1 = '0' * (8 - int(math.log(num, 2)))
    binary_num = var + var1
    network_mask = network_mask.split('.')
    network_mask = [i for i in network_mask if i != '0']
    network_mask.append (str(int(binary_num, 2)))
    while len(network_mask) < 5:
        network_mask.append('0')
    print('Subnet Mask – ', ".".join(network_mask[0:4]))

ip = input("Enter the IP address : ")
ip = ip.split(".")
ip = [int(i) for i in ip]
lst = findClass(ip)
networkClass = lst[0]
print("Given IP address belongs to class : ", networkClass)
ip = [str(i) for i in ip]
network_mask = lst[1]
print('Network Mask : ', network_mask)
num_subnet = int(input('\nNo. of subnets (power of 2) : '))
num_ip = int(2 ** (8 * (68 - ord(networkClass))) / num_subnet)
print('The no. of bits in the subnet id : ', int(math.log(num_subnet, 2)))
if ord(networkClass) < 68:
    print('Total no. of IP addresses possible in each subnet : ', num_ip)
Subnetting(ip, num_subnet, networkClass, num_ip)
subnetmask(num_subnet, network_mask)
```

OUTPUT:


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS D:\CN> python -u "d:\CN\ipv4.py"
Enter the IP address : 192.37.58.12
Network Id is : 192.37.58
No. of IP addresses possible : 256
Given IP address belongs to class : C
Network Mask : 255.255.255.0
No. of subnets (power of 2) : 4
The no. of bits in the subnet id : 2
Total no. of IP addresses possible in each subnet : 64
Subnet 0 =>
Subnet Address : 192.37.58.0
Broadcast address : 192.37.58.63
Valid range of host IP address : 192.37.58.1 - 192.37.58.62
Subnet 1 =>
Subnet Address : 192.37.58.64
Broadcast address : 192.37.58.127
Valid range of host IP address : 192.37.58.65 - 192.37.58.126
Subnet 2 =>
Subnet Address : 192.37.58.128
Broadcast address : 192.37.58.191
Valid range of host IP address : 192.37.58.129 - 192.37.58.190
Subnet 3 =>
Subnet Address : 192.37.58.192
Broadcast address : 192.37.58.255
Valid range of host IP address : 192.37.58.193 - 192.37.58.254
Subnet Mask - 255.255.255.192
PS D:\CN>

EXPERIMENT - 8

AIM: Use basic networking commands in linux (ping, tracert, nslookup, netstat, ARP, RARP, ip, ipconfig, dig, route)

THEORY:

- ifconfig: This command is utilized in network inspection, initializing the interface, enabling disabling ip address and configuring an interface with an IP address.
  Syntax: ifconfig

- ping: It is used for network troubleshooting. It checks network connectivity between two different nodes.
  Syntax: ping <destination>

- traceroute: It identifies the delay and identifies or decides pathway to our target.
  Syntax: traceroute <destination>

- netstat: It is a short network statistics. It displays various network related information such as network connections, routing tables, interface statistics, multicast membership, etc.
  → netstat -a : shows the state of all the sockets.
  → netstat -at: List all TCP ports.
  → netstat -au: List all UDP ports.
  → netstat -l : Lists only the listening ports.
  → netstat -s : Lists statistics for all ports.

- nslookup : It is an old edition of dig command. It is useful for getting information from DNS server. It is also used to troubleshoot DNS related problems.
  Syntax: nslookup <domain-names

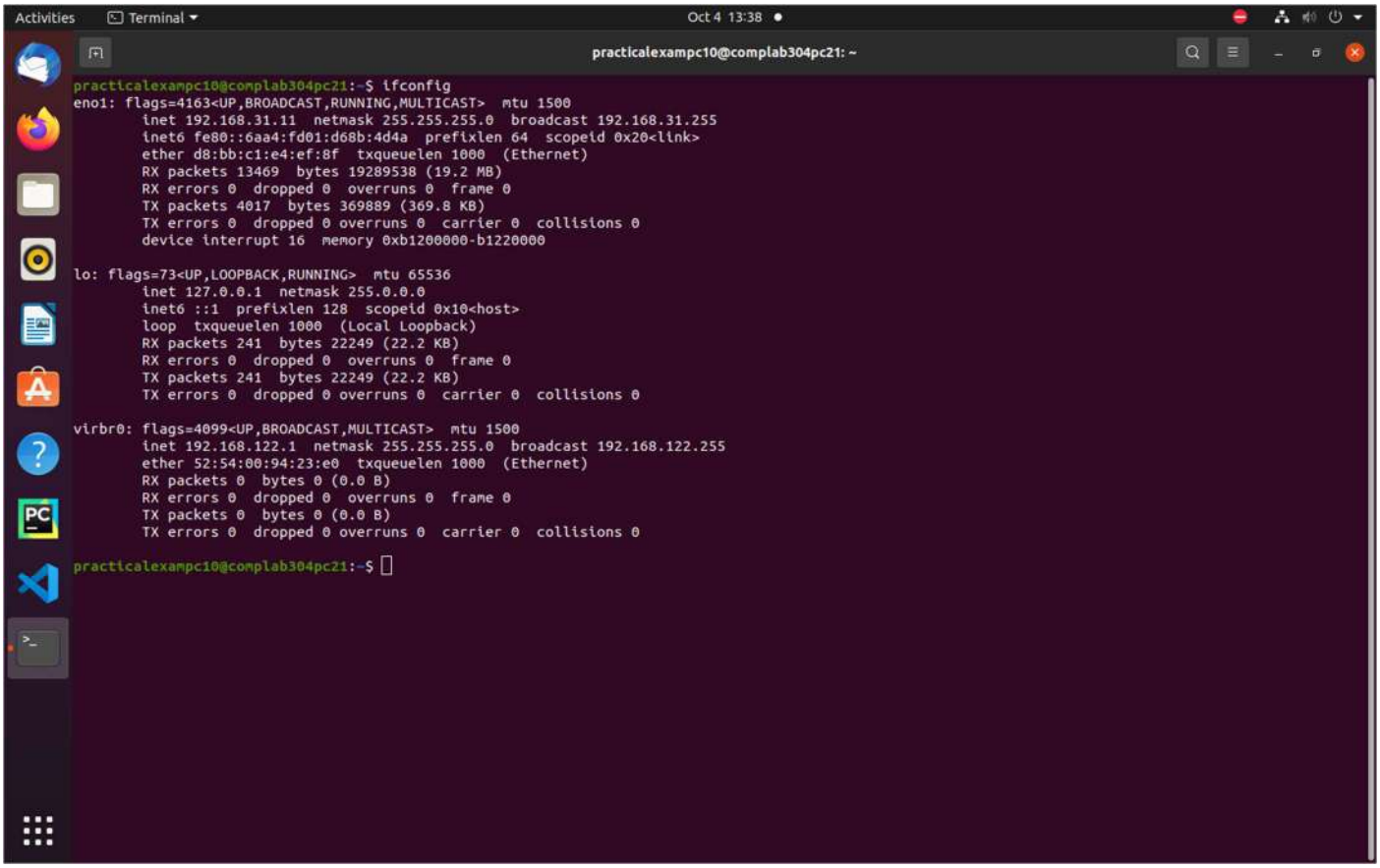
- dig : The Domain Information Graper is the improved version of nslookup. It is used for verifying and troubleshooting DNS problems and to perform DNS lookup.

- arp : Address Resolution Protocol is used to restore/resolve the ip address of a system to mac address and hence it works between Data Link Layer and Network Layer.

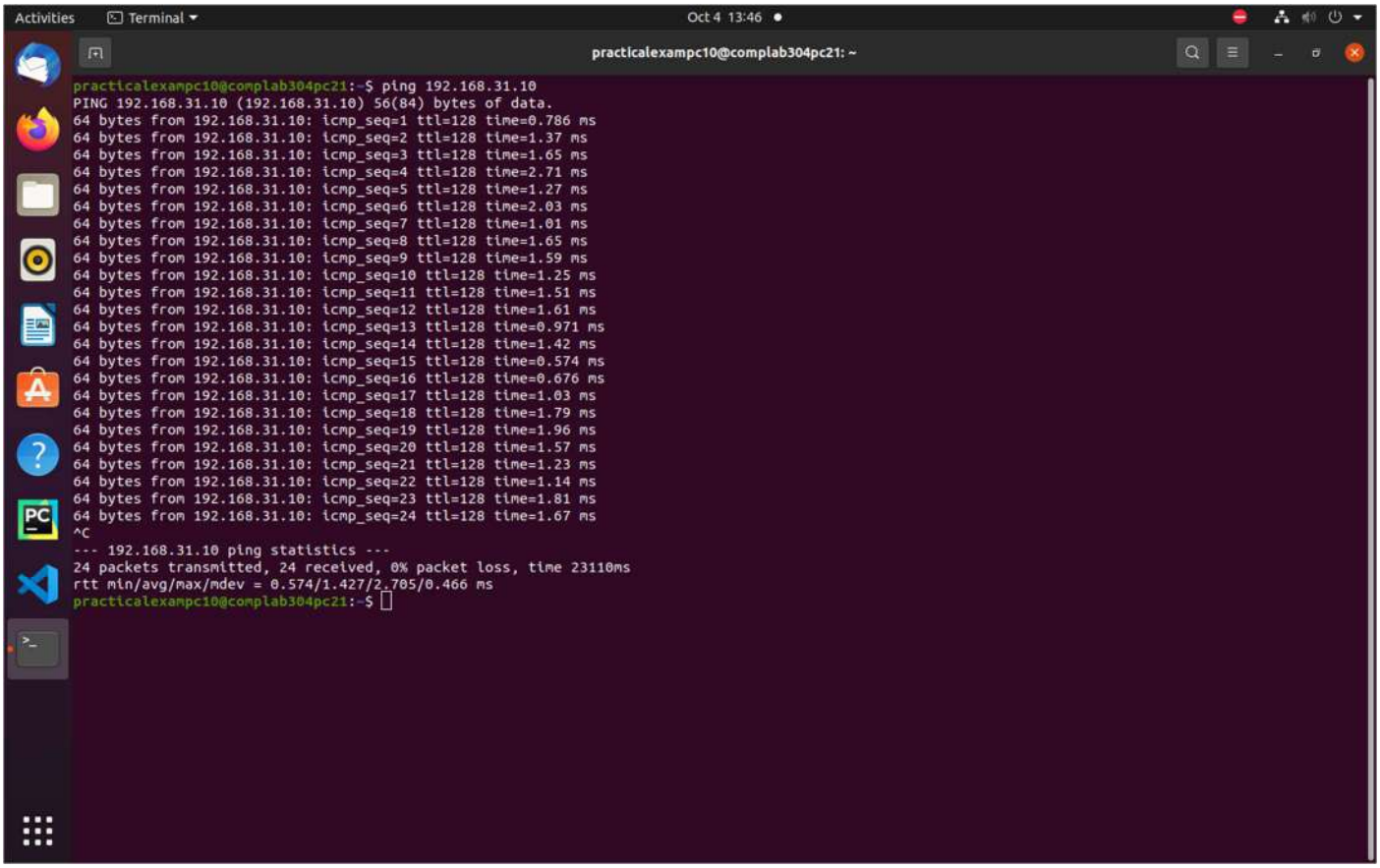
- rarp : Reverse Address Resolution Protocol is used to resolve the mac address of a system to ip address and hence it works between Data-Link layer and Network Layer.

- ip : It is the updated and latest version of ifconfig command. The command provide the information of every network, such as ifconfig. It can also be used to get information about a particular interface. It is also used to show or manipulate routing devices and tunnels.

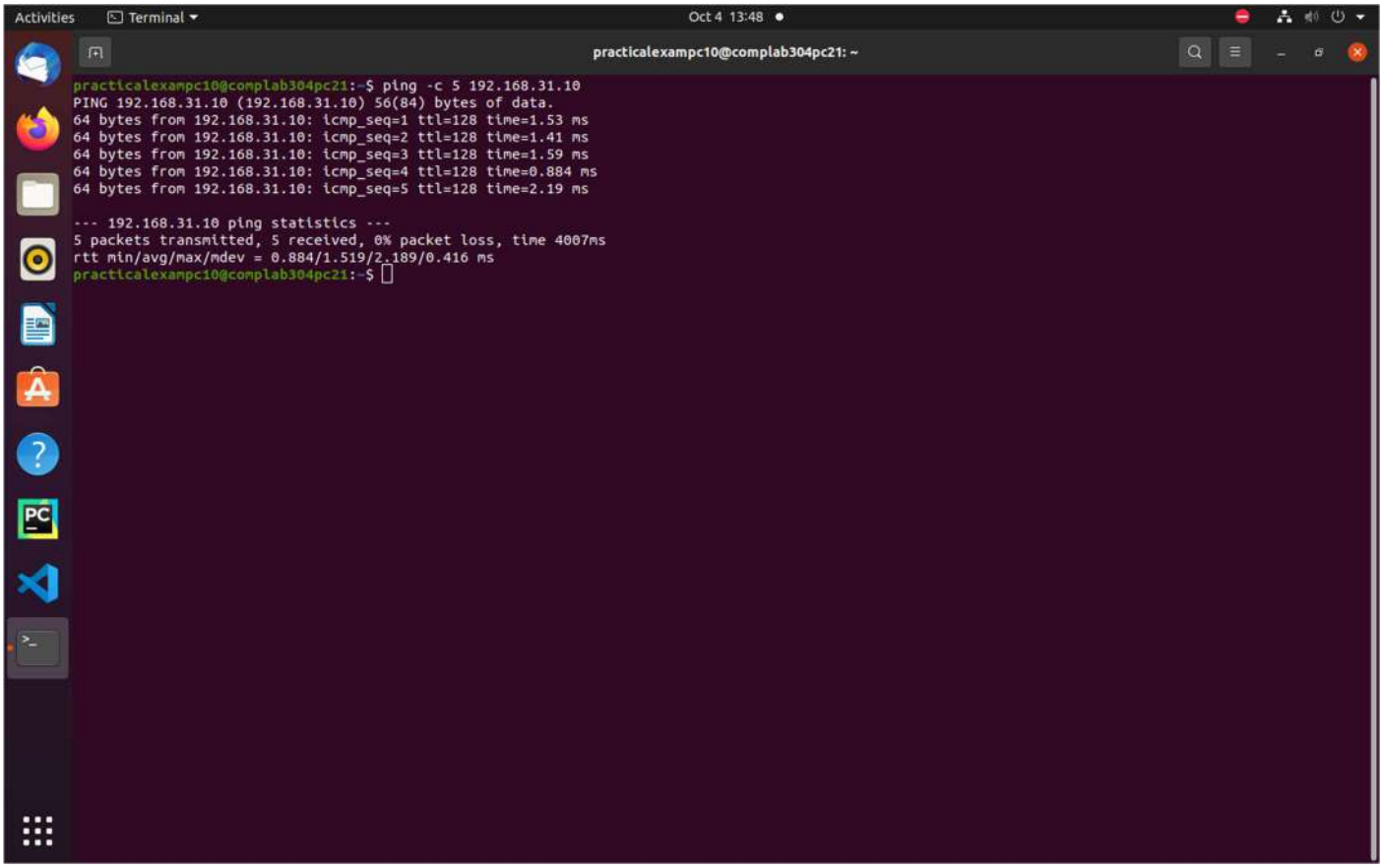
1. ifconfig



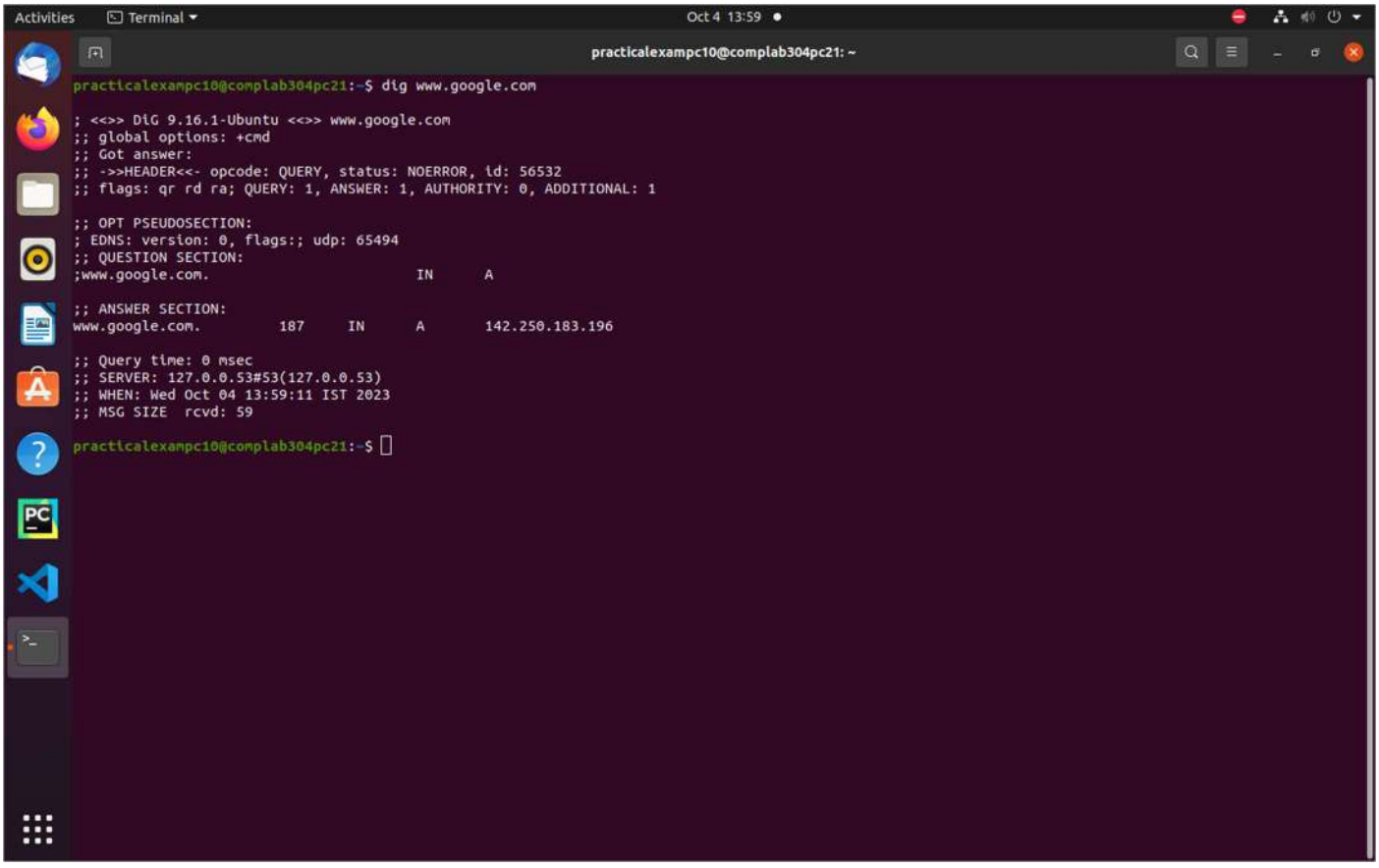
2. ping without count



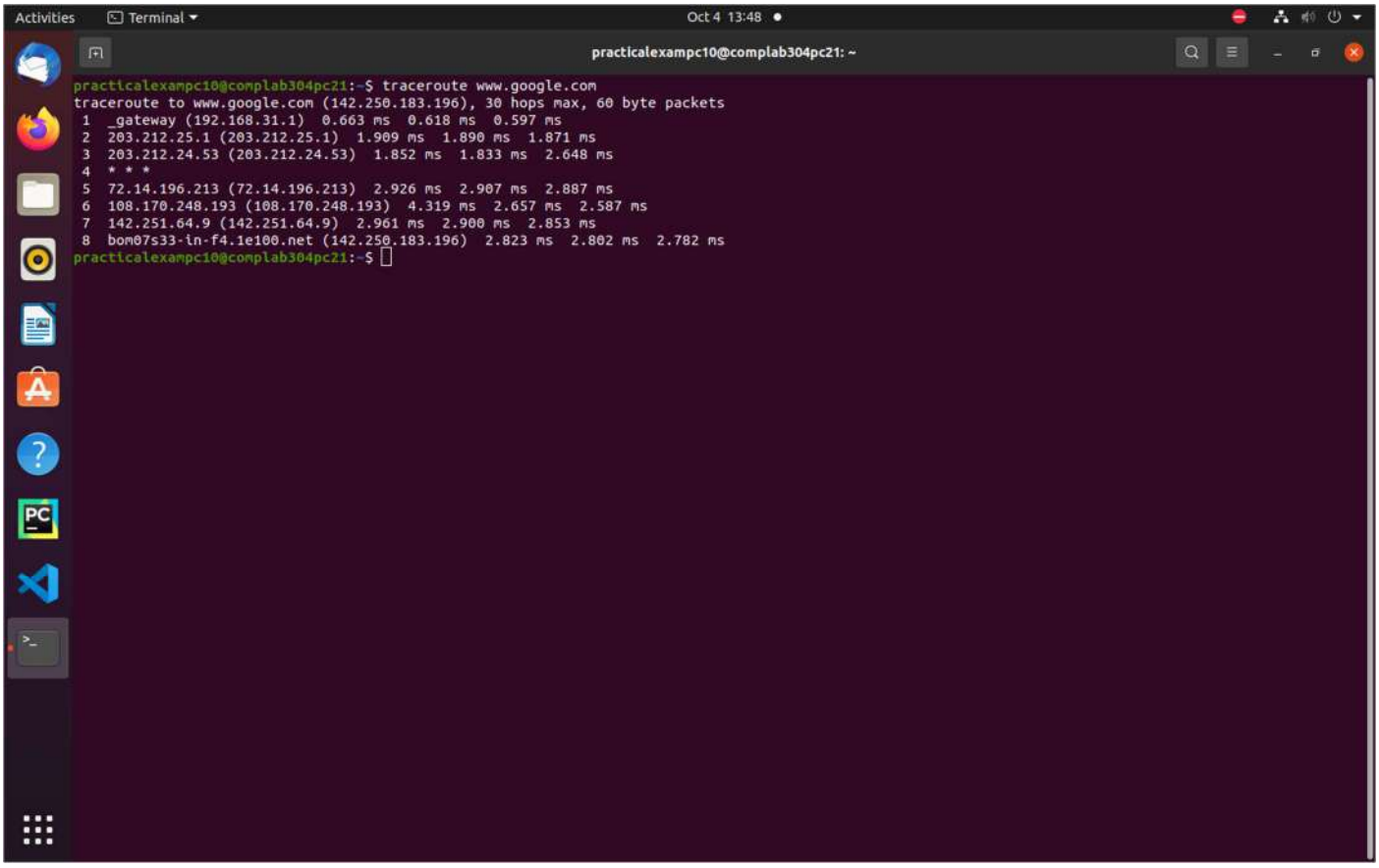
3. ping with count



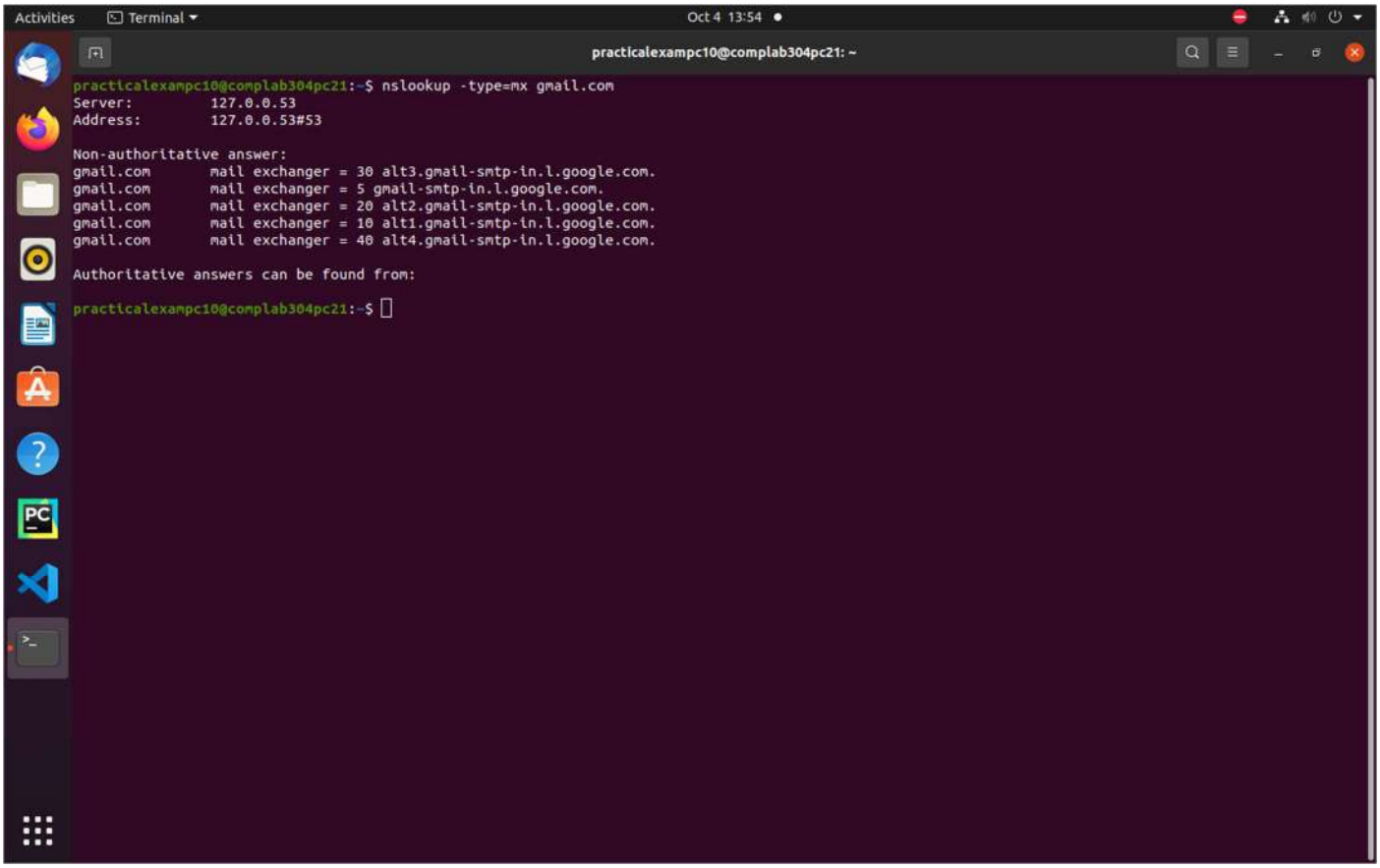
4. dig



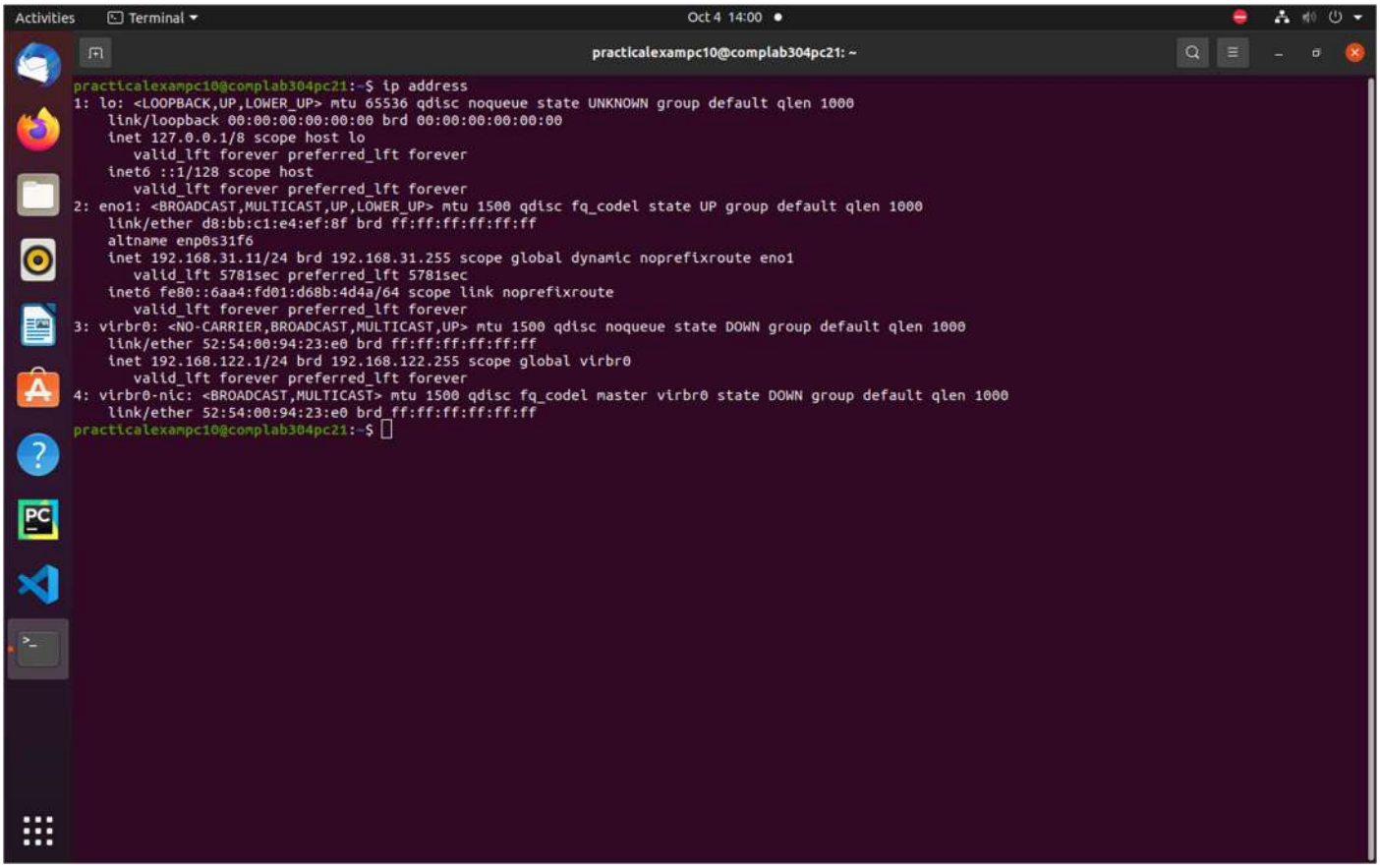
5. traceroute



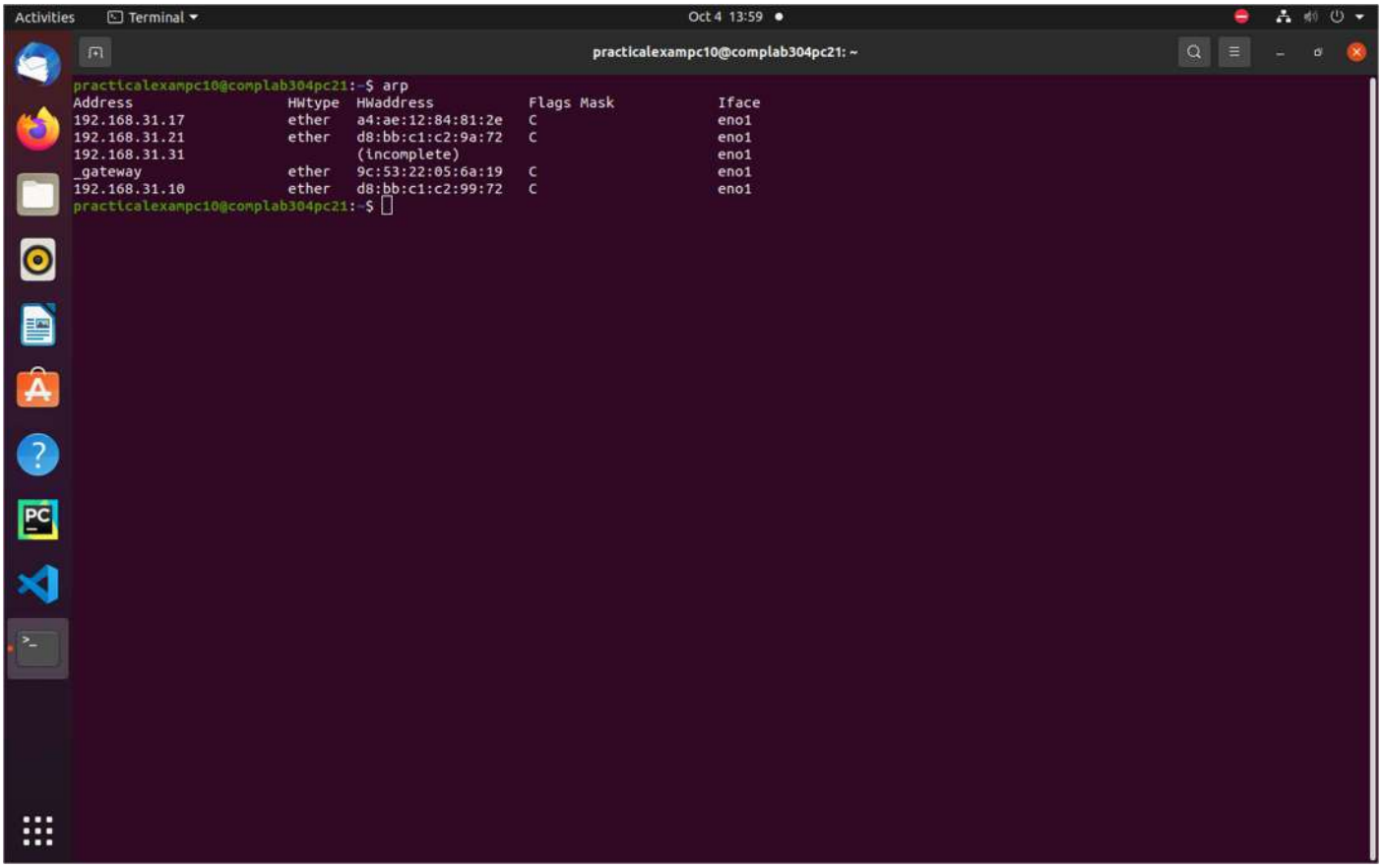
6. nslookup



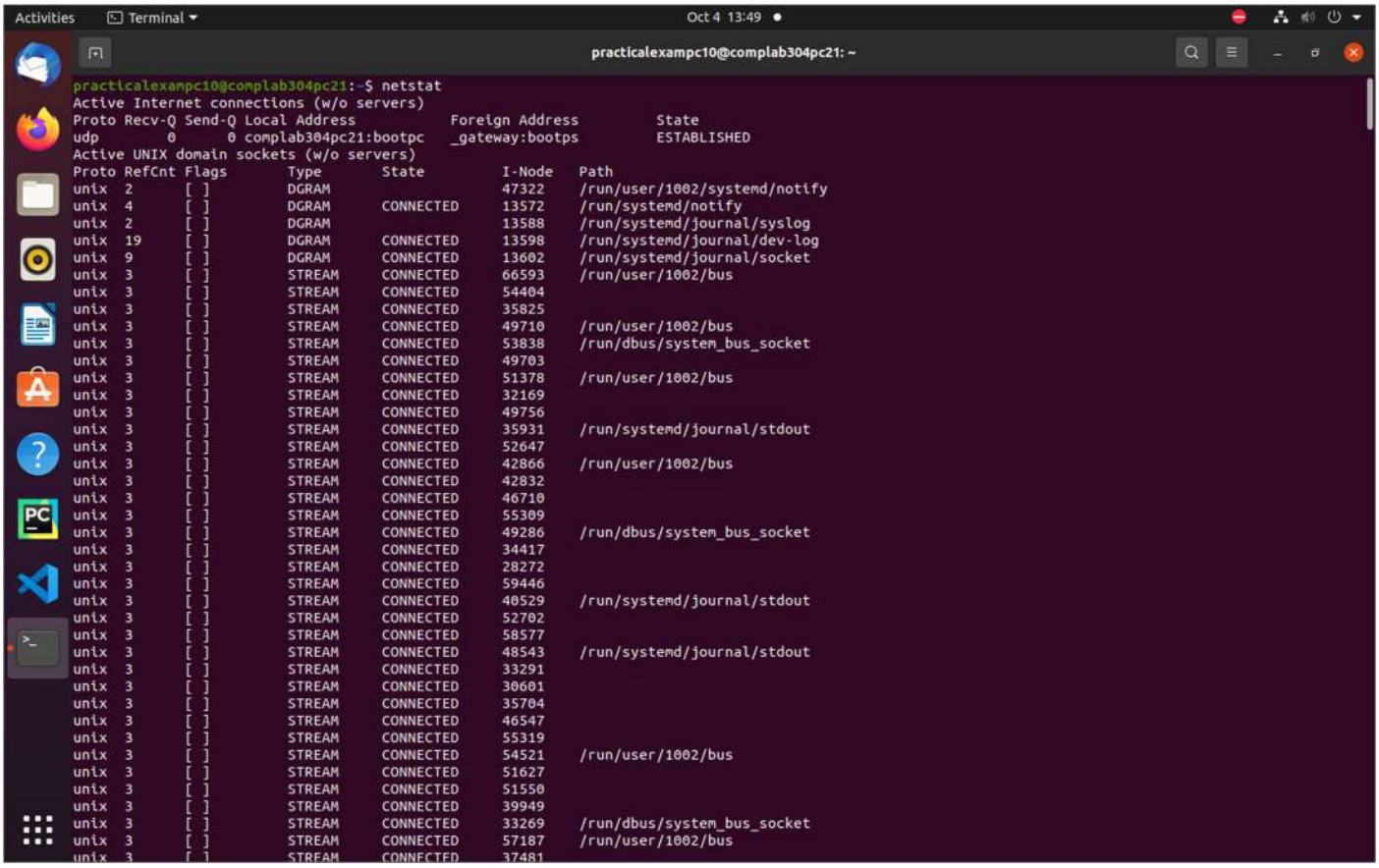
7. ip address



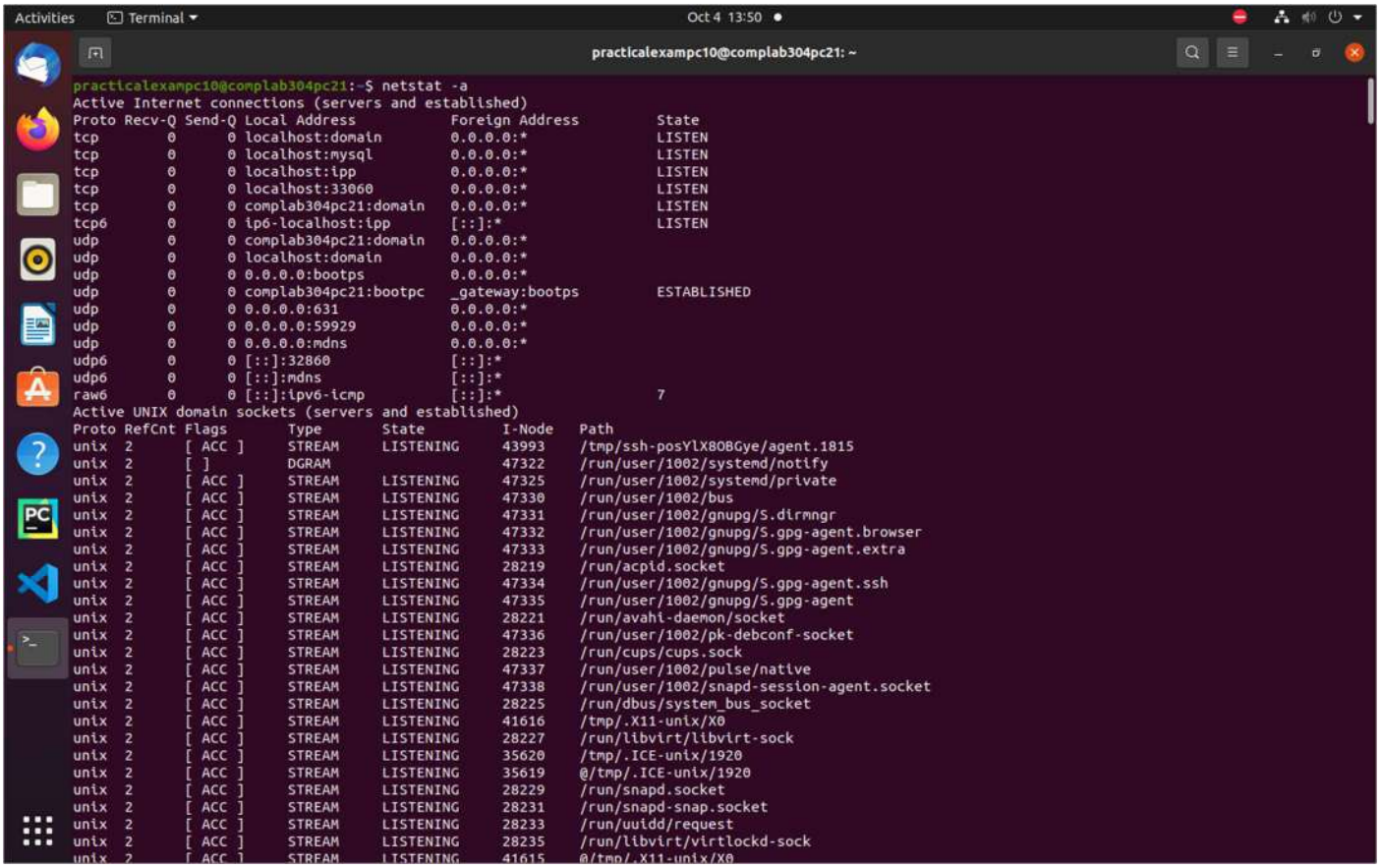
8. arp



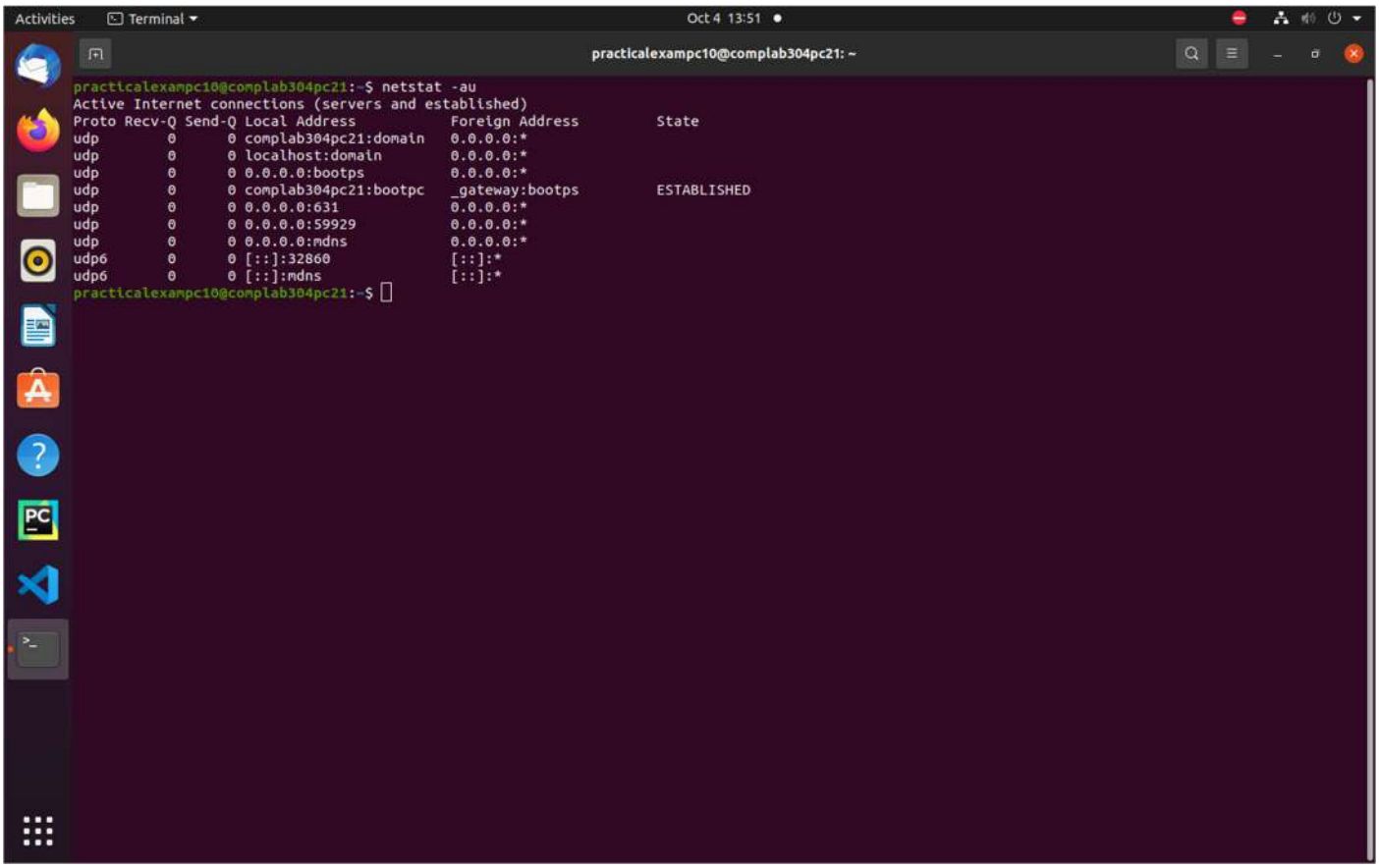
9. netstat



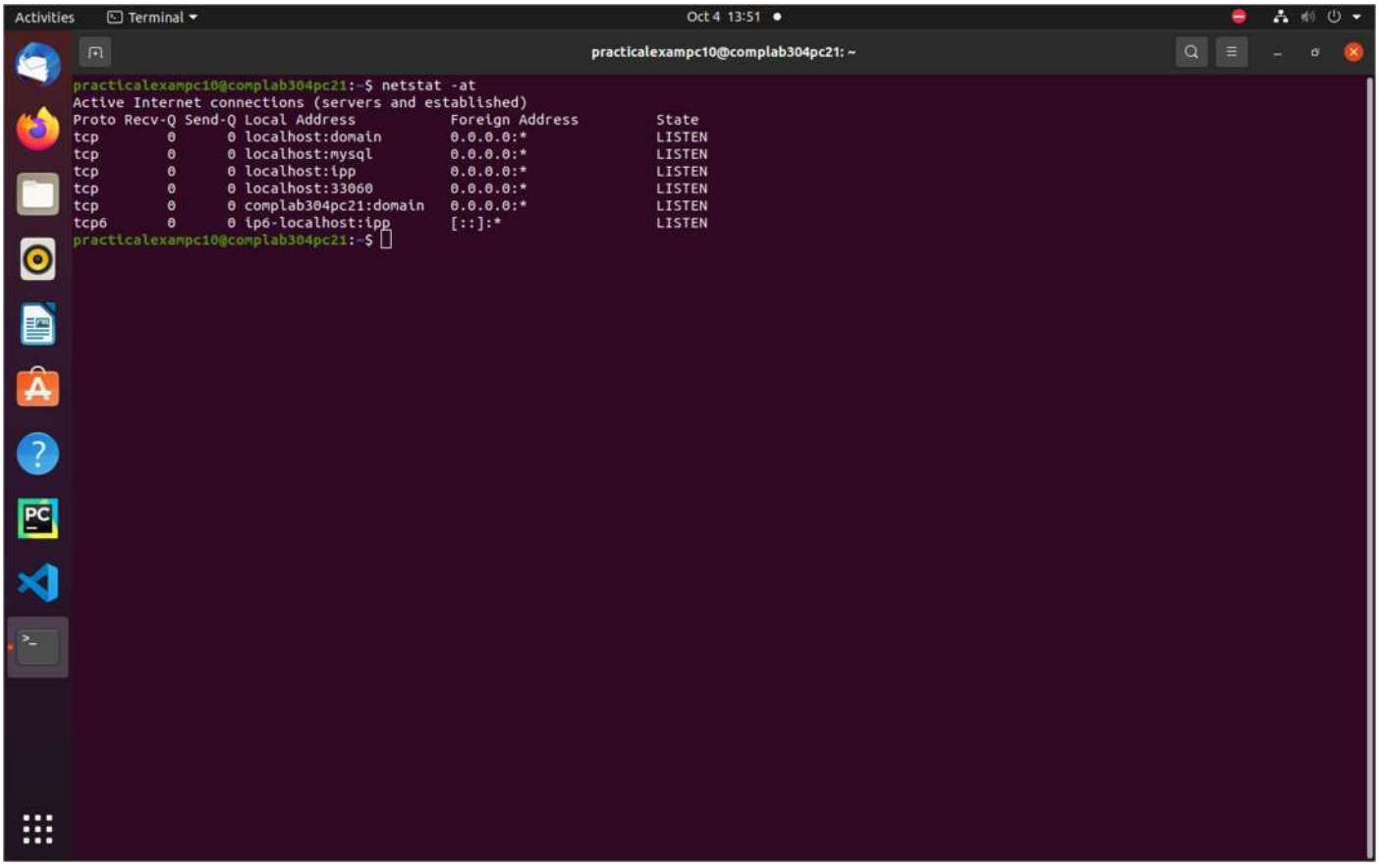
10. netstat -a



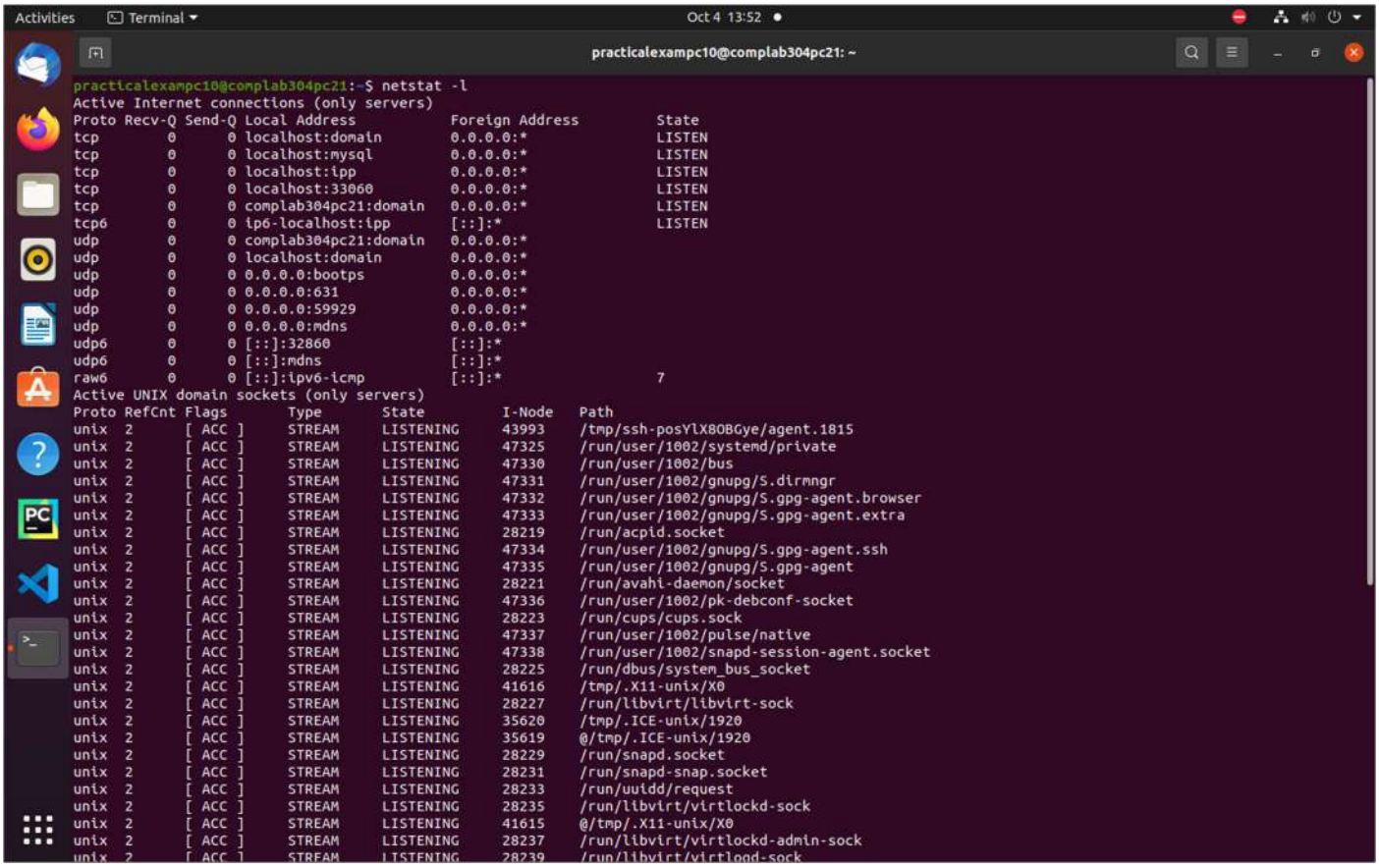
11. netstat -au



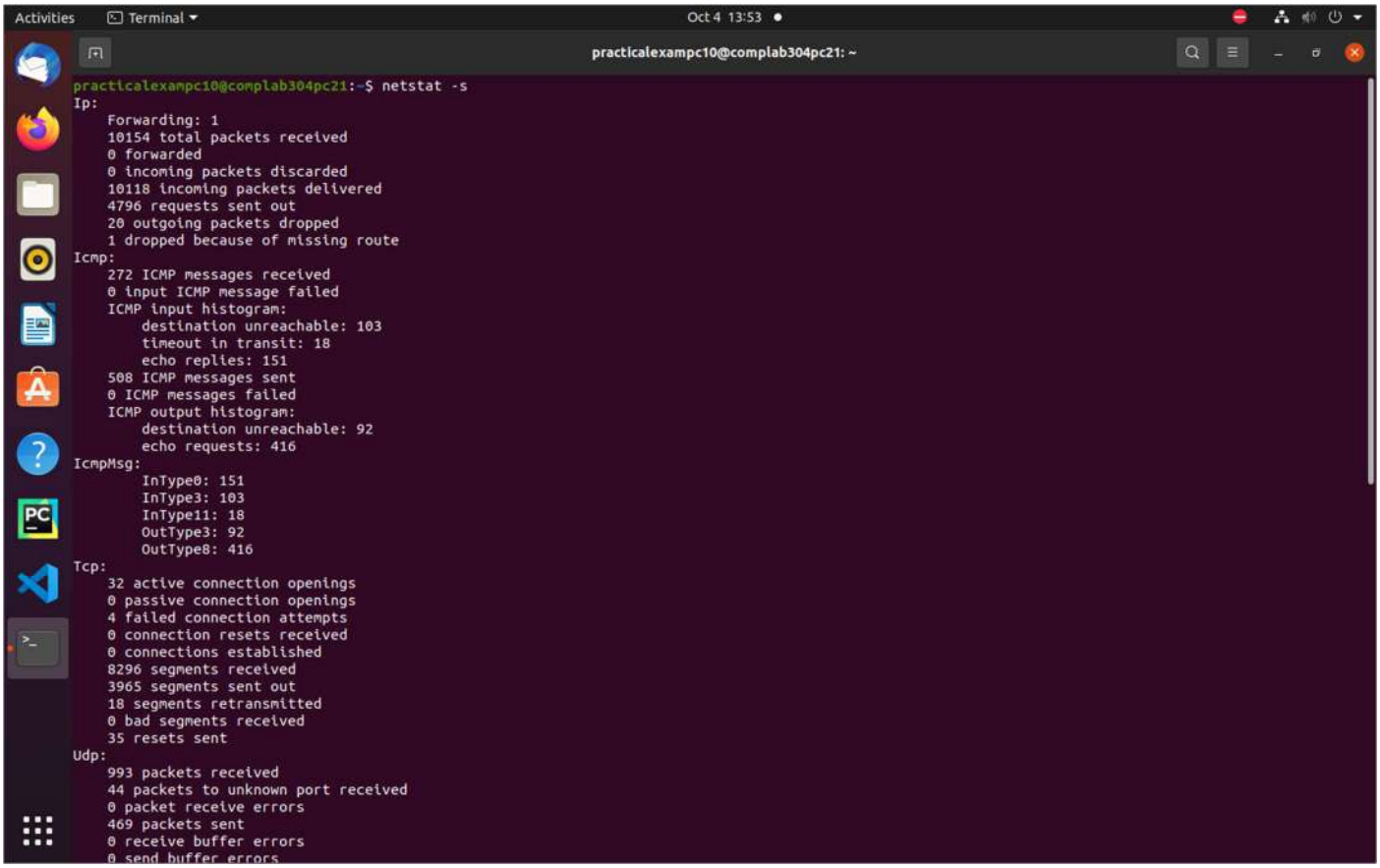
12. netstat -at



13. netstat -l



14. netstat -s



EXPERIMENT-9

AIM: Use wireshark to understand the operation of TCP/IP layers:
Ethernet Layer: Frame header, Frame size etc.
Data Link Layer: MAC address, ARP (IP and MAC)
Network Layer: IP Packet (header, fragmentation), ICMP (Query and Echo)
Transport Layer: TCP Ports, TCP handshake segments etc.
Application Layer: DHCP, FTP, HTTP header formats.

THEORY:
wireshark is a popular open-source network protocol analyser used for capturing and inspecting packets on a network. It is a powerful tool for network administrators security professionals and students.

The objective of this experiment is to use the wireshark, a network protocol analyser, to capture and analyze network traffic to understand the operations and various layers and TCP/IP protocol suite.

1. Ethernet Layer: This layer deals with frames which are packets of data at the ethernet layer. one can observe the size of the ethernet frames in wireshark. Ethernet frames have headers containing information like source and destination MAC address.

2. Data Link layer: the data link layer deals with MAC

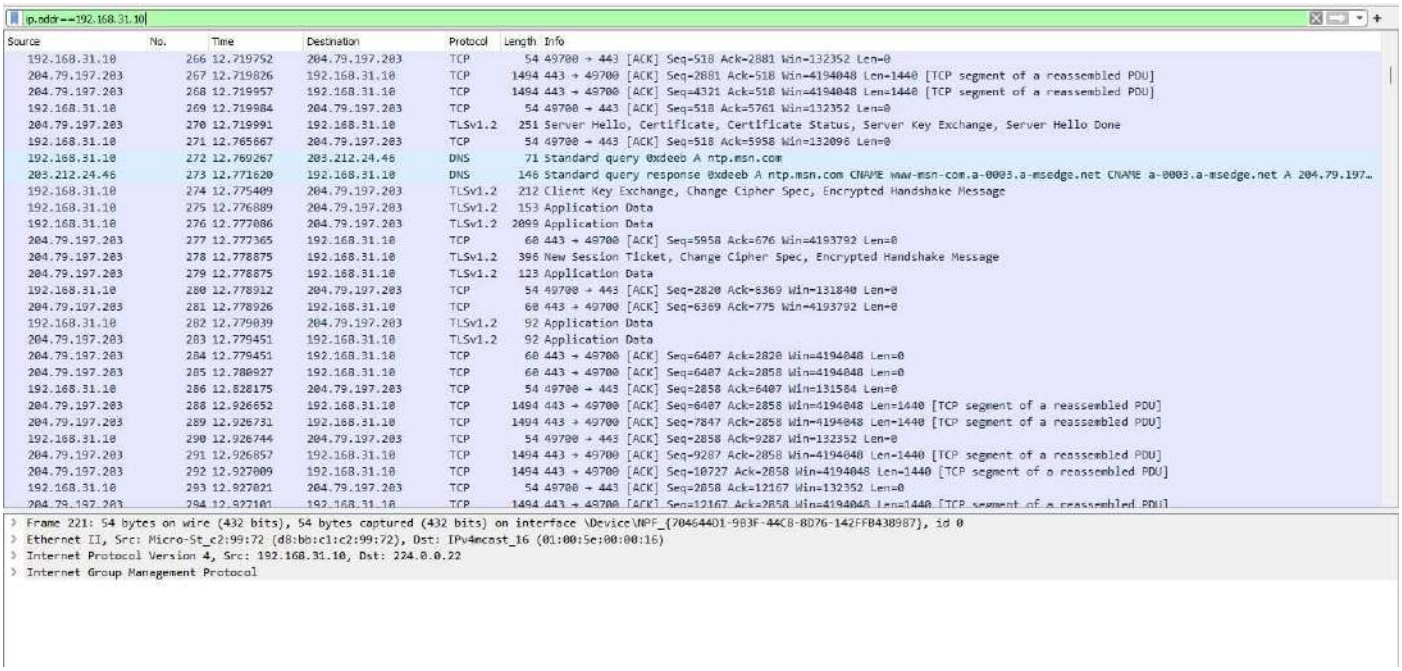
addresses. Wireshark will show you the MAC address of the device communicating in the network. User can capture ARP packets to see how device map IP addresses to MAC addresses.

3. Network Layer: This layer involves a lot of processes. It involves IP packets. Wireshark displays IP header with information like source and destination IP addresses. ICMP packet can be captured to observe network troubleshooting messages, like ping requests and replies.

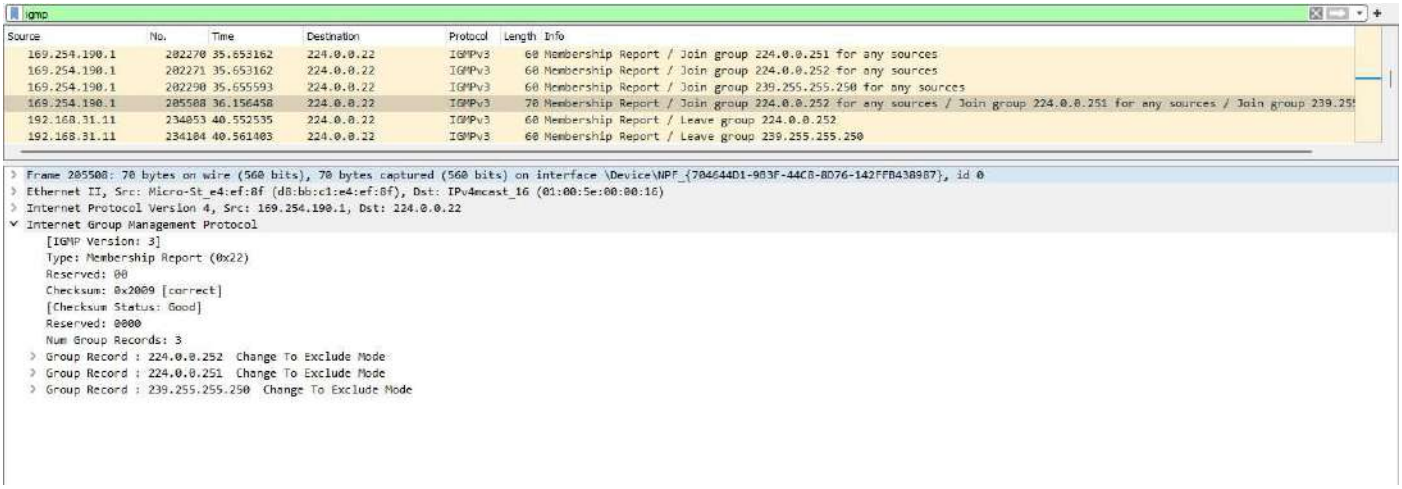
4. Transport Layer: User can see which ports are used by applications (Eg: web browsers, email clients) for communication. By capturing TLP traffic, you can see the three way handshake process, which is how two devices establish a connection.

Protocol Name	color in Wireshark
Error in Packets	Black / Dark shades
TCP	Light Purple
UDP	Light Blue
HTTP	Light Green
ARP	Peach
ICMP	Pink
SMB	Light yellow
Routing	Dark yellow
TCP SYN, FIN, ACK	Dark Gray
TCP RST	Red

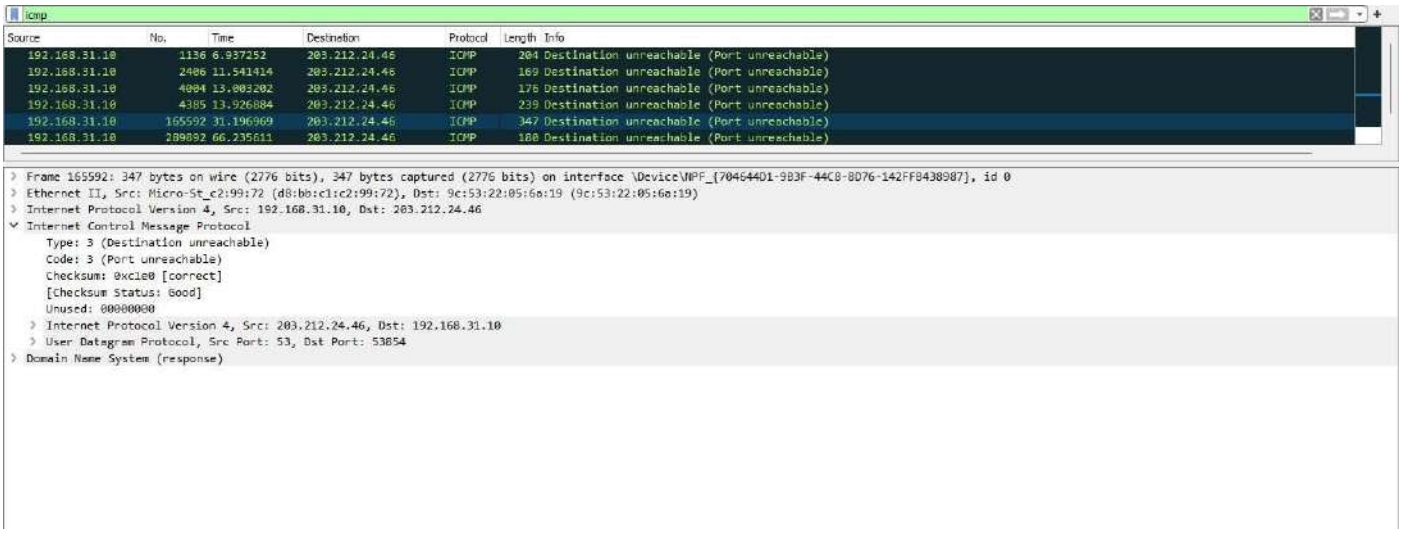
1. Internet Protocol (IP)



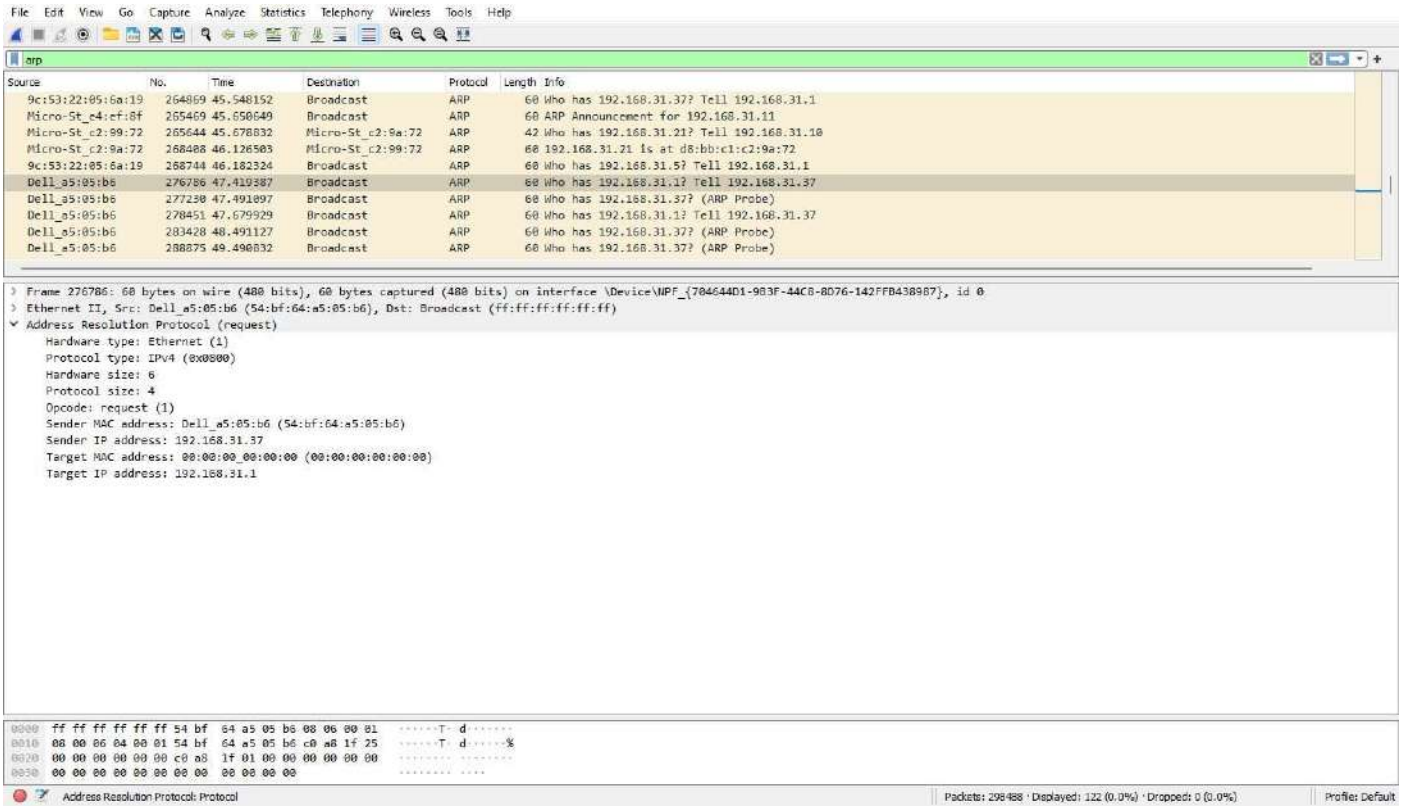
2. Internet Group Management Protocol (IGMP)



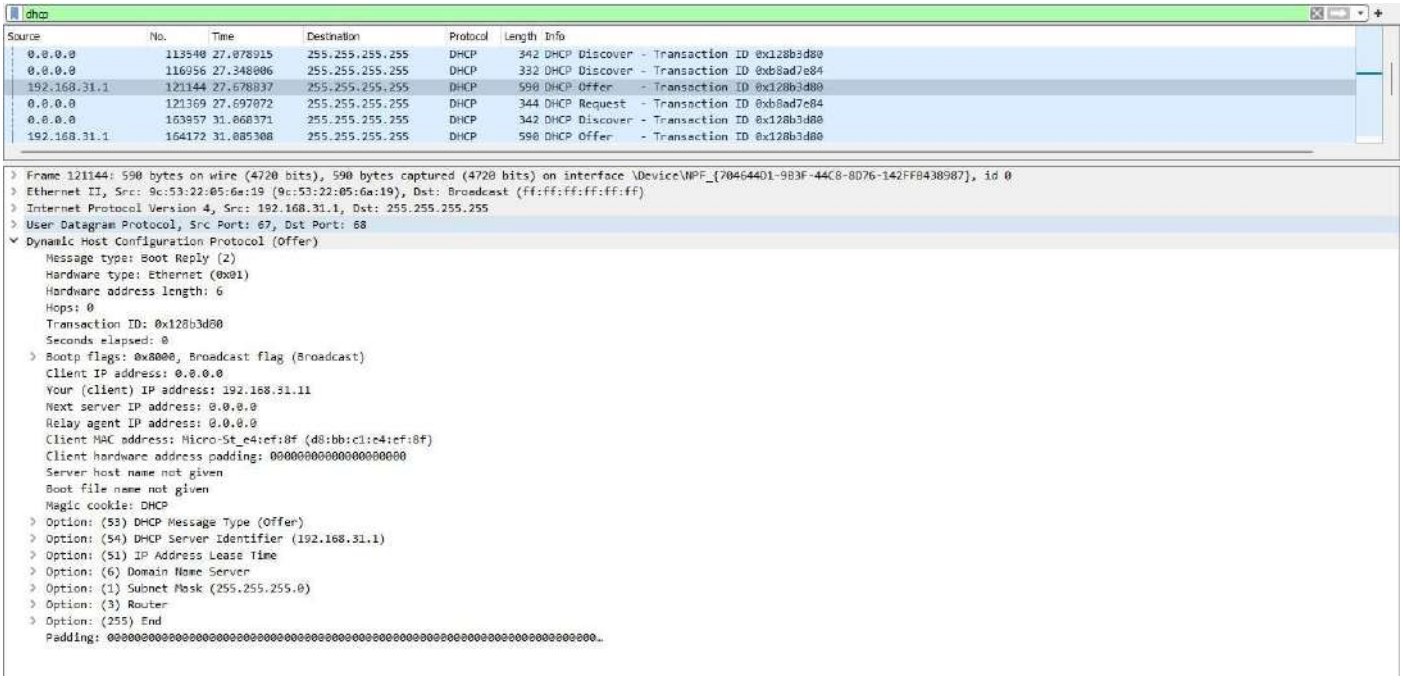
3. Internet Control Message Protocol (ICMP)



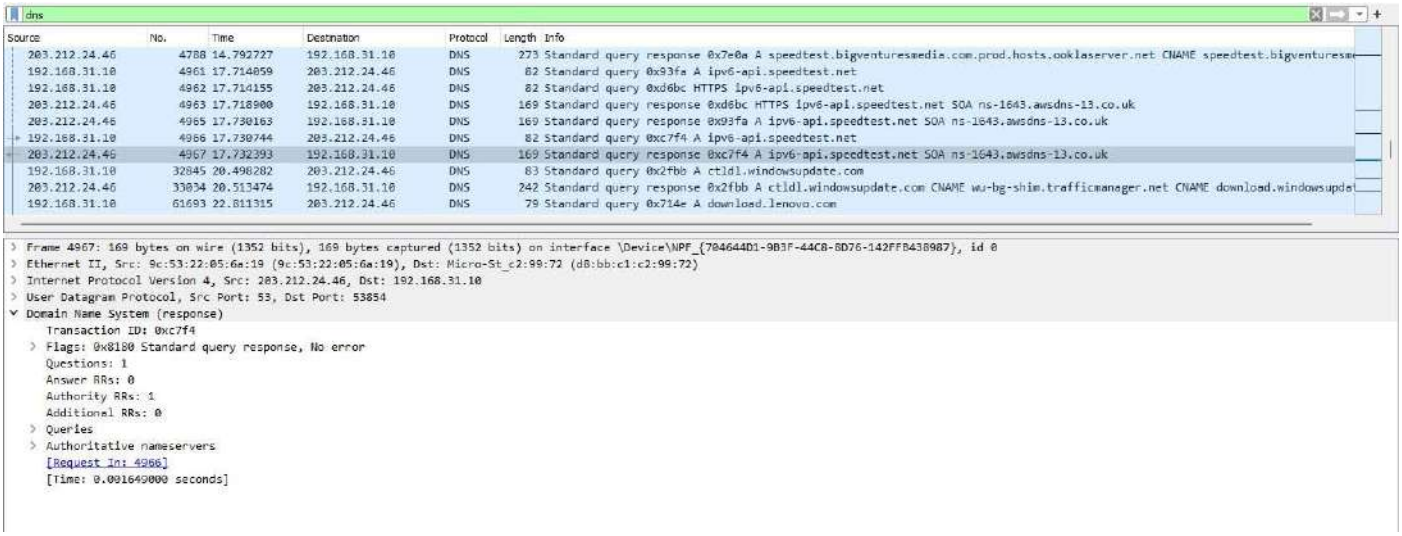
4. Address Resolution Protocol (ARP)



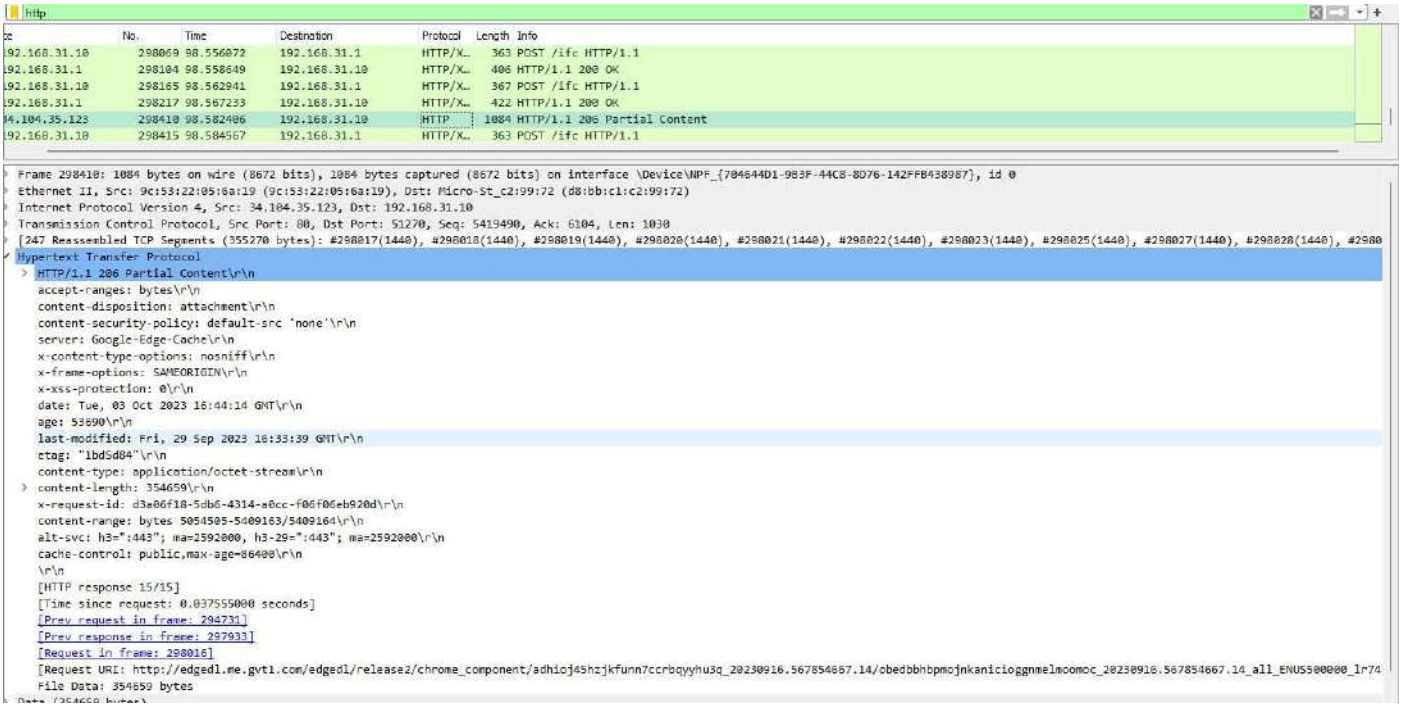
5. Dynamic Host Configuration Protocol (DHCP)



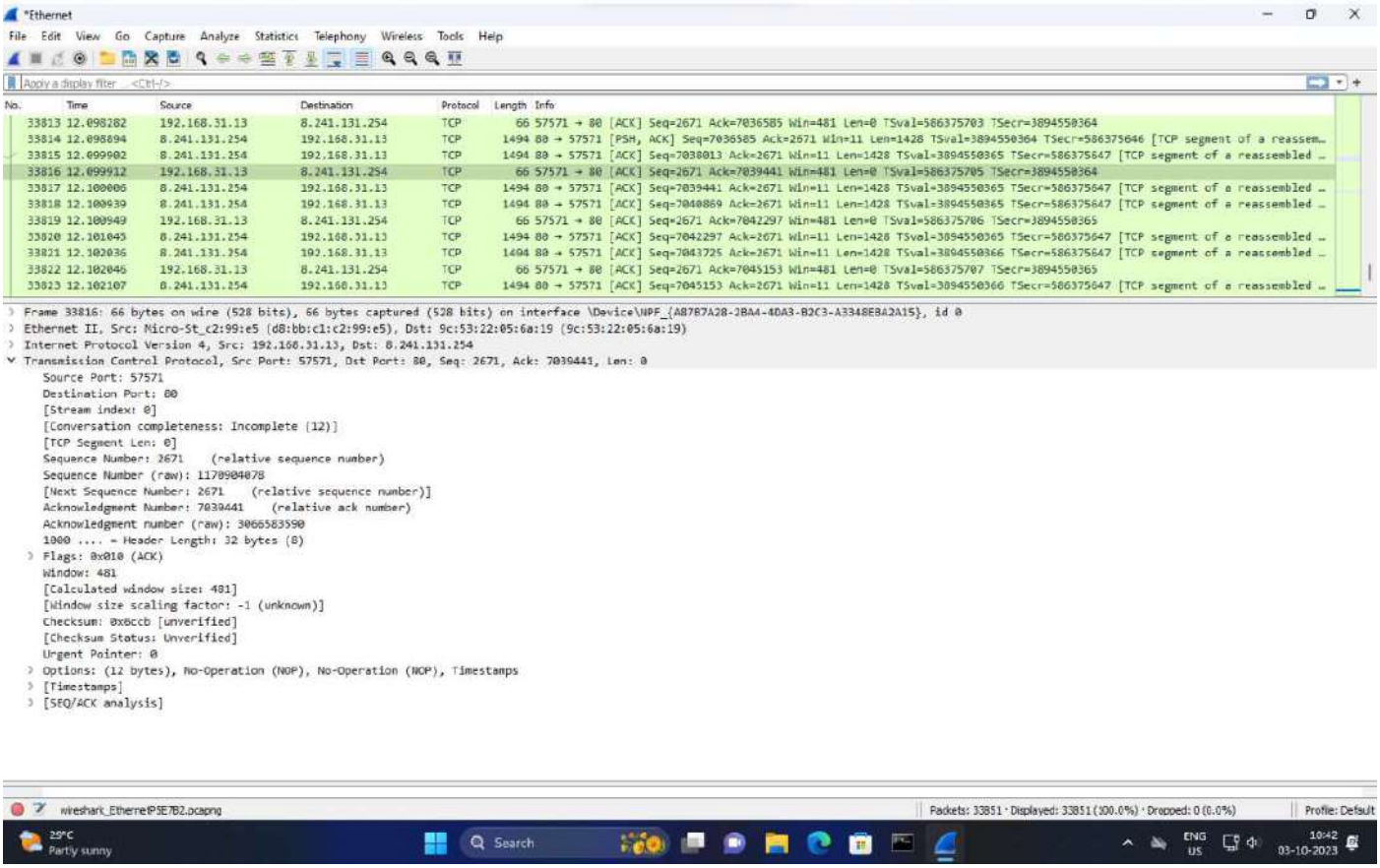
6. Domain Name System (DNS)



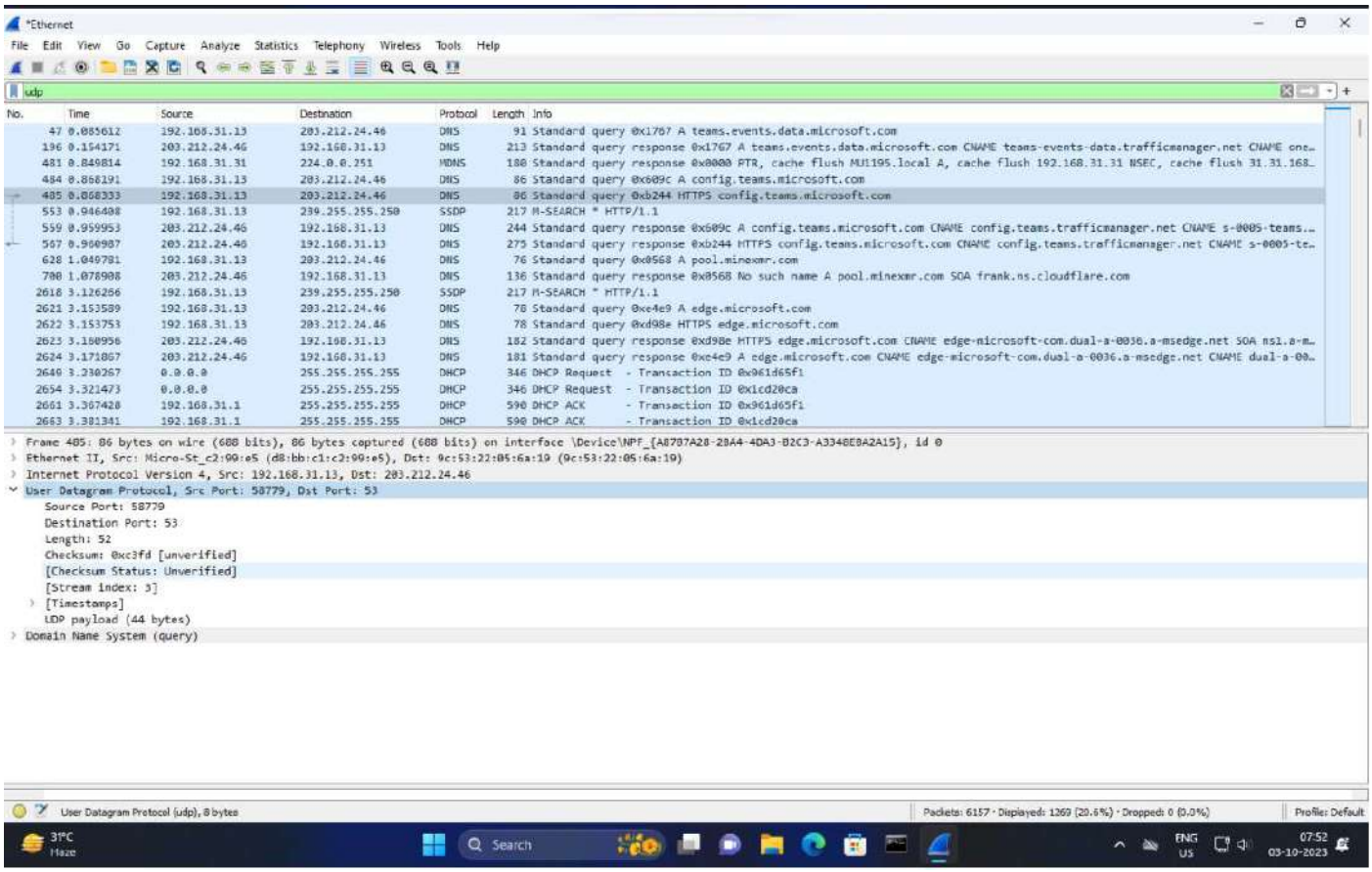
7. Hypertext Transfer Protocol (HTTP)



8. Transmission Control Protocol (TCP)



9. User Datagram Protocol (UDP)



EXPERIMENT : 10

AIM : Implement Socket Programming using Java.

THEORY : Java socket programming is used for communication between the applications running on different JRE.
Java socket programming can be connection-oriented or connection-less.
Socket and ServerSocket class are used for connection-oriented socket programming and DatagramPacket classes are used for connection-less socket programming.

The client in socket programming must know two information:
a. IP Address of Server
b. Port Number.

Here, we are going to implement two-way client and server information communication. In this application, client sends a message to server, server reads the message. In the same way, server sends a message to the client and the client can read the message. Here, two classes are being used : Socket and ServerSocket.

The socket class is used to communicate client and server. Through this class, we can read and write messages. The ServerSocket class is used at server side. The accept() method of ServerSocket class blocks the console until the client is connected. After the successful connection of client, it returns the instance of Socket at the server-side.

Socket Class :

A socket is simply an endpoint for communication between the machines. The Socket class can be used to create Socket.

Server Socket class:

The server socket class can be used to create a server socket. This object is used to establish communication with the clients.

Creating Server: To create the server application, we need to create the instance of server Socket Class. Here, we are using 6666 port number for the communication between the client and the server. You may also choose any port number. The accept() method waits for the client. If the client connects with the given port number, it returns an instance of socket.

```
ServerSocket ss = new ServerSocket(6666);
Socket s = ss.accept(); (Establishes connection)
```

Creating Client : To create the client application, we need to create the instance of a Socket class. Here, we need to pass the IP address or hostname of the server and a port number. Here, we are using "localhost" because our server is running on the same computer system.

```
Socket s = new Socket("localhost", 6666);
```

To use these classes we need to import :

```
import java.io.*;
import java.net.*;
```

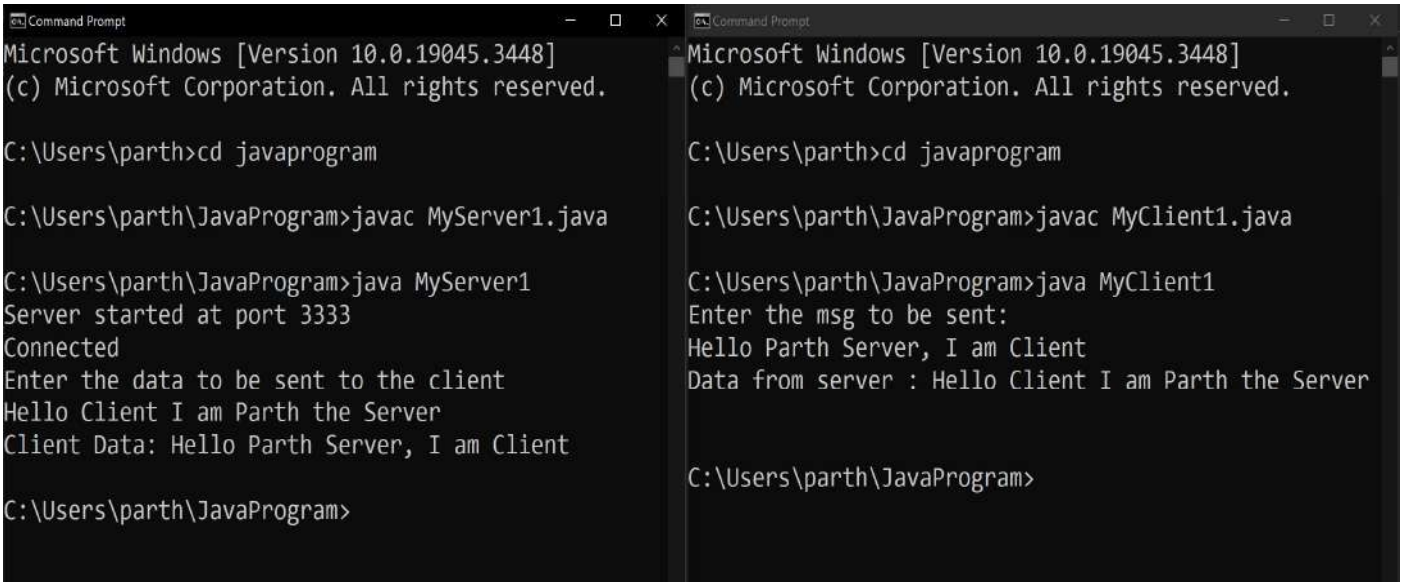
SERVER PROGRAM:

```
import java.io.*;
import java.net.*;

public class MyServer1
{
	public static void main(String args[])throws IOException
	{
		System.out.println("Server started at port 3333");
		ServerSocket ss = new ServerSocket(3333);
		Socket s = ss.accept(); //This will establish connection between client and server
		System.out.println("Connected");
		BufferedReader br1 = new BufferedReader(new InputStreamReader(System.in));
		System.out.println("Enter the data to be sent to the client");
		String nickname = br1.readLine();
		BufferedReader br = new BufferedReader(new InputStreamReader(s.getInputStream()));
		String msg = br.readLine();
		System.out.println("Client Data: "+msg);
		OutputStreamWriter os = new OutputStreamWriter(s.getOutputStream());
		PrintWriter out = new PrintWriter(os);
		out.println(nickname);
		out.flush();
		s.close();
		ss.close();
	}
}
```

CLIENT PROGRAM:

```
import java.net.*;
import java.io.*;
public class MyClient1
{
	public static void main(String args[])throws IOException
	{
		Socket s = new Socket("localhost",3333);
		System.out.println("Enter the msg to be sent: ");
		BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
		String str = br.readLine();
		OutputStreamWriter os = new OutputStreamWriter(s.getOutputStream());
		PrintWriter out = new PrintWriter(os);
		out.println(str);
		os.flush();
		BufferedReader br1 = new BufferedReader(new InputStreamReader(s.getInputStream()));
		String nickname = br1.readLine();
		System.out.println("Data from server : "+nickname);
		s.close();
	}
}
```

OUTPUT:



CN-C12-32

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Assignment no:1

1. Write a short note on following devices:-
   a. Repeater
   b. Hub
   c. Bridge
   d. Switch
   e. Router
   f. Gateway
   g. Modem

a. **REPEATER:** A repeater is a powerful network device that regenerates an incoming signal from the sender before retransmitting it to the reciever. It is also known as a signal booster, and it helps in extending the coverage area of network. Since it is a hardware device, it only works at the physical level layer of the OSI (Open System Interconnection) model. Therefore, it is also known as a layer-1 device. It is a two port device.

Functions: i) Repeaters do not amplify the signal. They strengthen the signal by regenerating it.
ii) When the incoming signals are attenuated, it copies them bit by bit and retransmits them at their original strength.
iii) It is used for longer distance data transmission without comprimising data security and quality.

Types: 1. According to the type of signals that they regenerate, repeaters can be classified into two categories:-
   a. **Analog Repeaters:** - They can only amplify signal.
   b. **Digital Repeaters:** - They can reconstruct a distorted signal.

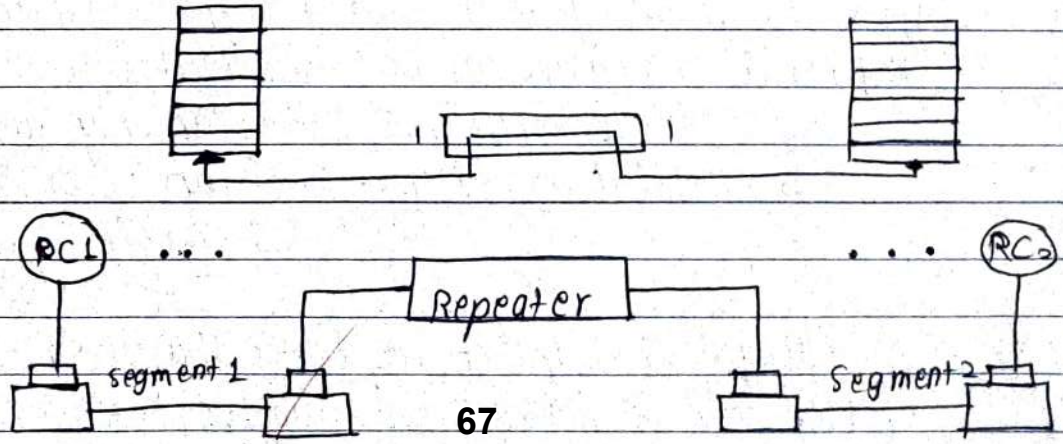
2. According to the types of networks that they connect, repeaters can be categorised into two types:-

a. Wired Repeaters :- They are used in wired LANs.
b. Wireless Repeaters :- They are used in wireless LANs and cellular networks.

3. According to the domain of LANs they connect, repeaters can divided into two categories :-
   a. Local Repeaters :- They connect LAN segments seperated by small distance.
   b. Remote Repeaters :- They connect LANs that are far from each other.

Advantages :- i) Repeaters are simple to install and can easily extend the coverage area of networks.
ii) They are cost effective.
iii) They can connect signals using different types of cables.

Disadvantages :- i) Repeaters cannot connect dissimilar networks.
ii) They cannot differentiate between actual signal and noise.
iii) They cannot reduce network traffic or congestion.
iv) Most networks have limitations upon the number of repeaters that can be deployed.



B. Hub: A hub is a physical layer networking devices which is used to connect multiple devices in a network. They are generally used to connect computers in a LAN. Since it works as physical layer, it is also known as layer 1 device. It is a multiport repeater. Therefore, it can connect multiple systems to a network.

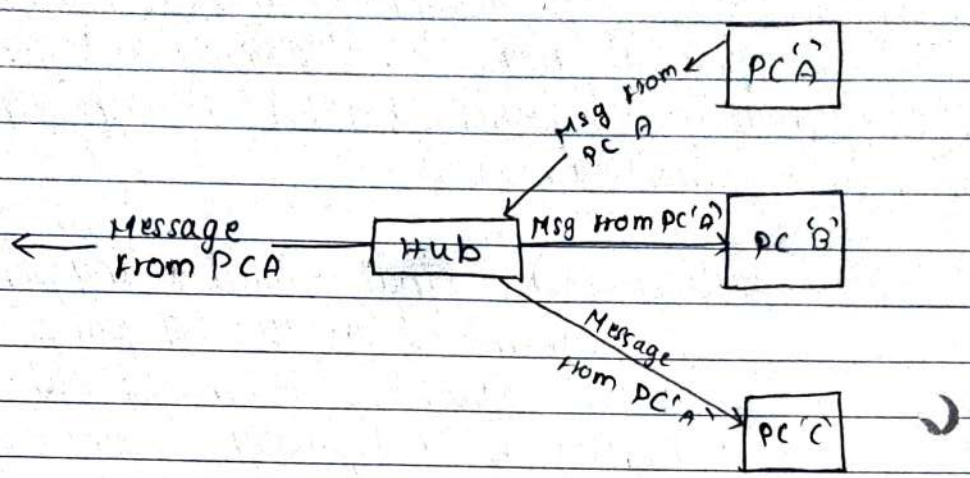
Functions :- i) A signal recieved at any port on the hub is retransmitted on all other ports.
ii) Network segments that employ hubs are often described as having a star topology, in which the hub forms the wiring centre of the star.

Types of Hub :- i) Active Hub :- They have a power supply for regenerating, and amplify the signals.
ii) Passive Hub : Used to connect signals of different network cables as they do not have any computerized element.
iii) Intelligent Hub : These are smarter hubs which consists of Management Information Base (MIB) software that helps in analyzing and troubleshooting networks.

Advantages : i) Less Expensive
ii) Does not impact network performance.
iii) Support different network media.
iv) Easily connects with different media.

Disadvantages : i) It cannot find the best/shortest path of the network.
ii) No mechanism for traffic detection.
iii) No mechanism for data filteration.
iv) Not capable of connecting to different

network topologies like token, ring, ethernet, etc.



c. **Bridge:** A bridge in a computer network is a device used to connect multiple LANs together with a Local Area Network (LAN). The mechanism of network aggregation is known as bridging. The bridge is a physical or hardware device but operates at the OSI model's data link layer and is also known as layer two device.

Functions: i) Bridge can be used to connect LANs of different layers.
ii) Bridges are also used to connect 2 LANs but they have to be identical.
iii) Bridges only have two ports. Thus, they connect only two LAN networks.
iv) Bridges perform the forwarding functions. If a packet is recieved on a bridge, it forwards it.

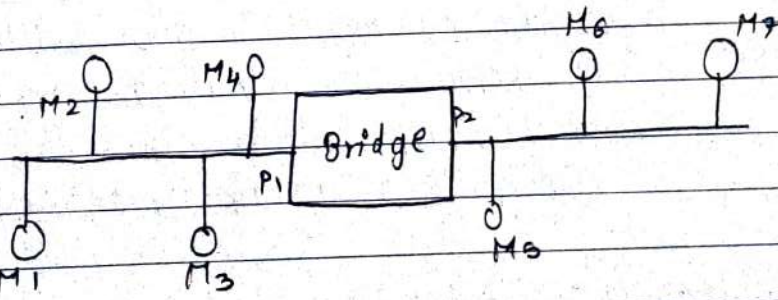
Types of Bridges: i) **Static Bridge:** Static bridges are that type of bridges which maintains a table that consists of two colounmns - MAC Address and Port Number

This table is manually handled by network admin.

ii) Dynamic - Bridge: Dynamic bridges are also known as self learning bridges or transparent bridges. In this type of the bridge the table is initially empty and is made using it's intelligence.

Advantages :- i) Bridges can be used as a network extension like they can connect two network topologies network.
ii) It has a seperate collision domain, which results in increased bandwidth.
iii). It can create a buffer when different MAC protocols are there for different segments.
iv) Highly reliable and maintainable. The network can be divided into multiple LAN segments.

Disadvantages : i) Expensive as compared to hubs and repeaters.
ii) Slow in speed.
iii) Poor performance as additional processing is required to view the MAC address of the device on the network.
iv) As the traffic recieved is in bulk or is broadcasted traffic, individual filtering of data is not possible.



D. Switch: A network device that connect multiple LAN networks irrespective of their types. It is a multiport bridge. It works at data link layer in the OSI model. Thus, it also known as Layer 2 device. Different devices can be connected to its port including a router. The switch is a network device that is used to segment the networks into different subnetworks called subnets or LAN segments.

Functions: i) It performs error checking before forwarding data.
ii) It transfers the data only to the device that has been addressed.
iii) Switches have a more significant number of ports.
iv) It operates in full duplex mode.

Types of Switches: i) Virtual Switches: Virtual switches are the switches that are inside virtual machine hosting environment.
ii) Routing switches: These are the switches that are used to connect LANs. They also have the work of performing functions in network layer of the OSI model.

Advantages: i) Prevents traffic loading over in a network by segmenting the network into smaller subnets.
ii) Increases the bandwith of the networks.
iii) Less frame collision as the switches creates the collision domain for each connection.

Disadvantages: i) It cannot stop traffic designed for a different LAN segment from travelling to

all other LAN segments
ii) switches are more expensive.

E. ROUTER: A router is a networking device that forwards data packets between computer networks. One or more packet-switched networks or subnetworks can be connected using a router. By sending data packets to their internal IP addresses, it manages traffic between different networks and permits several devices to share an Internet connection.

Functions: i) Forwarding: The router recieves the packets from its input ports, check its header, performs some basic functions like checksum, and then looks up to the routing table to find the appropriate output port to dump the packets onto, and forwards the packets onto that output port. It transfers the data only to that device that has been addressed.

ii) Routing: Routing is the process by which the router ascertains what is the best path for the packets to reach the destination, it maintains a routing table that is made using different algorithms by the router only.

iii) Filtering: Decides whether to forward/block the packet.

Types of Routers: i) Broadband Routers: These are one of the important kinds of routers. It is used to do different ways types of things. It is used to connect computers or it is also used to connect to the internet.

ii) - Wireless Routers: These routers are used to create a wireless signal in your office or home

Core Routers: Core routers distribute packets within the same networks. The main task is to carry heavy data transfers.

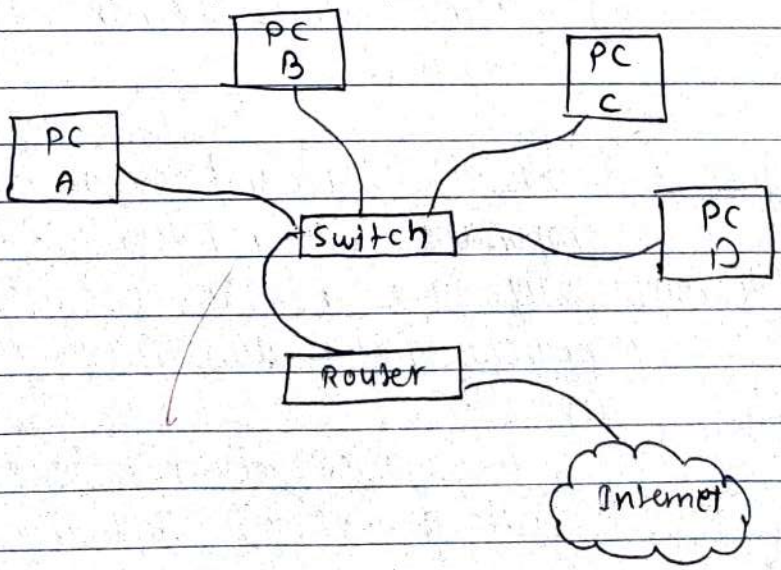
Advantages: i) Easier connection: sharing a single network connection among numerous machines is the router's main job.

ii) Security: Undoubtedly, installing a router is the first step in securing a network connection.

iii) Filtering Packets: Switching between packets and filtering packets are two more router services.

Disadvantages: i) slower: Routers are analyze multiple layers of information, from the physical layer to the network layer which slows down connection.

ii) High Cost: They are more expensive than some other tools for systems administration. This includes security, extension and the focal point. As a result, routers are typically not the greatest option for issues.



F. Gateway :- A gateway is a network node that forms a passage between two networks operating with different transmission protocols. The most common type of gateways, the network gateway operates at layer 3, i.e. the network layer of the OSIC Open systems Interconnection) model.

Functions :- i) Gateway is located at the boundary of a network and manages all data that inflows or outflows from that network.
ii) It forms a passage between two different networks operating with different transmission protocols.
iii) A gateway operators as a protocol convertor, providing compatibility between different protocols used in the two different networks.
iv) The feature that differentiates a gateway from other network devices is that it can operate at any layer of the OSI model.
v) It also stores information about the routing paths of the communicating networks.

Types of Gateways :- i) Unidirectional-Gateway :
They allow data to flow in only one direction. changes made in the source node are replicated in the destination node, but not vice-versa. They can be used as archiving tools.

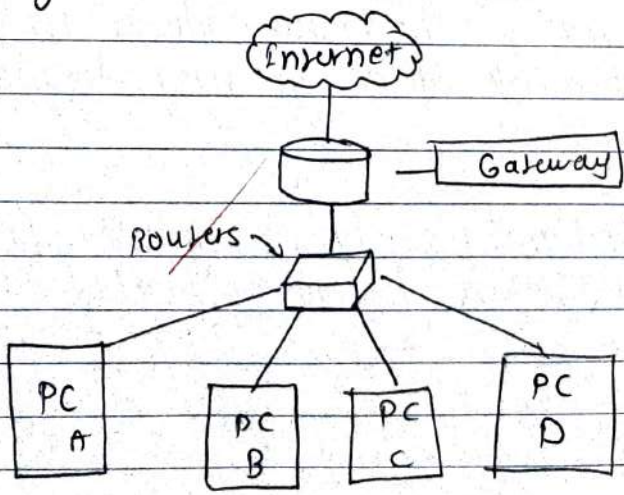
ii) Bidirectional-Gateways : They allow data to flow in both directions. They can be used as synchronization tools.

Advantages : i) Connectivity: As mentioned earlier, the main benefit of gateway is the connectivity it provide. A gateway can expand the network by connecting computers with different systems together. Through this, same kind of information will be able to be accessed by different computers.

ii) Security : Gateways are known to possess improved security since they allow user authentication. Forms of security such as user ID and password can be imposed on gateway so that all the unwanted access will be prevented.

iii) Filtering : Filtering process is another important capability of a gateway. Without them whatever the services that arrives at the gateway carries the risk of theft.

Disadvantages : i)- Gateway networks always causes time delay since information must be translated. There is no gateway can a instant transfer take place.
ii) If there are possibilities of failure causing occuring at the gateway, it can lead to communication loss.



6. MODEM :- Modem stands for Modulation Demodulation. A modem converts the digital data signals into analogue data signals. They can be installed within the computer in a development slot applicable to it.

Functions :- i) when a computer sends digital signals, it is converted into analog signals by modem. once it reaches the destination computer, the signals are converted back into its original form.
ii) Modem posses high transmission rates since telephones lines are used here. Usually these rates are measured in a unit called baud. Despite offering high speeds, it can cost more too.
iii) Modems are considerably expensive. A typical 46 modem costs more than a router. However the prices may vary depending upon the router's battery capacity. And also modems offers flexible internet plans in terms of pricing.

Types of Modems : i) Standard Modem : The standard modems use generic device drivers, and they can be internal and external forces ones.

ii) Window Modem :- A window modem is a private plugs and plays tool. It requires a particular device driver supported by the window operating framework to function properly/correctly.

Advantages : i) Signal Conversion :- When a computer sends digital signals it is converted into analog signals by a modem. Once it reaches the destination computer, the signals are converted back into its original form.

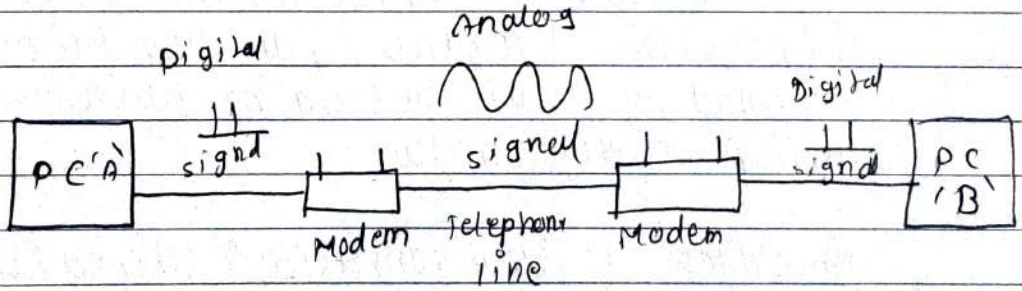
ii) Speed: Modem possesses high transmission rates since telephone lines are used here. Usually these rates are measured by in a unit called baud. Despite offering high speeds, it can cost more too.

iii) Cost:- Modems are considerebaly expencive. A typical 4G modem costs more than a router. However the prices may vary depending upon the router's capacity. And also modems offers flexible internet plans in terms of pricing.

Disadvantages: i). One major drawback of connecting a modem is that it can make your computer vulnerable to hackers and malwares.

ii). While external modems lack mobility, all the internal modems does support them. Users can use internal modems to benefit from wireless internet.

iii) - Modem especially of DSL type has a problem in availability. Services are not available in rural and remote locations. It solely depends upon the local phone company.



CIV-C12-32

Parth D.
C12
32

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

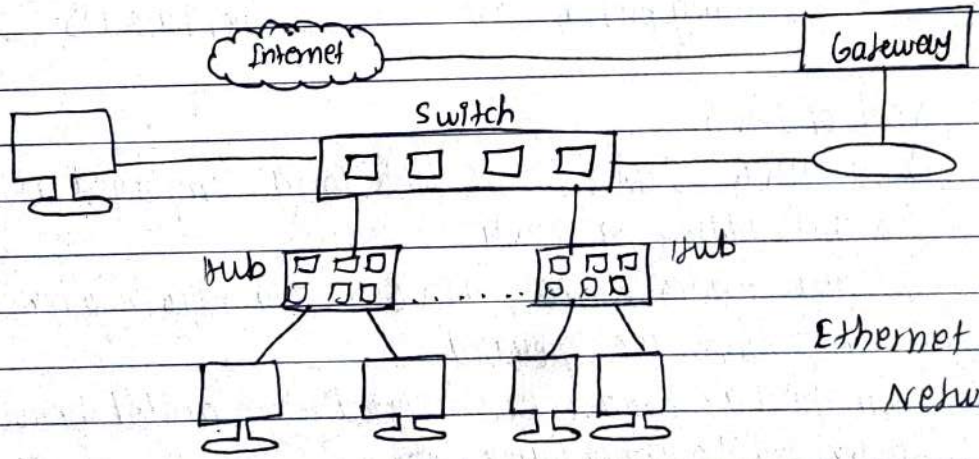
Assignment no: 2

1) Write a short note on:

i) Ethernet:

Ethernet is a type of communication protocol that connect computers on a network over a wired communication. It is a widely used LAN protocol, which is also known as Alto Aloha Network. It connects computers within the local area network and wide area network. Numeros devices like printers and laptops can be connected by LAN and WAN within buildings, homes and even small neighbours.

The wireless networks replaced ethernet in many areas; however Ethernet is still more common for wired networking. Wifi reduces the need for cabling as it allows users to connect smartphones or laptops to a network without the required cable.



Ethernet Network

Advantages of Ethernet:

i) It is costly, but still inexpensive than other options.
ii) Provides high security.
iii) Quality of data is maintained.

Disadvantages of Ethernet:

i) Distance covered is less.
ii) No acknowlegement from reciever side.
iii) Cannot determine which node is faulty.

ii) IPv6 :
IPv6 was developed by Internet Engineering Task Force (IETF) to deal with the problem of IPv4 exhaushion. IPv6 is a 128 bit address having an address space of 2^{128}, which is way bigger than IPv4. IPv6 use hexadecimal format seperated by colon(:)
Components of IPv6 address:
i) 8 - groups, each group represents 2 bytes (16 bits)
ii) Each Hex-digit is of 4 bits (1 nibble)
iii) Delimeter used - colon(:).

ABCD:EF8:1234:0:A1EA:A004:4001:53C8
Routing prefix | subnet id | Interface ID

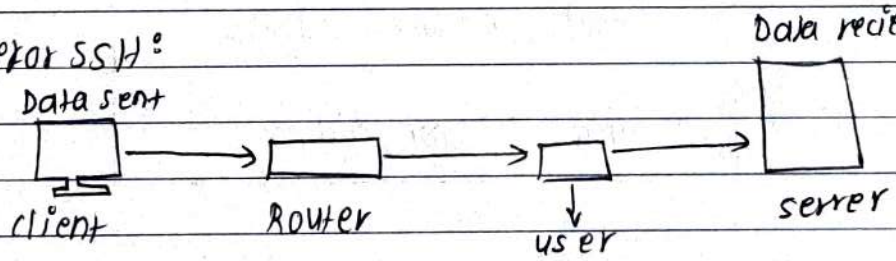
Need of IPv6 :
i) An IPv6 address is 128 bits long compared with the 32-bit address of IPv4.
ii) Better Header format; IPv6 uses a new header format in which options are seperated.
From the base header and inserted when needed between the base header and the upper layer data.
This simplifies and speeds up the routing process.
iii) IPv6 has new options to allow for functionalities.
iv) IPv6 allows extension of the protocol if required by new technologies.

Advantages of IPv6:
i) Real time data transmission
ii) Supports authentication.
iii) Performs encryption
iv) Faster processing at router side.

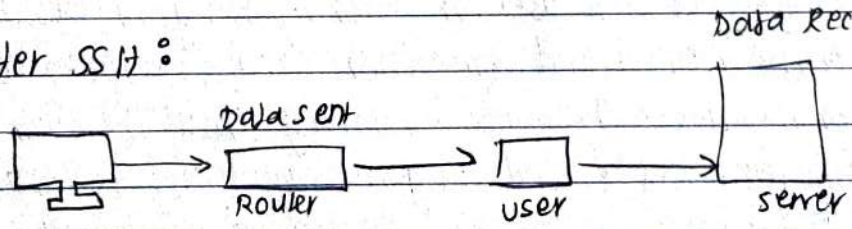
iii) SSH:
SSH stands for secure shell or secure socket shell. It is a cryptographic network protocol that allows two communicate and share the data over an insecure network such as internet. It is used to login to a remote center to execute commands and data transfer from one machine to another machine.

It provides a strong password encryption and password authentication communication with a public key over an insecure channel. It is used to replace unprotected remote login protocols such as telnet, login etc.

Befor SSH:



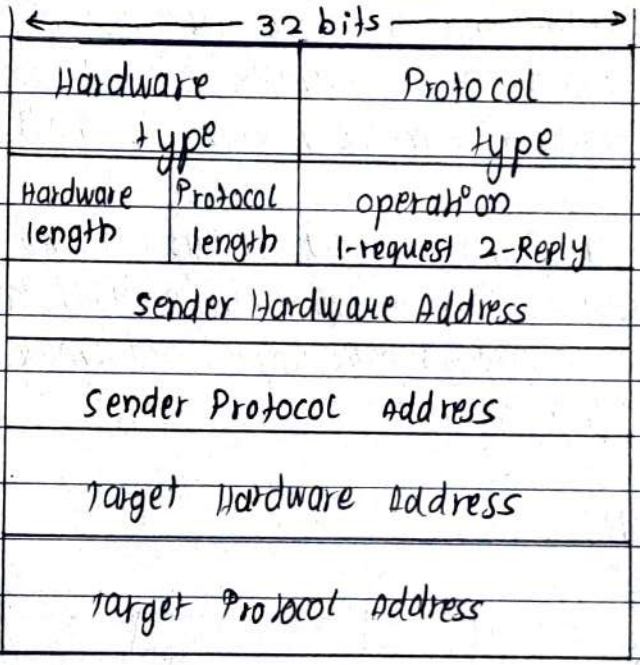
After SSH:



Advantages of SSH:
i) It prevents malicious activities like DNS spoofing, Data manipulat eavesdropping or sniffing of transmitted data. IP address routing.
ii) It uses strong authentication methods which provide more safety
iii) SSH enables port forwarding.

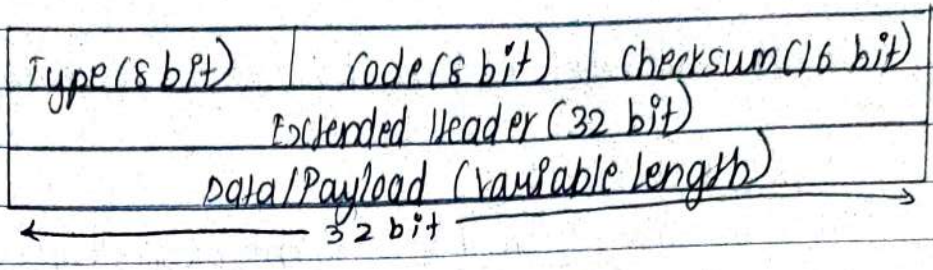
Q2] Explain the purpose of following protocols with their header:

i) ARP: The address Resolution Protocol (ARP) is a fundamental networking protocol that plays a pivotal role in local network communication by mapping known IP address to their corresponding physical MAC address. It enables devices on a network to discover and associate MAC addresses with IP addresses.



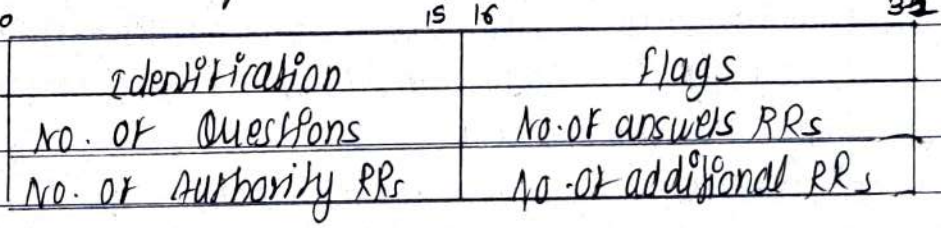
ii) ICMP: ICMP is used for error reporting if two devices connect over the internet and some error occurs, so router sends an ICMP error message to the source informing about the error. For example, whenever a device sends any message which is large enough for the reciever, in that case, the reciever will drop the message and replace ICMP message to the source.

Another important use of ICMP protocol is used to perform network diagnosis by making use of trace route and ping utility.



Type: Basic description of message
Code: Additional information
Checksum: Checksum of ICMP.
Extended header: Points out problem in IP message.
Data or Payload of variable length.

iii) DNS: The Domain Name System (DNS) serves the vital role of translating user friendly domain names into numerical IP addresses enabling seamless internet communication. It ensures user can access websites and services using easily memorable names, simplifying web browsing and resource location. It also provides data packets addressing and routing.



Identification: Used to match the response with the request.
Flags: 16 bit flags for indication of value. No. of Questions, Number of authority RRs, No. of answers RRs and No. of additional RRs are self-explanatory.

Q3] Discuss the persistent and non persistent protocols used in transport and application layers of TCP/IP protocol suite.

Persistent and Non Persistent protocol are two different approaches to handling connections in the transport and application layers of TCP/IP protocol suite. These protocols

govern how data is exchanged between two devices over a network.

Persistent Protocol:

i) Connection Oriented: Persistent protocols are connection oriented which means they establish a communication between the sender and reciever before data exchange begins. This connection remains open for multiple transactions.

ii) Examples: The most common example of a persistent protocol in TCP/IP suite is the transmission Control Protocol (TCP). TCP ensures reliable, ordered and error checked delivery of data. It maintains a continuous until explicitly closed by either the sender or reciever.

iii) Use Cases: Persistent protocols are suitable for applications that require reliable and ordered data transfer such as web browsing, email, file transfers and most client server instructions.

iv) Overhead: They typically have overhead due to the need for connection setup & maintainance, which involves three way handshake, Acknowledgement message and error recieving mechanisms.

Non Persistent Protocol:

i) Connectionless: Non persistent protocols are connection-less meaning they don't establish a continuos connection between sender and reciever. Instead a new communication is established for each data exchange and closed after that exchange is complete.

ii) Examples: A common example is User Datagram Protocol (UDP)

iii) Use Cases: Non persistent protocols are suitable for applications that prioritize speed and efficiency over reliability. Common used in real-time data sharing.

iv) Overhead: Lesser overhead as compared to present protocols because they skip communication setup.

Persisten Protocols: App: HTTP, SMTP, Transport Layer: TCP

Non-persistent protocols: App: DNS, SNMP, DHCP, Transport layer: UDP, SCTP