

UNIVERSITY OF VISVESVARAYA COLLEGE OF ENGINEERING

A State Autonomous University on IIT Model

K.R. Circle Bengaluru-560001

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

A

Practical Laboratory Manual on

Network Programming Lab

18CS1L01

Submitted for the I Semester

MASTER OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

SUBMITTED BY:

I Sem, M.Tech (CSE)

Batch of 2024–2026

UNDER THE GUIDANCE OF

Dr. Dilip Kumar S M

Professor

Dept of CSE, UVCE

2024–2025

Contents

1 Java Program to Design a TCP Iterative Client-Server Application to Reverse the Given Input Sequence	6
2 Write a Java program to design a TCP concurrent Client-Server application to reverse the given input sequence.	10
3 Write a Java program to design a TCP Client-Server application to transfer a file	15
4 Write a Java program to design UDP Client-Server application to transfer a file	19
5 Write a Java program to design a ARP/RARP protocol	23
6 Write a Java program to design a DHCP protocol	27
7 Write a Java program to Distance Vector Routing protocol	32
8 Write a Java program to Dijkstra's Shortest Path Routing protocol	36
9 Write a C++ program to connect two nodes on NS-3 (for practice only)	53
10 Write a C++ program to connect three nodes considering one as a central node on NS-3 (for practice only)	58
11 Write a C++ program to implement a star topology on NS-3	63
12 Write a C++ program to implement a bus topology on NS-3	71

- 13 Write a C++ program showing the connection of two nodes and four routers such that the extreme nodes act as client and server on NS-3** 77
- 14 Implement and study the performance of a typical GSM network on NS-3 (using MAC layer).** 84

Sl No	Student's Name	Register No.
1	ARFA THAREEN K	P25UV24T029001
2	ARPITHA M K	P25UV24T029002
3	BATTA NAGASAIKIRAN	P25UV24T029003
4	KAVYA R	P25UV24T029004
5	LAXMAN BASAVAYYA GONDA	P25UV24T029005
6	MEGHANA B M	P25UV24T029006
7	MOUNESH	P25UV24T029007
8	NAYAN C A	P25UV24T029008
9	NIZAMPURAM BHAVANI	P25UV24T029009
10	PANCHAMI M C	P25UV24T029010
11	RAHUL B	P25UV24T029011
12	RAKSHITH R	P25UV24T029012
13	RAMYA T	P25UV24T029013
14	SHARANPRAKASH K PATIL	P25UV24T029014
15	SRIKANTHA L	P25UV24T029015
16	V A NAVYASHREE	P25UV24T029016

GUIDE:

Dr. Dilip Kumar S M
Professor
Department of CSE
UVCE

CHAIRPERSON:

Dr. THRIVENI J
Professor
Department of CSE
UVCE

PART - A

1 Java Program to Design a TCP Iterative Client-Server Application to Reverse the Given Input Sequence

1. Server

- The server opens a `ServerSocket` on port 5000.
- It listens in an infinite loop for incoming client connections.
- When a client connects, the server accepts the connection and uses input/output streams (`BufferedReader` and `PrintWriter`) to receive data and send the reversed result.
- After handling a single request, the server closes the connection (hence “iterative”).
- The string reversal is performed using: `StringBuilder(input).reverse().toString()`.

2. Client

- The client connects to the server on `localhost` and port 5000.
- It takes user input from the console, sends it to the server, and then waits for a response.
- Once the reversed string is received, it is displayed to the user.
- Finally, the client closes the connection.

Use Cases

This type of architecture is useful for simple **command-response models** such as chat servers, string manipulation tools, or basic file transfer protocols. It forms the foundation for more advanced topics like **multithreaded servers**, **encryption**, and **REST-based APIs**.

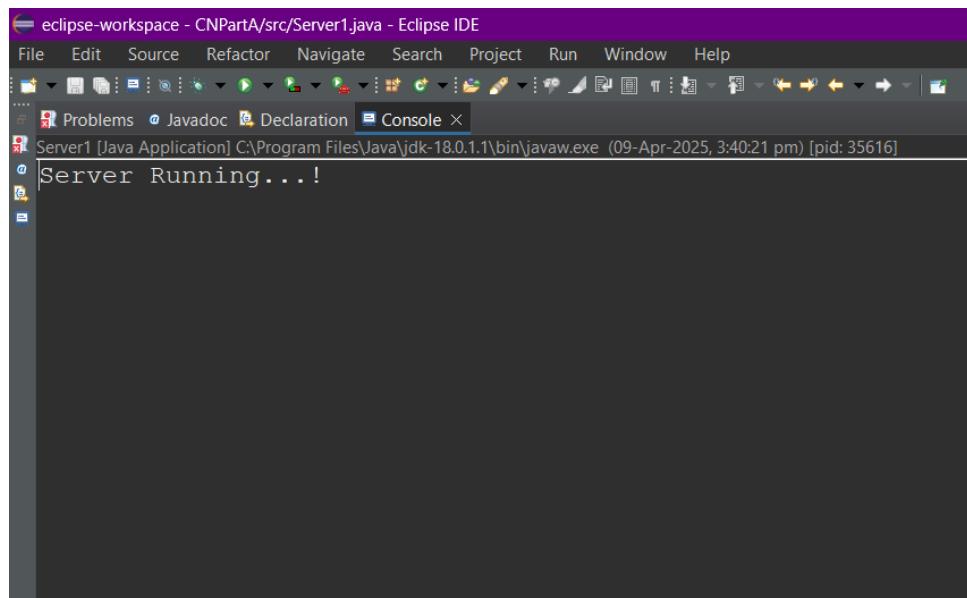
Client Code

```
1 import java.io.*;
2 import java.net.*;
3 class Client1
4 {
5     public static void main(String argv[]) throws Exception
6     {
7         String str="Hello networking";
8         String revStr;
9         Socket cliCon;
10        System.out.println("Connecting the Server...!");
11        cliCon = new Socket("localhost", 5555);
12
13        DataOutputStream ToServer = new DataOutputStream(cliCon.
14             getOutputStream());
15        BufferedReader FromServer = new BufferedReader(new
16             InputStreamReader(cliCon.getInputStream()));
17        ToServer.writeBytes(str + '\n');
18        revStr= FromServer.readLine();
19        System.out.println("Original String is: " + str);
20        System.out.println("Revers String is: " + revStr);
21        cliCon.close();
22    }
23 }
```

Server Code

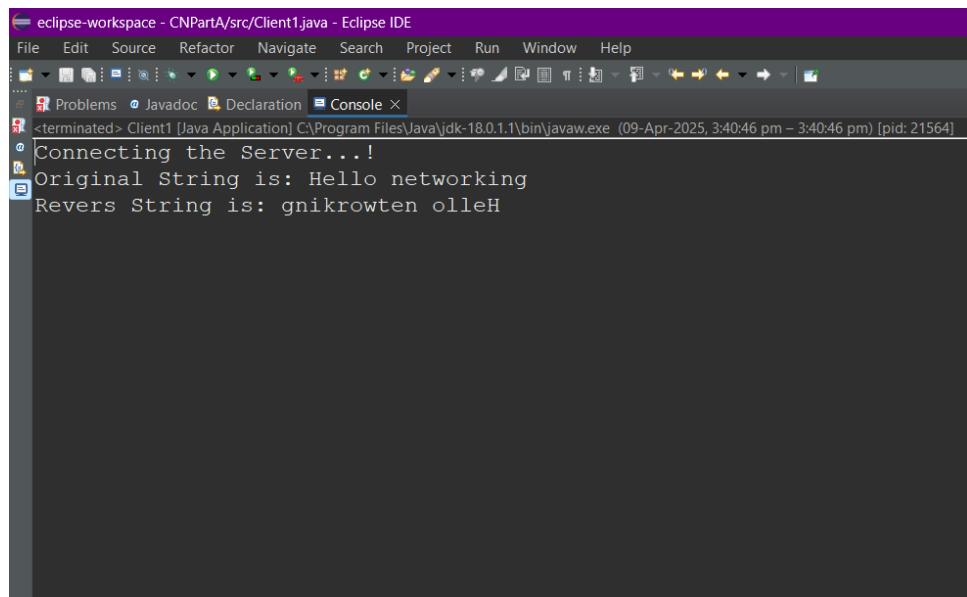
```
1 import java.io.*;
2 import java.net.*;
3 class Server1
4 {
5     public static void main(String argv[]) throws Exception
6     {
7         String str;
8         String revStr;
9         ServerSocket ss = new ServerSocket(5555);
10
11        System.out.println("Server Running...! ");
12        Socket serCon= ss.accept();
13        BufferedReader FromClient = new BufferedReader(new
14            InputStreamReader(serCon.getInputStream()));
15        DataOutputStream ToClient = new DataOutputStream(serCon.
16            getOutputStream());
17        str = FromClient.readLine();
18        revStr = new StringBuilder(str).reverse().toString() + '\n';
19        ToClient.writeBytes(revStr) ;
20        ss.close();
21        serCon.close();
22    }
23 }
```

OUTPUT



The screenshot shows the Eclipse IDE interface with a purple header bar. The title bar reads "eclipse-workspace - CNPartA/src/Server1.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. Below the menu is a toolbar with various icons. The central workspace contains a "Console" tab which is selected. The console output window shows the message "Server Running...!".

Figure 1: TCP Server



The screenshot shows the Eclipse IDE interface with a purple header bar. The title bar reads "eclipse-workspace - CNPartA/src/Client1.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. Below the menu is a toolbar with various icons. The central workspace contains a "Console" tab which is selected. The console output window shows the messages "<terminated> Client1 [Java Application] C:\Program Files\Java\jdk-18.0.1\bin\javaw.exe (09-Apr-2025, 3:40:46 pm – 3:40:46 pm) [pid: 21564]", "Connecting the Server....!", "Original String is: Hello networking", and "Revers String is: gnikrowten olleH".

Figure 2: TCP Client

2 Write a Java program to design a TCP concurrent Client-Server application to reverse the given input sequence.

This TCP-based application demonstrates a simple client-server model where the server listens on a specific port for incoming client connections. Upon connection, the server spawns a new thread to handle that specific client, enabling multiple clients to connect and communicate concurrently. The client sends a string to the server, which the server then reverses and sends back to the client.

Key Concepts

- **TCP (Transmission Control Protocol):** Ensures reliable, ordered, and error-checked delivery of a stream of data between applications.
- **Concurrency:** Achieved using Java Threads. Each client is served by a separate thread on the server side.
- **Socket Programming:** Utilizes Java's Socket and ServerSocket classes to establish communication.

How It Works

Server Side

1. A ServerSocket is created and bound to a specific port (e.g., 5000).
2. The server continuously listens for incoming client connections.
3. For each new connection, a new ClientHandler thread is instantiated and started.
4. The thread:
 - Reads input from the client.

- Reverses the received string.
- Sends the reversed string back to the client.

Client Side

1. A Socket is created to connect to the server.
2. The client sends a string to the server via the socket's output stream.
3. The client reads the reversed string from the server's response and displays it.

Client Code

```

1 // Client.java
2 import java.net.*;
3 import java.io.*;
4 public class Client2
5 {
6 @SuppressWarnings({ "resource", "deprecation" })
7 public static void main(String[] args) throws Exception
8 {
9 while(true)
10 {
11 Socket s=new Socket("localhost",5001);
12 System.out.println("Enter String to reverse:");
13 DataInputStream in=new DataInputStream(System.in);
14 String str=in.readLine();
15 DataOutputStream dout=new DataOutputStream(s.getOutputStream());
16 dout.writeUTF(str);
17 DataInputStream din=new DataInputStream(s.getInputStream());
18 String rev=din.readUTF();
19 System.out.println("Reversed String:\t"+rev);
20 }
21 }
22 }
```

Server Code

```

1 // Server.java
2 import java.net.*;
3 import java.io.*;
4 public class Server2
5 {
6 @SuppressWarnings("resource")
7 public static void main(String[] args) throws Exception
8 {
9 int count=1;
10 System.out.println("Server Running.....!");
11 ServerSocket ss=new ServerSocket(5001);
12 while(true)
13 {
14 new RevThread(ss.accept(),count).start();
15 System.out.println(count+" Client connected.");
16 count++;
17 }
18 }
19 }
20 class RevThread extends Thread
21 {
22 Socket s=null;
23 int n;
24 public RevThread(Socket socket,int count)
25 {
26 s=socket;
27 n=count;
28 }
29 public void run()
30 {
31 try
32 {
33 while(true)
34 {
35 System.out.println("Receiving from client "+n);
36 DataInputStream din=new DataInputStream(s.getInputStream());
37 String str=din.readUTF();
38 System.out.println("Processing data of Client "+n);
39 String revStr=new StringBuilder(str).reverse().toString();
40 System.out.println("Sending to client "+n);
41 DataOutputStream dout=new DataOutputStream(s.getOutputStream());
42 dout.writeUTF(revStr);
43 }
44 }

```

```
45 }
46 catch(IOException e)
47 {
48
49 System.out.println(e);
50 }
51 }
52 }
```

OUTPUT

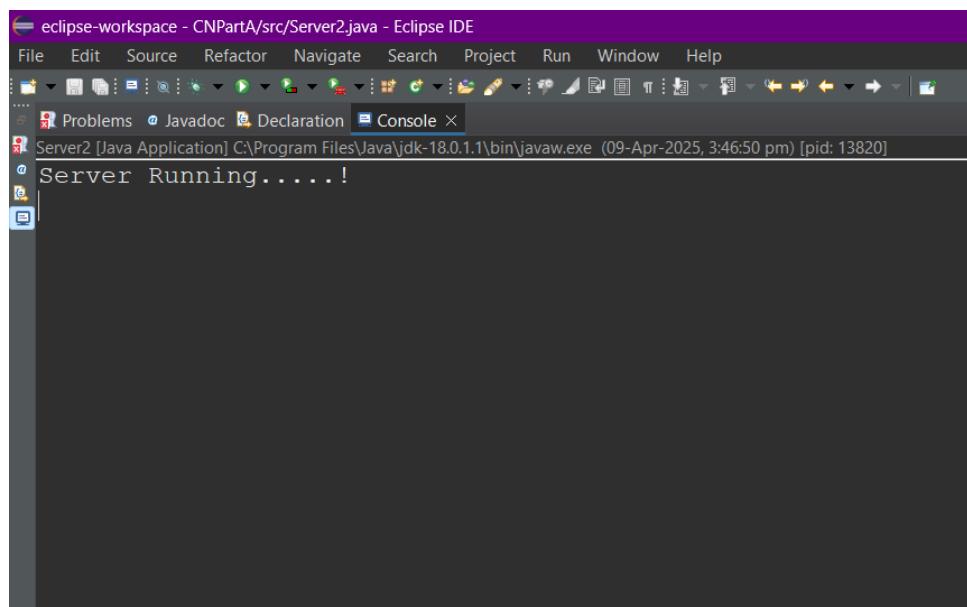
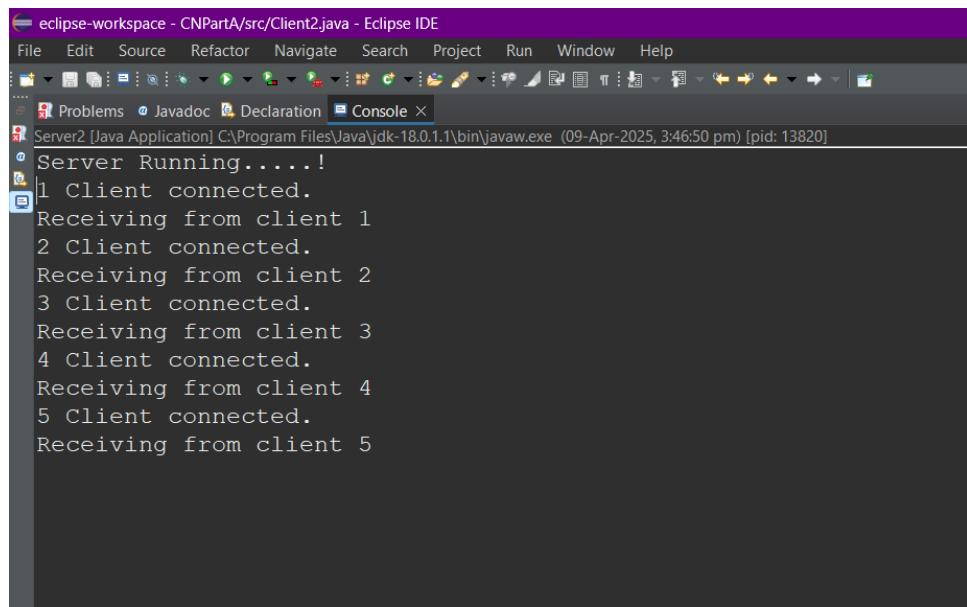


Figure 3: TCP Server

OUTPUT



The screenshot shows the Eclipse IDE interface with a purple header bar. The title bar reads "eclipse-workspace - CNPartA/src/Client2.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. Below the menu is a toolbar with various icons. A central workspace area contains a "Console" tab which is selected. The console output window displays the following text:

```
Server2 [Java Application] C:\Program Files\Java\jdk-18.0.1\bin\javaw.exe (09-Apr-2025, 3:46:50 pm) [pid: 13820]
@ Server Running.....!
@ 1 Client connected.
Receiving from client 1
2 Client connected.
Receiving from client 2
3 Client connected.
Receiving from client 3
4 Client connected.
Receiving from client 4
5 Client connected.
Receiving from client 5
```

Figure 4: TCP Client

3 Write a Java program to design a TCP Client-Server application to transfer a file

This Java application demonstrates how to send a file from a client to a server using TCP sockets. The server listens on a specified port and receives files sent by the client. On the client side, a file is opened, read into a byte stream, and transmitted over the network. TCP ensures reliable and ordered delivery of data, making it ideal for file transfer operations.

Key Concepts

- **TCP (Transmission Control Protocol):** Guarantees reliable, sequential, and error-checked transmission of data between applications.
- **Socket Programming:** Employs Java's `Socket` and `ServerSocket` classes for establishing communication.
- **File I/O Streams:** Uses byte streams to read and write files across the network.

How It Works

Server Side

1. A `ServerSocket` is created and listens on a specific port (e.g., 5000).
2. When a client connects, the server accepts the incoming socket.
3. It reads the incoming byte stream from the client.
4. The received data is saved to a file (e.g., `received_file.txt`).

Client Side

1. The client connects to the server using a Socket.
2. A file (e.g., sample.txt) is opened and read into a byte array.
3. The byte stream is sent over the socket to the server.
4. Once the file is completely sent, the connection is closed.

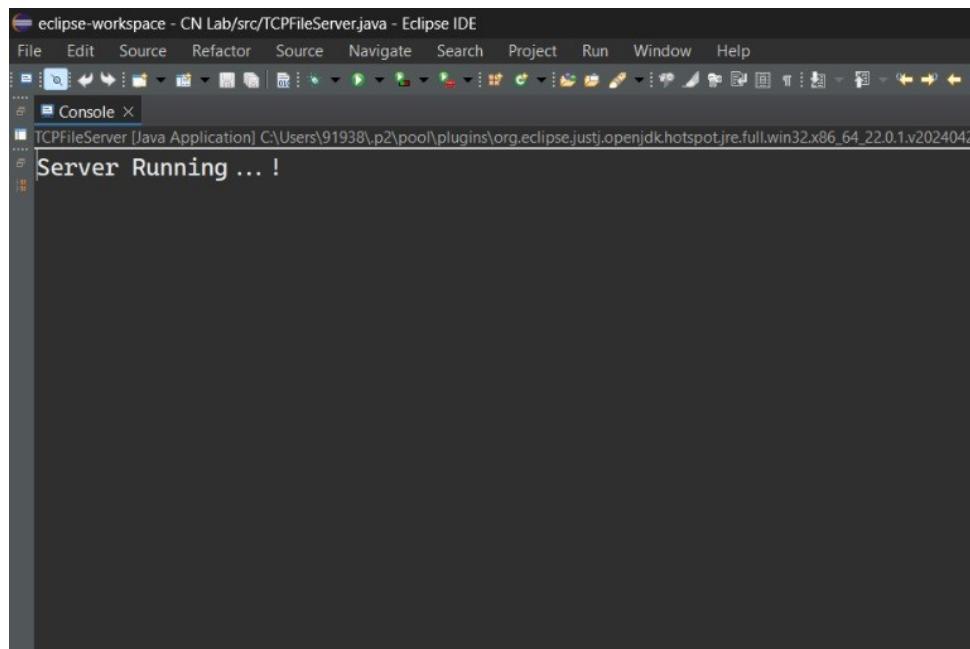
Client Code

```
1 // Client.java
2 import java.io.*;
3 import java.net.*;
4 public class TCPFileClient
5 {
6
7     public static void main(String[] args) throws Exception
8     {
9         Socket cliCon=new Socket("localhost",4002);
10        System.out.println("Connected to Server ...!");
11        FileOutputStream fout=new FileOutputStream("C:/Users/moune/Desktop/
12             Run.txt");
13        DataInputStream din=new DataInputStream(cliCon.getInputStream());
14        int r;
15
16        while((r=din.read())!=-1)
17        {
18            fout.write((char)r);
19        }
20
21        fout.close();
22        cliCon.close();
23    }
24 }
```

Server Code

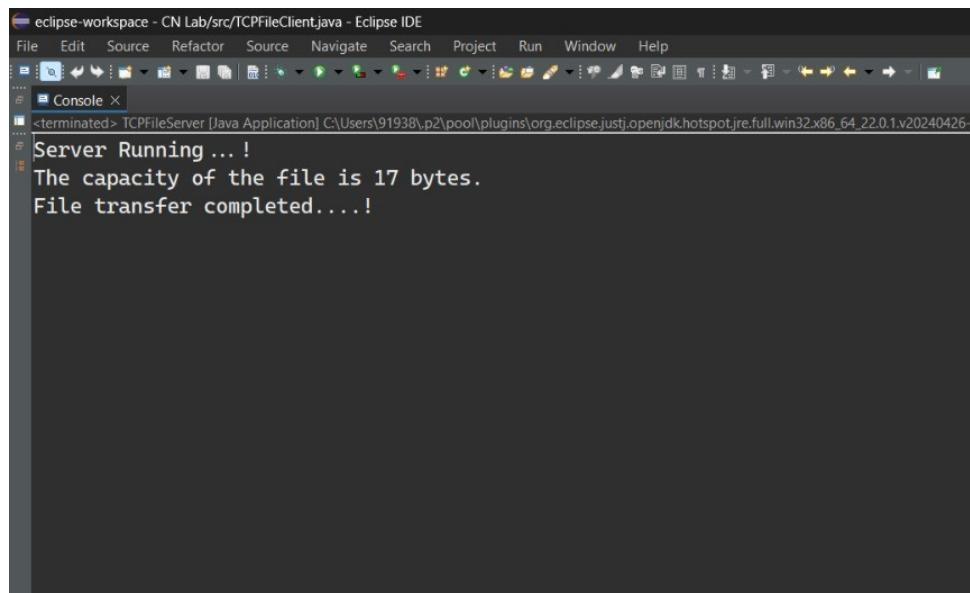
```
1 // Server.java
2 import java.io.*;
3 import java.net.*;
4 public class TCPFileServer
5 {
6     public static void main(String[] args) throws Exception
7     {
8         ServerSocket ss=new ServerSocket(4002);
9         System.out.println("Server Running...!");
10        Socket serCon=ss.accept();
11        FileInputStream fin=new FileInputStream("C:/Users/moune/Desktop/
12             transfer.txt");
13        DataOutputStream dout=new DataOutputStream(serCon.getOutputStream())
14            ;
15        int r;
16        int count=0;
17        while((r=fin.read())!=-1)
18        {
19            dout.write(r);
20            count++;
21        }
22        System.out.println("The capacity of the file is " +count+" bytes.");
23        System.out.println("File transfer completed....!");
24        fin.close();
25        ss.close();
26        serCon.close();
27    }
}
```

OUTPUT



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - CN Lab/src/TCPFileServer.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Source, Navigate, Search, Project, Run, Window, and Help. The toolbar below the menu bar has various icons for file operations like Open, Save, Cut, Copy, Paste, Find, and Select. A "Console" tab is selected in the top-left corner. The main console window displays the message "Server Running ... !".

Figure 5: TCP Server



The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - CN Lab/src/TCPFileClient.java - Eclipse IDE". The menu bar and toolbar are identical to Figure 5. The "Console" tab is selected. The main console window displays the server's response: "Server Running ... !", "The capacity of the file is 17 bytes.", and "File transfer completed....!".

Figure 6: TCP Client

4 Write a Java program to design UDP Client-Server application to transfer a file

What is UDP?

UDP (User Datagram Protocol) is a connectionless transport protocol that transmits data as individual packets called **datagrams**, without the need to establish or maintain a connection between the sender and receiver.

It offers faster data transfer compared to TCP but does not guarantee:

- Delivery of packets
- Order of arrival
- Avoidance of duplication

Therefore, while UDP is suitable for scenarios where speed is more critical than reliability, manual mechanisms are often needed to handle lost or disordered packets.

How This Program Works

Client Side

1. Opens and reads a file (e.g., `sample.txt`) in byte chunks.
2. Each chunk is sent as a UDP packet to the server using a `DatagramSocket`.
3. After sending all data, a special packet with content “EOF” (End Of File) is transmitted to inform the server that the file transmission is complete.

Server Side

1. A `DatagramSocket` is initialized to listen on a specific port (e.g., 5000).

2. The server receives incoming UDP packets.
3. The received data is written into a new file (e.g., `received_file.txt`).
4. When the “EOF” packet is detected, the server stops receiving data.

Key Concepts Used

- **DatagramSocket:** Used for sending and receiving UDP packets.
- **DatagramPacket:** Encapsulates data to be sent or received over the network.
- **FileInputStream / FileOutputStream:** Handles reading from and writing to files as byte streams.

UDP Client Code

```

1 // UDPClient.java
2 import java.net.*;
3 import java.io.*;
4 public class UDPClient4
5 {
6     @SuppressWarnings("resource")
7     public static void main(String args[]) throws Exception
8     {
9         byte b []=new byte [1024];
10        System.out.println("Connecting UDP Server....!");
11        FileInputStream fin=new FileInputStream("C:\\\\Users\\\\moune\\\\Documents\\\\"
12                                         "UDP\\\\UDPProgram.txt");
13        DatagramSocket dsoc=new DatagramSocket();
14        int i=0;
15        while(fin.available()!=0)
16        {
17            b[i]=(byte)fin.read();
18            i++;
19        }
20        fin.close();
21        dsoc.send(new DatagramPacket(b,i,InetAddress.getLocalHost(),1000));
22    }
}

```

UDP Server Code

```

1 // UDPServer.java
2 import java.net.*;
3 import java.io.*;
4 public class UDPServer4
5 {
6     @SuppressWarnings("resource")
7     public static void main(String args[]) throws IOException
8     {
9         byte b []=new byte [2048];
10        System.out.println("UDP Server Running....!");
11        DatagramSocket dsoc=new DatagramSocket(1000);
12        FileOutputStream fout = new FileOutputStream("C:\\\\Users\\\\moune\\\\"
13                                         "Documents\\\\UDP\\\\UDPProgram.txt");
14        DatagramPacket dp=new DatagramPacket(b,b.length);
15        dsoc.receive(dp);
16        String str=new String(dp.getData());
}

```

```
16 byte[] strToBytes = str.getBytes();
17 fout.write(strToBytes);
18 System.out.println("File transfer completed....!");
19 fout.close();
20
21 }
22 }
```

OUTPUT

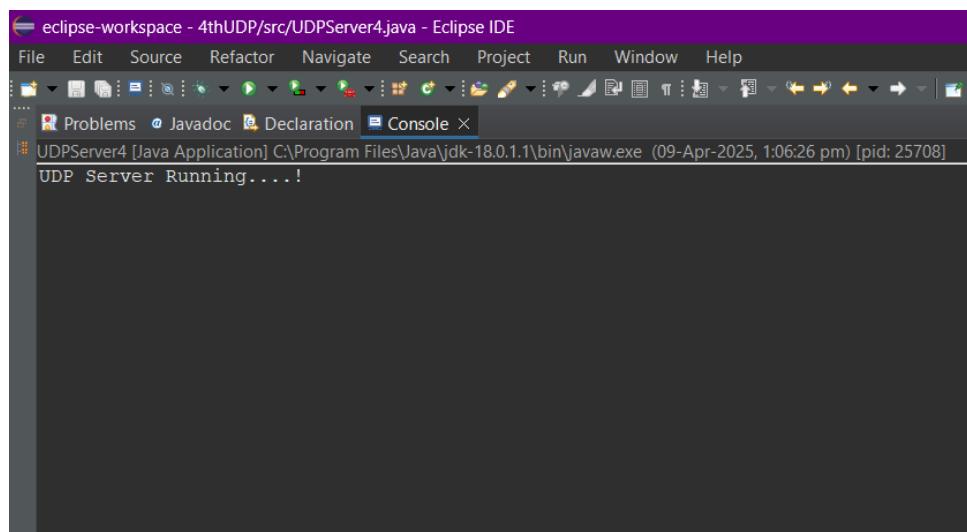


Figure 7: UDP Server

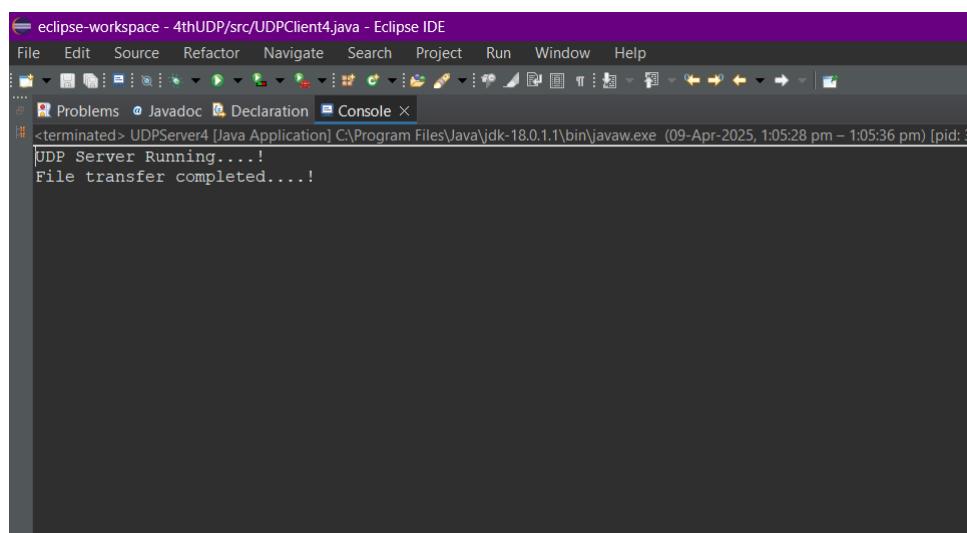


Figure 8: UDP Client

5 Write a Java program to design a ARP/RARP protocol

What is ARP?

Address Resolution Protocol (ARP) is a protocol used to map a known IP address to its corresponding MAC (Media Access Control) address on a local network.

Example

Suppose you want to send data to a device with the IP address 192.168.0.2. Your computer will use ARP to ask:

“Who has IP 192.168.0.2? Tell me your MAC address.”

If the device with that IP address is present on the network, it will respond with its MAC address. This MAC address is then used to deliver the packet at the data link layer.

What is RARP?

Reverse Address Resolution Protocol (RARP) is used to determine the IP address of a device when only its MAC address is known. This protocol is mainly useful for devices that do not have the capability to store IP addresses, such as diskless workstations.

Example

A diskless computer may only know its MAC address. It will broadcast a message:

“I am MAC AA:BB:CC:DD:EE:01. What is my IP address?”

A RARP server on the network will respond with the appropriate IP address for that MAC address.

ARP Client Code

```
1 // ARPClient.java
2 import java.net.*;
3
4 public class ARPServer {
5     @SuppressWarnings("resource")
6     public static void main(String args[]) {
7         try {
8             DatagramSocket sdsoc = new DatagramSocket(1309);
9             while (true) {
10                 System.out.println("ARP Server Running...!");
11                 byte[] sendByte = new byte[1024];
12                 byte[] receiveByte = new byte[1024];
13                 DatagramPacket recePak = new DatagramPacket(receiveByte,
14                     receiveByte.length);
15
16                 sdsoc.receive(recePak);
17                 String str = new String(recePak.getData());
18                 String s = str.trim();
19
20                 InetAddress addr = recePak.getAddress();
21                 int port = recePak.getPort();
22
23                 // Expanded ARP table
24                 String ip[] = {
25                     "10.0.3.186",
26                     "10.0.3.187",
27                     "10.0.3.188"
28                 };
29                 String mac[] = {
30                     "D4:3D:7E:12:A3:D9",
31                     "00:1A:2B:3C:4D:5E",
32                     "11:22:33:44:55:66"
33                 };
34
35                 boolean found = false;
36
37                 for (int i = 0; i < mac.length; i++) {
38                     if (s.equalsIgnoreCase(mac[i])) {
39                         sendByte = ip[i].getBytes();
40                         DatagramPacket sendPak = new DatagramPacket(
41                             sendByte, sendByte.length, addr, port);
42                         sdsoc.send(sendPak);
43                         found = true;
44                         break;
45                     }
46                 }
47             }
48         } catch (IOException e) {
49             e.printStackTrace();
50         }
51     }
52 }
```

```

43         }
44     }
45
46     if (!found) {
47         String msg = "MAC Address not found in ARP table.";
48         sendByte = msg.getBytes();
49         DatagramPacket sendPak = new DatagramPacket(
50             sendByte, sendByte.length, addr, port);
51         sdsoc.send(sendPak);
52     }
53 } catch (Exception e) {
54     System.out.println(e);
55 }
56 }
57 }
```

ARP Server Code

```

1 // ARPServer.java
2 import java.io.*;
3 import java.net.*;
4 public class ARPCClient
5 {
6     public static void main(String args[])
7     {
8         try
9         {
10             DatagramSocket cdSoc = new DatagramSocket();
11             InetAddress addr = InetAddress.getByName("localhost");
12             byte[] sendByte = new byte[1204];
13             byte[] receiveByte = new byte[1024];
14
15             BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
16             System.out.println("Enter the Physical Address: ");
17             String str = in.readLine();
18             sendByte = str.getBytes();
19
20             DatagramPacket sendPak = new DatagramPacket(sendByte, sendByte.length,
21                 addr, 1309);
22             cdSoc.send(sendPak);
23             DatagramPacket recePak = new DatagramPacket(receiveByte, receiveByte.
24                 length);
25             cdSoc.receive(recePak );
```

```
24 String s = new String(recePak.getData());
25
26 System.out.println("The Logical Address is :" + s.trim());
27 cdSoc.close();
28 }
29 catch(Exception e)
30 {
31 System.out.println(e);
32 }
33 }
34 }
```

OUTPUT

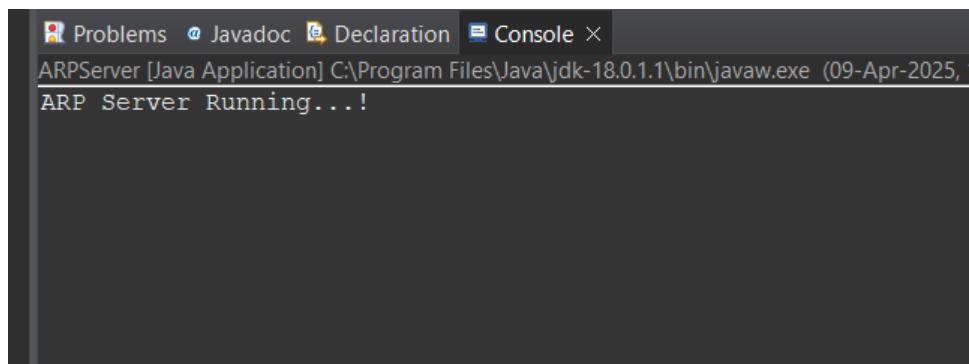


Figure 9: ARP Server

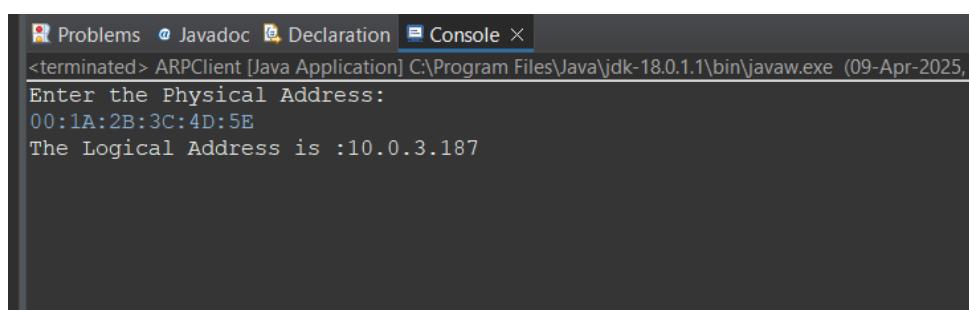


Figure 10: ARP Client

6 Write a Java program to design a DHCP protocol

Dynamic Host Configuration Protocol

The Dynamic Host Configuration Protocol (DHCP) is a network management protocol used to dynamically assign IP addresses to devices on a network. In this project, we simulate the DHCP protocol in Java using UDP sockets. While a real DHCP implementation uses raw Ethernet frames and operates at a lower level in the OSI model, this simulation provides a high-level understanding using Java's networking libraries.

Basic Idea Behind DHCP

The DHCP process typically involves the following steps:

- **DHCP Discover:** The client broadcasts a request for an IP address.
- **DHCP Offer:** The server responds with an available IP address.
- **DHCP Request:** The client accepts the offered IP address.
- **DHCP Acknowledgement:** The server confirms the assignment of the IP address.

Simulation in Java

In our Java simulation, we use UDP sockets to mimic the DHCP communication. The simulation includes:

- A **DHCP Server** that listens for incoming requests and responds with available IP addresses.
- A **DHCP Client** that sends a discover message and processes the server's response.

Concepts Used

- **DatagramSocket** and **DatagramPacket** for sending and receiving UDP packets.
- **Threads** to allow the server to continuously listen for requests while clients can operate independently.

Implementation Outline

Here is a simplified outline of how the simulation is structured:

DHCP Server

```

1 DatagramSocket serverSocket = new DatagramSocket(9876);
2 byte[] receiveData = new byte[1024];
3
4 while (true) {
5     DatagramPacket receivePacket = new DatagramPacket(receiveData,
6             receiveData.length);
7     serverSocket.receive(receivePacket);
8     // Parse message, send offer or ack
9 }
```

DHCP Client

```

1 DatagramSocket clientSocket = new DatagramSocket();
2 InetAddress IPAddress = InetAddress.getByName("localhost");
3
4 String discoverMessage = "DHCPDISCOVER";
5 byte[] sendData = discoverMessage.getBytes();
6
7 DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.
8         length, IPAddress, 9876);
9 clientSocket.send(sendPacket);
```

DHCP Client Code

```
1 // DHCPClient.java
2 import java.net.*;
3 public class DhcpClient
4 {
5     public static void main(String args[])
6     {
7         try
8         {
9             DatagramSocket cdsoc = new DatagramSocket();
10            byte[] sendByte = new byte[1204];
11            byte[] receiveByte = new byte[1024];
12            System.out.println("Sending the mac address to server : ");
13            InetAddress address = InetAddress.getLocalHost();
14            NetworkInterface ni = NetworkInterface.getByInetAddress(address);
15            sendByte = ni.getHardwareAddress();
16
17            for (int i = 0; i < sendByte.length; i++)
18            {
19                System.out.format("%02X%s", sendByte[i], (i < sendByte.length - 1) ? " "
20                                : " ");
21            }
22            DatagramPacket sender = new DatagramPacket(sendByte, sendByte.length,
23                  address, 67);
24            cdsoc.send(sender);
25            DatagramPacket recePak = new DatagramPacket(receiveByte, receiveByte.
26                  length);
27            cdsoc.receive(recePak);
28            String s = new String(recePak.getData());
29            System.out.println("\nThe Logical Address is :" + s.trim());
30            cdsoc.close();
31        }
32        catch(Exception e)
33        {
34            System.out.println(e);
35        }
36    }
37}
```

DHCP Server Code

```

1 // DHCPServer.java
2 import java.net.*;
3 import java.util.Random;
4 public class DhcpServer
5 {
6
7     @SuppressWarnings("resource")
8     public static void main(String args[])
9     {
10
11         try
12         {
13             System.out.println("DHCP Server Running...! ");
14             DatagramSocket sdsoc = new DatagramSocket(67);
15             while(true)
16             {
17                 byte[] sendByte = new byte[1204];
18                 byte[] receiveByte = new byte[1204];
19                 DatagramPacket recePak = new DatagramPacket(receiveByte,receiveByte.
20                     length);
21
22                 sdsoc.receive(recePak);
23                 byte[] mac=recePak.getData();
24
25                 System.out.print("Server allocating IP Address to:");
26                 for (int i = 0; i <=5; i++)
27                 {
28                     System.out.format("%02X%s", mac[i], (i <5) ? "-" : "");
29                 }
30
31                 InetAddress addr = recePak.getAddress();
32                 int port = recePak.getPort();
33                 String ip[] = {"10.0.3.186","10.0.3.187","10.0.3.188","10.0.3.189","10.0.3.189"};
34
35                 Random rand = new Random();
36                 int i = rand.nextInt(4);
37                 ip[i]=ip[i]+" lease time->24hrs";
38                 System.out.println("->"+ ip[i]);
39
40                 sendByte = ip[i].getBytes();
41                 DatagramPacket sendPak = new DatagramPacket(sendByte,sendByte.length,
42                     addr,port);
43                 sdsoc.send(sendPak);
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
317
318
319
319
320
321
322
323
324
325
326
327
327
328
329
329
330
331
332
333
334
335
336
337
337
338
339
339
340
341
342
343
344
345
346
347
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
367
368
369
369
370
371
372
373
374
375
376
377
377
378
379
379
380
381
382
383
384
385
386
387
387
388
389
389
390
391
392
393
394
395
396
397
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
411
412
413
413
414
415
415
416
416
417
417
418
418
419
419
420
420
421
421
422
422
423
423
424
424
425
425
426
426
427
427
428
428
429
429
430
430
431
431
432
432
433
433
434
434
435
435
436
436
437
437
438
438
439
439
440
440
441
441
442
442
443
443
444
444
445
445
446
446
447
447
448
448
449
449
450
450
451
451
452
452
453
453
454
454
455
455
456
456
457
457
458
458
459
459
460
460
461
461
462
462
463
463
464
464
465
465
466
466
467
467
468
468
469
469
470
470
471
471
472
472
473
473
474
474
475
475
476
476
477
477
478
478
479
479
480
480
481
481
482
482
483
483
484
484
485
485
486
486
487
487
488
488
489
489
490
490
491
491
492
492
493
493
494
494
495
495
496
496
497
497
498
498
499
499
500
500
501
501
502
502
503
503
504
504
505
505
506
506
507
507
508
508
509
509
510
510
511
511
512
512
513
513
514
514
515
515
516
516
517
517
518
518
519
519
520
520
521
521
522
522
523
523
524
524
525
525
526
526
527
527
528
528
529
529
530
530
531
531
532
532
533
533
534
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
143
```

```
42    }
43 }
44 catch(Exception e)
45 {
46 System.out.println(e);
47 }
48 }
49 }
```

OUTPUT

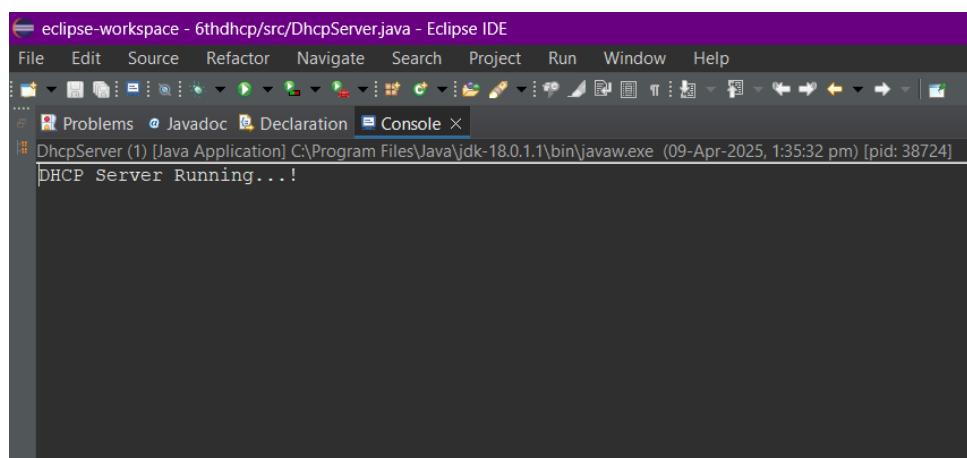


Figure 11: DHCP Server

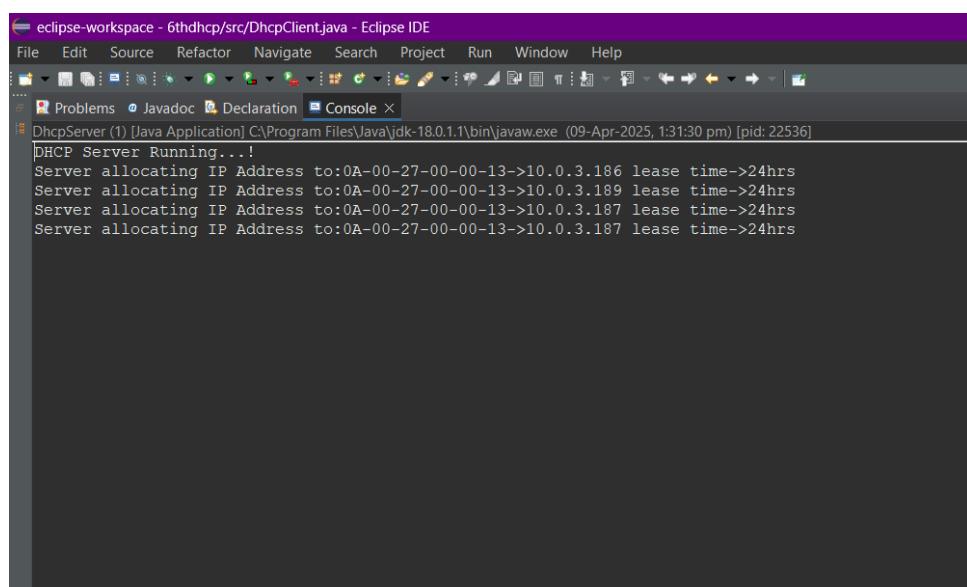


Figure 12: DHCP Client

7 Write a Java program to Distance Vector Routing protocol

Distance Vector Routing Protocol

The Distance Vector Routing Protocol is a dynamic routing algorithm used in computer networks. Each router maintains a routing table that stores the best-known distance (cost) to reach every other router and the next hop to take. It is a *distributed protocol*, meaning that routers update their tables independently based on periodic information exchanges with their immediate neighbors.

How It Works

- Each router periodically shares its routing table with directly connected neighbors.
- Upon receiving a neighbor's routing table, a router updates its own table using the **Bellman-Ford algorithm**.
- If the cost to reach a destination via a neighbor is lower than the current cost, the routing table is updated.
- Over time, all routers converge to the shortest paths to every destination in the network.

Key Concepts

Distance Vector: Each router maintains a vector (table) of distances to all possible destinations and the corresponding next hops.

Bellman-Ford Algorithm: A dynamic programming algorithm used to compute the shortest paths from one node to all others.

Cost Matrix: Represents direct link costs between routers; a very large number (e.g., 9999) indicates no direct link.

Next Hop: The neighbor to which packets are forwarded to reach a particular destination efficiently.

Java Program Highlights

The Java simulation models this protocol for a fixed number of routers.

Input

- Number of routers in the network.
- Cost matrix indicating the direct link cost (e.g., hop count or latency) between routers.

Routing Table Update

- Each router uses nested loops to simulate receiving distance vectors from all other routers.
- The Bellman-Ford algorithm is used to update the values in the distance and next hop tables.
- Specifically, the program updates `distance[i][j]` and `nextHop[i][j]` for all routers i and j .

Output

- The final routing table for each router is printed.
- Each routing table displays:
 - Destination router
 - Next hop
 - Distance (cost)

```

1 // DistanceVectorRouting.java
2 import java.util.Scanner;
3 public class DistanceVectorRouting
4 {
5     private int distances[];
6     private int ver;
7     public static final int MAX_VALUE = 999;
8     public DistanceVectorRouting(int numofver)
9     {
10         this.ver = numofver;
11         distances = new int[numofver + 1];
12     }
13     public void BellmanFordEvaluation(int source, int matrix[][])
14     {
15         for (int node = 1; node <= ver; node++)
16         {
17             distances[node] = MAX_VALUE;
18         }
19         distances[source] = 0;
20         for (int node = 1; node <= ver - 1; node++)
21         {
22             for (int src = 1; src <= ver; src++)
23             {
24                 for (int dest = 1; dest <= ver; dest++)
25                 {
26                     if (matrix[src][dest] != MAX_VALUE)
27                     {
28                         if (distances[dest] > distances[src]
29                             + matrix[src][dest])
30                             distances[dest] = distances[src]
31                             + matrix[src][dest];
32                     }
33                 }
34             }
35         }
36         for (int vertex = 1; vertex <= ver; vertex++)
37         {
38             System.out.println("distance from source " + source + " to "
39 + vertex + " is " + distances[vertex]);
40         }
41     }
42     public static void main(String... arg)
43     {
44         int numofver = 0;
45         int source;
46         Scanner scanner = new Scanner(System.in);
47         System.out.println("Enter the number of vertices");

```

```

48     numofver = scanner.nextInt();
49     int matrix[][] = new int[numofver + 1][numofver + 1];
50     System.out.println("Enter the adjacency matrix");
51     for (int src = 1; src <= numofver; src++)
52     {
53         for (int dest = 1; dest <= numofver; dest++)
54         {
55             matrix[src][dest] = scanner.nextInt();
56             if (src == dest)
57             {
58                 matrix[src][dest] = 0;
59                 continue;
60             }
61             if (matrix[src][dest] == 0)
62             {
63                 matrix[src][dest] = MAX_VALUE;
64             }
65         }
66     }
67     System.out.println("Enter the source vertex");
68     source = scanner.nextInt();
69     DistanceVectorRouting dvr = new DistanceVectorRouting(numofver);
70     dvr.BellmanFordEvaluation(source, matrix);
71     scanner.close();
72 }
73 }
```

OUTPUT

```

Problems Javadoc Declaration Console X
<terminated> DistanceVectorRouting (1) [Java Application] C:\Program Files\Java\jdk-18.0.1\bin\javaw.exe (08-Apr-2025)
Enter the number of vertices
4
Enter the adjacency matrix
0 1 3 5
4 0 3 5
1 0 4 3
3 4 1 5
Enter the source vertex
2
distance from source 2 to 1 is 4
distance from source 2 to 2 is 0
distance from source 2 to 3 is 3
distance from source 2 to 4 is 5

```

Figure 13: Distance Vector Routing

8 Write a Java program to Dijkstra's Shortest Path Routing protocol

Dijkstra's Algorithm

Dijkstra's algorithm is a graph-based routing algorithm used to find the shortest path from a source router to all other routers in a network. It is commonly implemented in link-state routing protocols, such as **OSPF (Open Shortest Path First)**.

How It Works

Initialization

- Set the distance to the **source node** as 0.
- Set the distance to all other nodes as ∞ (infinity).
- Mark all nodes as **unvisited**.

Visit Nodes

- Repeatedly select the unvisited node with the smallest known distance.
- For each unvisited neighbor of the current node:
 - Calculate the tentative distance through the current node.
 - If the new distance is smaller than the previously recorded distance, update it.
- Mark the current node as **visited**.
- Repeat the process until all nodes are visited.

Java Program Highlights

The Java implementation of Dijkstra's algorithm includes the following features:

- Accepts a **cost matrix**, where `graph[i][j]` represents the cost between routers i and j .
- Uses an array `dist[]` to track the shortest known distance from the source router to every other router.
- A helper function `minDistance()` identifies the unvisited router with the smallest tentative distance.
- Updates each neighbor's distance if a shorter path through the current router is found.

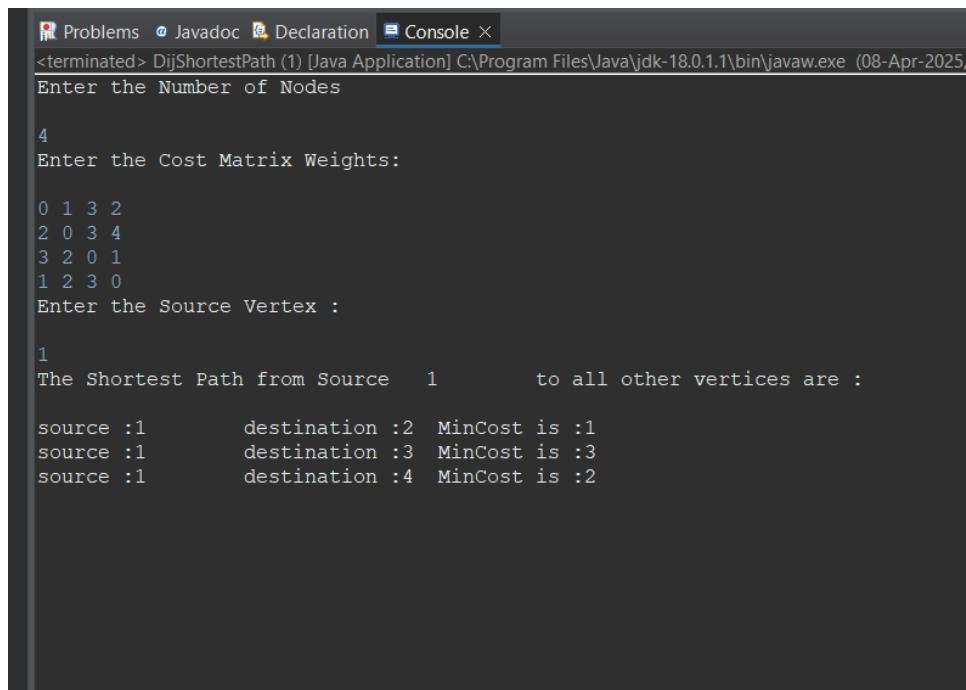
```

1 // DijShortestPath.java
2 import java.util.*;
3 public class Dijk
4 {
5     public int distance[] = new int[10];
6     public int cost[][] = new int[10][10];
7     public void calc(int n,int s)
8     {
9         int flag[] = new int[n+1];
10        int i,minpos=1,k,c,minimum;
11
12        for(i=1;i<=n;i++)
13        {
14            flag[i]=0;
15            this.distance[i]=this.cost[s][i];
16        }
17        c=2;
18        while(c<=n)
19        {
20            minimum=99;
21            for(k=1;k<=n;k++)
22            {
23                if(this.distance[k]<minimum && flag[k]!=1)
24                {
25                    minimum=this.distance[i];
26                    minpos=k;
27                }
28            }
29            flag[minpos]=1;
30            c++;
31            for(k=1;k<=n;k++)
32            {
33                if(this.distance[minpos]+this.cost[minpos][k] < this.distance[k] &&
34                    flag[k]!=1 )
35                this.distance[k]=this.distance[minpos]+this.cost[minpos][k];
36            }
37        }
38    }
39 @SuppressWarnings("resource")
40 public static void main(String args[])
41 {
42     int nodes,source,i,j;
43     Scanner in = new Scanner(System.in);
44     System.out.println("Enter the Number of Nodes \n");
45     nodes = in.nextInt();
46     Dijk d = new Dijk();

```

```
47 System.out.println("Enter the Cost Matrix Weights: \n");
48 for(i=1;i<=nodes;i++)
49 for(j=1;j<=nodes;j++)
50 {
51     d.cost[i][j]=in.nextInt();
52     if(d.cost[i][j]==0)
53         d.cost[i][j]=999;
54 }
55 System.out.println("Enter the Source Vertex :\n");
56 source=in.nextInt();
57
58 d.calc(nodes,source);
59 System.out.println("The Shortest Path from Source \t"+source+"\t to
       all other vertices are : \n");
60 for(i=1;i<=nodes;i++)
61 if(i!=source)
62 System.out.println("source :"+source+"\t destination :"+i+"\t MinCost
       is :"+d.distance[i]+\t");
63
64
65 }
66 }
```

OUTPUT



The screenshot shows a Java application window titled "DijShortestPath (1) [Java Application]". The console tab is active, displaying the following output:

```
<terminated> DijShortestPath (1) [Java Application] C:\Program Files\Java\jdk-18.0.1.1\bin\javaw.exe (08-Apr-2025)
Enter the Number of Nodes
4
Enter the Cost Matrix Weights:
0 1 3 2
2 0 3 4
3 2 0 1
1 2 3 0
Enter the Source Vertex :
1
The Shortest Path from Source 1 to all other vertices are :
source :1 destination :2 MinCost is :1
source :1 destination :3 MinCost is :3
source :1 destination :4 MinCost is :2
```

Figure 14: Dijkstra's

PART - B

Network Simulator-3

Network Simulator (NS-3)

The ns-3 (Network Simulator 3) is an open-source discrete-event network simulator designed for research and educational purposes. It enables users to simulate how packet-based networks operate, providing detailed models for Internet protocols and other networking mechanisms.

The primary advantage of ns-3 lies in its ability to simulate real-world networking scenarios in a controlled environment. This makes it especially valuable for:

- Conducting experiments that are difficult or costly to perform on actual hardware.
- Testing and evaluating protocols in a reproducible setting.
- Understanding the behavior of complex network systems.

While ns-3 emphasizes Internet protocol modeling, it also supports simulations for non-Internet-based networks, making it flexible for a wide range of research areas.

Figure 14 illustrates the general execution flow of an ns-3 simulation program. The simulation begins with configuring the network topology, including nodes, devices, channels, and protocol stacks. Applications (e.g., traffic generators) are then installed on the nodes, and simulation parameters such as timing and tracing are set. The simulator is run using a discrete-event engine, which processes all scheduled events. Finally, the results can be analyzed through trace files or visualized using tools like NetAnim.

This structured approach enables researchers to thoroughly analyze network performance, behavior under load, protocol interactions, and fault scenarios — all without the need for a physical testbed.

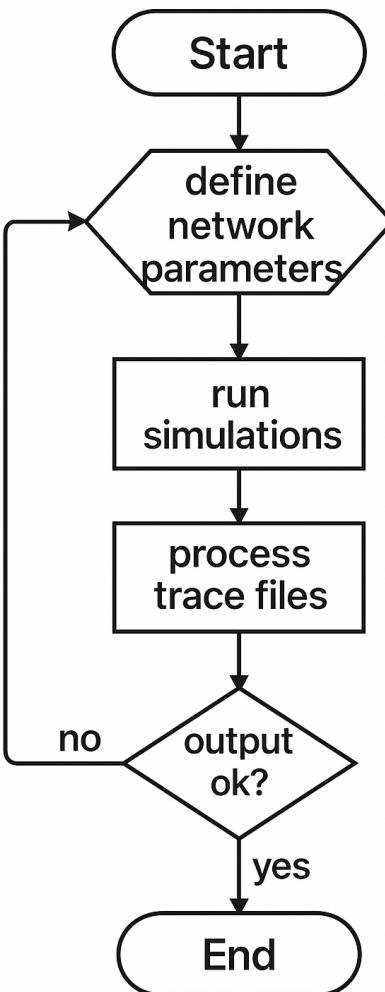


Figure 15: Flow chart of a typical ns-3 program execution process.

Distinguishing Features of NS-3

Several network simulators exist for conducting simulation-based research and performance evaluations. However, ns-3 stands out due to the following distinguishing features:

1. ns-3 is designed as a collection of modular libraries, allowing integration with other external software tools and libraries. Unlike simulation platforms that provide an all-in-one graphical user interface (GUI), ns-3 promotes modularity and extensibility.
2. External tools can be used with ns-3 for animation, data analysis, and visualization. This includes tools like NetAnim and PyViz for simulation visualization, and Gnuplot or Python-based tools for post-simulation analysis.
3. Users interact with ns-3 primarily through the command line, and simulations are written

in C++ and/or Python.

4. ns-3 is primarily designed for Linux systems, though it also supports FreeBSD and Cygwin for Windows. Native Windows support via Visual Studio is under development.

Important Points about NS-3

- ns-3 is not backward compatible with ns-2; it is a complete redesign built from the ground up to replace ns-2.
- The core simulation engine is written in C++, and Python can optionally be used as an interface for scripting and configuration.
- Many architectural limitations and bugs present in ns-2 are addressed in ns-3, making it more robust and modern.
- The number of contributed or community-developed modules in ns-3 is currently smaller compared to ns-2, but it is steadily growing.
- ns-2's use of a dual-language architecture (C++ and OTcl) made debugging more difficult. In contrast, ns-3's single-language design (C++ with optional Python) simplifies development and debugging.
- ns-3 offers an emulation mode, enabling interaction with real systems and live networks, which enhances its applicability in hybrid simulation-emulation environments.

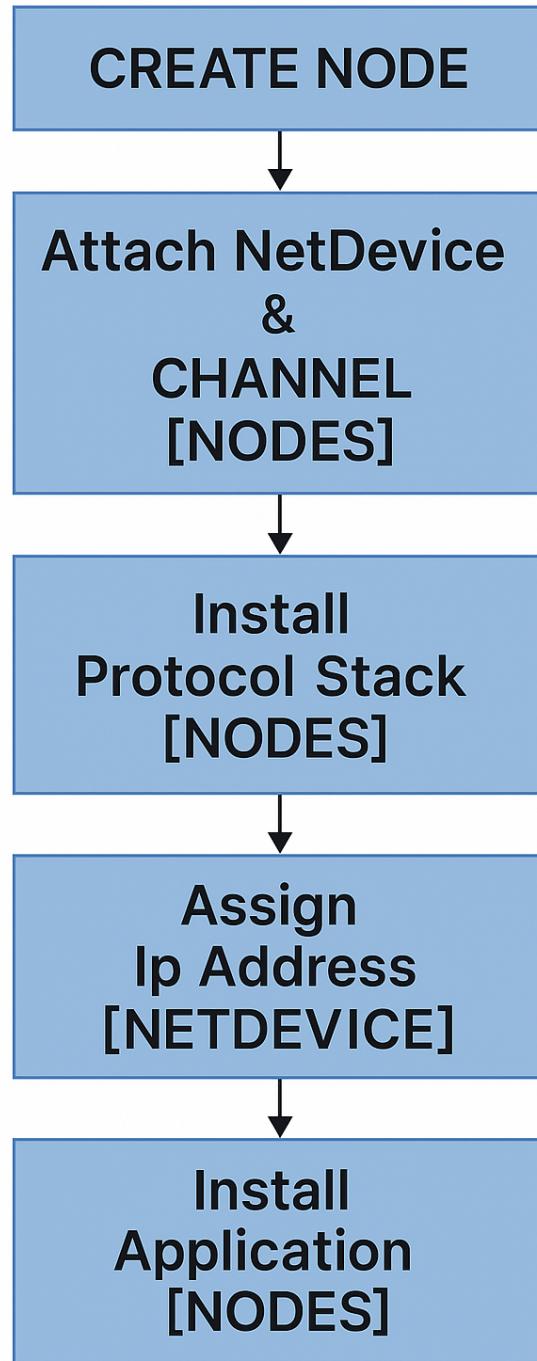


Figure 16: NS - 3 Program Structure flow chart

Steps to Install NS-3 (Version 3.27)

To install NS-3 on an Ubuntu system, follow the steps below:

Step 1: Open the Terminal

Once Ubuntu is installed, open the terminal using the shortcut:

Ctrl + Alt + T

Step 2: Install Prerequisite Packages

Update the system and install all required dependencies by executing the following commands:

```
sudo apt-get update
sudo apt-get install gcc g++ python
sudo apt-get install mercurial python-setuptools git
sudo apt-get install qt5-default
sudo apt-get install python-pygraphviz python-kiwi python-pygoocanvas
libgoocanvas-
dev ipython
sudo apt-get install openmpi-bin openmpi-common openmpi-doc libopenmpi-dev
sudo apt-get install autoconf cvs bzip2 unrar
sudo apt-get install gdb valgrind
sudo apt-get install uncrustify
sudo apt-get install doxygen graphviz imagemagick
sudo apt-get install texlive texlive-extra-utils texlive-latex-extra \
texlive-font-utils texlive-lang-portuguese dvipng
sudo apt-get install python-sphinx dia
sudo apt-get install gsl-bin libgsl2 libgsl-dev
sudo apt-get install flex bison libfl-dev
sudo apt-get install tcpdump
sudo apt-get install sqlite sqlite3 libsqlite3-dev
sudo apt-get install libxml2 libxml2-dev
sudo apt-get install cmake libc6-dev libc6-dev-i386 libclang-dev
sudo pip install cxxfilt
sudo apt-get install libgtk2.0-0 libgtk2.0-dev
sudo apt-get install vtun lxc
sudo apt-get install libboost-signals-dev libboost-filesystem-dev
```

Step 3: Download NS-3 (Version 3.27)

Create a directory for NS-3 and download the release:

```
mkdir ns3
cd ns3
wget https://www.nsnam.org/release/ns-allinone-3.27.tar.bz2
tar xjf ns-allinone-3.27.tar.bz2
cd ns-allinone-3.27/
ls
```

You will find a file named build.py along with other necessary files.

Step 4: Build NS-3 with Examples and Tests

Run the following command to build NS-3 along with its examples and tests:

```
./build.py --enable-examples --enable-tests
```

If the build is successful, you will see the message:

```
Build finished successfully
```

Step 5: Configure with Waf Build Tool

Navigate to the NS-3 directory and configure it using the Waf build system:

```
cd ns-3.27
./waf -d debug --enable-examples --enable-tests configure
```

Step 6: Build with Waf (Optional)

To build the simulator using Waf:

```
./waf
```

Step 7: Verify the Installation

Run the test script to ensure everything is working correctly:

```
./test.py
```

If all tests pass successfully, your NS-3 installation is complete!

NS-3 installation done successfully

Steps to Install NetAnim

To visualize NS-3 network simulations, NetAnim can be installed using the following steps:

Step 1: Open the Terminal

Open your terminal using the keyboard shortcut:

Ctrl + Alt + T

Step 2: Install Required Dependencies

Install the necessary Qt4 and XML libraries by running:

```
sudo apt-get install qt4-qmake  
sudo apt-get install libqt4-dev  
sudo apt-get install libxml2-dev
```

Step 3: Navigate to the NetAnim Directory

Change the directory to where NetAnim is located within your NS-3 package:

```
cd ns3/ns-allinone-3.27/netanim-3.108
```

Step 4: Build NetAnim

Clean any previous builds, configure the project, and compile using the following commands:

```
make clean  
qmake NetAnim.pro  
make
```

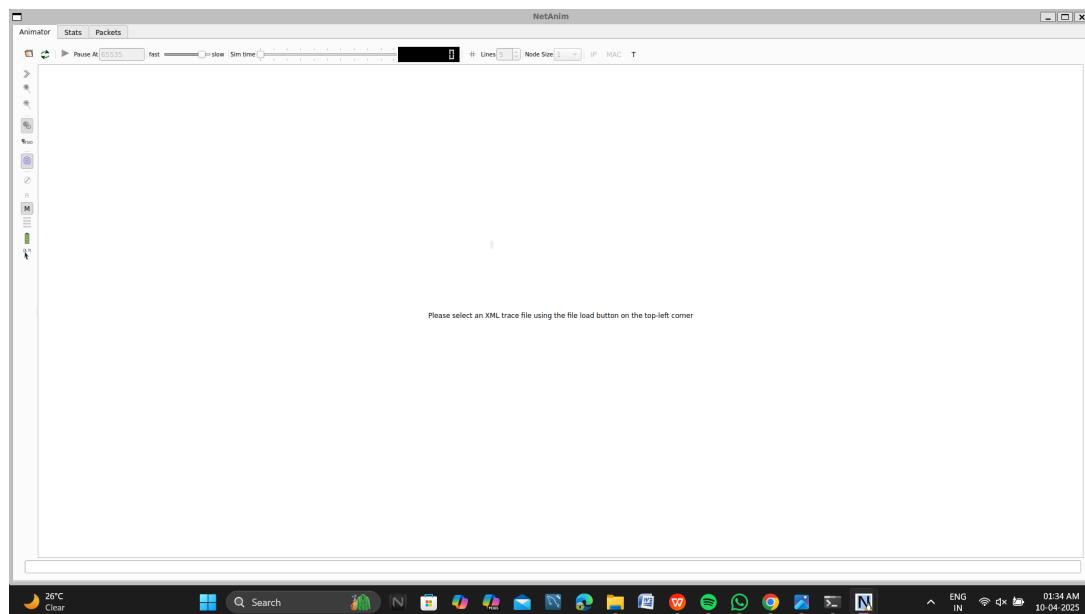
Step 5: Run NetAnim

Once the build is successful, launch NetAnim with:

```
./NetAnim
```

NetAnim should now be ready to use for visualizing NS-3 simulations.

SNAPSHOT



Steps to Install TraceMetrics

TraceMetrics is a Java-based tool used for analyzing trace files generated by NS-2 and NS-3. Follow the steps below to install and run it:

Step 1: Open the Terminal

Use the shortcut to open your terminal:

Ctrl + Alt + T

Step 2: Download TraceMetrics

Visit the official SourceForge page to download the tool:

<http://sourceforge.net/projects/tracemetrics/>

Step 3: Extract the Archive

After downloading, open the .zip file using the Archive Manager and extract the files to your desired location.

Step 4: Navigate to the TraceMetrics Directory

In the terminal, navigate to the extracted directory:

cd tracemetrics-1.4.0/

Step 5: Run TraceMetrics

Execute the following command to launch TraceMetrics:

```
java -jar "tracemetrics.jar"
```

The GUI interface of TraceMetrics should now open, allowing you to load and analyze NS simulation trace files.

9 Write a C++ program to connect two nodes on NS-3 (for practice only)

A simple star-like topology in NS-3 involving three nodes. One node acts as the central node, while the other two are connected directly to it. This connects two nodes using a point-to-point link. This is the simplest form of communication topology used in NS-3 and serves as a foundation for more complex simulations.

Network Topology

The network contains:

- Node0: One end of the communication (e.g., client)
- Node1: The other end (e.g., server)

The connection is as follows:

Node0 \leftrightarrow Node1

Simulation Details

Node Creation

Two nodes are created using the NS-3 NodeContainer class.

Link Configuration

A single point-to-point link is set up between Node0 and Node1 with specified attributes like data rate (e.g., 5Mbps) and delay (e.g., 2ms).

Internet Stack and Addressing

Both nodes are installed with the Internet stack. The point-to-point link is assigned a subnet (e.g., 10.1.1.0/24) and each node receives an IP address.

```

1 #include "ns3/core-module.h"
2 #include "ns3/network-module.h"
3 #include "ns3/internet-module.h"
4 #include "ns3/point-to-point-module.h"
5 #include "ns3/applications-module.h"
6 #include "ns3/netanim-module.h"
7
8 using namespace ns3;
9
10 NS_LOG_COMPONENT_DEFINE ("FirstScriptExample");
11
12 int main (int argc, char *argv[])
13 {
14     CommandLine cmd;
15     cmd.Parse (argc, argv);
16
17     // Log settings
18     Time::SetResolution (Time::NS);
19     LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
20     LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
21
22     // Create two nodes
23     NodeContainer nodes;
24     nodes.Create (2);
25
26     // Setup point-to-point channel
27     PointToPointHelper pointToPoint;
28     pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("1Mbps"))
29         ;
30     pointToPoint.SetChannelAttribute ("Delay", StringValue ("1ms"));
31
32     // Install devices on nodes
33     NetDeviceContainer devices;
34     devices = pointToPoint.Install (nodes);
35
36     // Install internet stack
37     InternetStackHelper stack;
38     stack.Install (nodes);
39
40     // Assign IP addresses
41     Ipv4AddressHelper address;
42     address.SetBase ("10.1.1.0", "255.255.255.0");
43     Ipv4InterfaceContainer interfaces = address.Assign (devices);
44
45     // Setup UDP echo server on node 1
46     UdpEchoServerHelper echoServer (9);

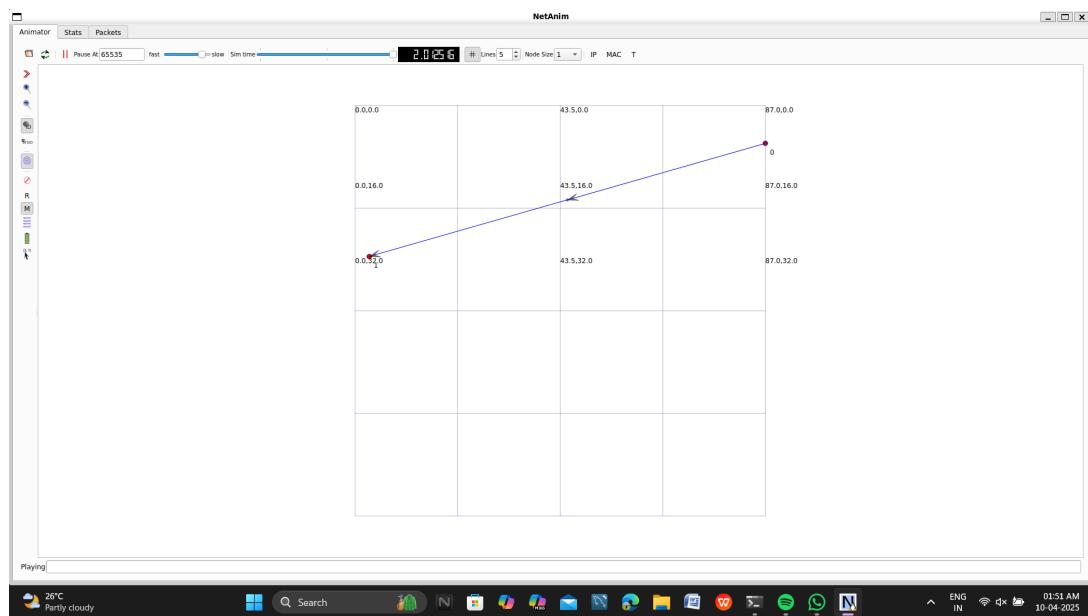
```

```

46     ApplicationContainer serverApps = echoServer.Install (nodes.Get (1)
47         );
48     serverApps.Start (Seconds (1.0));
49     serverApps.Stop (Seconds (10.0));
50
51     // Setup UDP echo client on node 0
52     UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
53     echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
54     echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
55     echoClient.SetAttribute ("PacketSize", UintegerValue (2048));
56     ApplicationContainer clientApps = echoClient.Install (nodes.Get (0)
57         );
58     clientApps.Start (Seconds (2.0));
59     clientApps.Stop (Seconds (10.0));
60
61     // Setup animation and tracing
62     AnimationInterface anim("np1.xml");
63     AsciiTraceHelper eventTraces;
64     pointToPoint.EnableAsciiAll(eventTraces.CreateFileStream("np1.tr"))
65         ;
66     pointToPoint.EnablePcapAll ("np1");
67
68     Simulator::Run ();
69     Simulator::Destroy ();
70     return 0;
71 }
```

Listing 1: Simple NS-3 Script to Connect Two Nodes

OUTPUT



10 Write a C++ program to connect three nodes considering one as a central node on NS-3 (for practice only)

In a topology in NS-3. Node 1 acts as the central node connecting node 0 and node 2 via different point-to-point links. The UDP echo server runs on node 2, and the client runs on node 0.

Network Topology

The network consists of:

- Node0: Peripheral node (could be a client)
- Node1: Central node (acts as a hub or router)
- Node2: Peripheral node (could be a server)

The connections are:

$$\text{Node0} \leftrightarrow \text{Node1} \leftrightarrow \text{Node2}$$

Each connection uses a point-to-point link.

Simulation Details

Node Creation

Three nodes are created. Node1 is treated as the central node, with point-to-point connections to both Node0 and Node2.

Link Configuration

Two point-to-point links are created:

- Between Node0 and Node1
- Between Node1 and Node2

Each link can have attributes such as 5Mbps data rate and 2ms delay.

Internet Stack and Addressing

All three nodes are installed with the Internet stack. Each point-to-point link is assigned a unique IP subnet.

Routing

Global routing is enabled so packets can be routed from Node0 to Node2 via Node1. The following NS-3 program demonstrates how to connect two nodes using a point-to-point link, configure a UDP echo server and client, and trace the simulation with NetAnim and PCAP.

```

1 #include "ns3/core-module.h"
2 #include "ns3/network-module.h"
3 #include "ns3/internet-module.h"
4 #include "ns3/point-to-point-module.h"
5 #include "ns3/applications-module.h"
6 #include "ns3/netanim-module.h"
7
8 using namespace ns3;
9
10 NS_LOG_COMPONENT_DEFINE ("SecondScriptExample");
11
12 int main (int argc, char *argv[])
13 {
14     CommandLine cmd;
15     cmd.Parse (argc, argv);
16
17     Time::SetResolution (Time::NS);
18     LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
19     LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
20
21     // Create three nodes
22     NodeContainer nodes;
23     nodes.Create (3);
24
25     // Setup two point-to-point links
26     PointToPointHelper p2p1;
27     p2p1.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
28     p2p1.SetChannelAttribute ("Delay", StringValue ("1ms"));
29
30     PointToPointHelper p2p2;
31     p2p2.SetDeviceAttribute ("DataRate", StringValue ("10Mbps"));
32     p2p2.SetChannelAttribute ("Delay", StringValue ("5ms"));
33
34     // Install Internet stack
35     InternetStackHelper stack;
36     stack.Install (nodes);
37
38     // Assign IP addresses and install devices
39     Ipv4AddressHelper address;
40     NetDeviceContainer devices;
41
42     // First link: node 0 <--> node 1
43     address.SetBase ("10.1.1.0", "255.255.255.0");
44     devices = p2p1.Install (nodes.Get (0), nodes.Get (1));
45     Ipv4InterfaceContainer interfaces = address.Assign (devices);
46
47     // Second link: node 1 <--> node 2

```

```

48     address.SetBase ("10.1.2.0", "255.255.255.0");
49     devices = p2p2.Install (nodes.Get (1), nodes.Get (2));
50     interfaces = address.Assign (devices);
51
52     // Populate routing tables
53     Ipv4GlobalRoutingHelper::PopulateRoutingTables ();
54
55     // Install UDP echo server on node 2
56     UdpEchoServerHelper echoServer (9);
57     ApplicationContainer serverApps = echoServer.Install (nodes.Get (2)
58         );
58     serverApps.Start (Seconds (1.0));
59     serverApps.Stop (Seconds (10.0));
60
61     // Install UDP echo client on node 0 targeting node 2
62     UdpEchoClientHelper echoClient (interfaces.GetAddress (1), 9);
63     echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
64     echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
65     echoClient.SetAttribute ("PacketSize", UintegerValue (1024));
66     ApplicationContainer clientApps = echoClient.Install (nodes.Get (0)
67         );
67     clientApps.Start (Seconds (2.0));
68     clientApps.Stop (Seconds (10.0));
69
70     // Animation and tracing
71     AnimationInterface anim("np2.xml");
72     AsciiTraceHelper eventTraces;
73     p2p1.EnableAsciiAll(eventTraces.CreateFileStream("np2.tr"));
74     p2p1.EnablePcapAll ("np2");
75
76     Simulator::Run ();
77     Simulator::Destroy ();
78     return 0;
79 }
```

Listing 2: Three Nodes with One Central Node in NS-3

OUTPUT

11 Write a C++ program to implement a star topology on NS-3

A star topology with one central hub and multiple spokes using NS-3. Each spoke communicates with the hub node using TCP applications. A star topology consists of a central node (hub or switch) that is directly connected to multiple peripheral nodes.

Network Topology

In a star topology:

- One central node is connected to all other nodes.
- Peripheral nodes communicate with each other through the central node.
- The failure of the central node results in network breakdown, but individual node failures do not affect the network.

Simulation Details

Node Creation

Multiple peripheral nodes (e.g., five) and one central node are created. The central node serves as a communication hub.

Link Configuration

Each peripheral node is connected to the central node using individual point-to-point links. These links can be configured with attributes like data rate and delay.

Internet Stack and Addressing

The Internet stack is installed on all nodes. Each link is assigned a unique subnet, and IP addresses are configured accordingly.

Routing

Global routing is used to populate routing tables. This ensures that any peripheral node can communicate with another through the central node.

Applications

A UDP Echo Server is installed on one peripheral node, and a UDP Echo Client is installed on another. Communication is routed via the central node.

```

1 #include "ns3/core-module.h"
2 #include "ns3/network-module.h"
3 #include "ns3/netanim-module.h"
4 #include "ns3/internet-module.h"
5 #include "ns3/point-to-point-module.h"
6 #include "ns3/applications-module.h"
7 #include "ns3/point-to-point-layout-module.h"

8
9 // Network Star topology (default)
10 //
11 //      n2  n3  n4      .
12 //          \ | /
13 //              \|/
14 //      n1--- n0---n5      .
15 //          /|\
16 //          / | \
17 //      n8  n7  n6      .
18 //
19
20
21 using namespace ns3;
22
23 NS_LOG_COMPONENT_DEFINE ("Star");
24
25 int
26 main (int argc, char *argv[])
27 {
28
29     //
30     // Set up some default values for the simulation.
31     //
32     Config::SetDefault ("ns3::OnOffApplication::PacketSize",
33                         UintegerValue (137));
34
35     // ??? try and stick 15kb/s into the data rate
36     Config::SetDefault ("ns3::OnOffApplication::DataRate", StringValue (
37                         "14kb/s"));
38
39     // Default number of nodes in the star. Overridable by command line
40     // argument.
41
42     uint32_t nSpokes = 8;
43     PointToPointHelper pointToPoint;
44     pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
45     pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
46     PointToPointStarHelper star (nSpokes, pointToPoint);
47
48     NS_LOG_INFO ("Install internet stack on all nodes.");

```

```

45     InternetStackHelper internet;
46     star.InstallStack (internet);
47
48     NS_LOG_INFO ("Assign IP Addresses.");
49     star.AssignIpv4Addresses (Ipv4AddressHelper ("10.1.1.0", "
50         255.255.255.0"));
51
52     NS_LOG_INFO ("Create applications.");
53     //
54     // Create a packet sink on the star "hub" to receive packets.
55     //
56     uint16_t port = 50000;
57     Address hubLocalAddress (InetSocketAddress (Ipv4Address::GetAny (), ,
58         port));
59     PacketSinkHelper packetSinkHelper ("ns3::TcpSocketFactory",
60         hubLocalAddress);
61     ApplicationContainer hubApp = packetSinkHelper.Install (star.GetHub
62         ());
63     hubApp.Start (Seconds (1.0));
64     hubApp.Stop (Seconds (10.0));
65
66     //
67     // Create OnOff applications to send TCP to the hub, one on each
68     // spoke node.
69     //
70     OnOffHelper onOffHelper ("ns3::TcpSocketFactory", Address ());
71     onOffHelper.SetAttribute ("OnTime", StringValue ("ns3::
72         ConstantRandomVariable[Constant=1]"));
73     onOffHelper.SetAttribute ("OffTime", StringValue ("ns3::
74         ConstantRandomVariable[Constant=0]"));
75
76     ApplicationContainer spokeApps;
77
78     for (uint32_t i = 0; i < star.SpokeCount (); ++i)
79     {
80         AddressValue remoteAddress (InetSocketAddress (star.
81             GetHubIpv4Address (i), port));
82         onOffHelper.SetAttribute ("Remote", remoteAddress);
83         spokeApps.Add (onOffHelper.Install (star.GetSpokeNode (i)));
84     }
85     spokeApps.Start (Seconds (1.0));
86     spokeApps.Stop (Seconds (10.0));
87
88     NS_LOG_INFO ("Enable static global routing.");
89     //
90     // Turn on global static routing so we can actually be routed across
91     // the star.

```

```
83     //  
84     Ipv4GlobalRoutingHelper::PopulateRoutingTables ();  
85  
86     NS_LOG_INFO ("Enable pcap tracing.");  
87     //  
88     // Do pcap tracing on all point-to-point devices on all nodes.  
89     //  
90     pointToPoint.EnablePcapAll ("star");  
91  
92  
93  
94 // Set the bounding box for animation  
95 star.BoundingBox (1, 1, 100, 100);  
96  
97 // Create the animation object and configure for specified output  
98 AnimationInterface anim ("star.xml");  
99  
100  
101 NS_LOG_INFO ("Run Simulation.");  
102 Simulator::Run ();  
103 Simulator::Destroy ();  
104 NS_LOG_INFO ("Done.");  
105  
106     return 0;  
107 }
```

Listing 3: Star Topology in NS-3

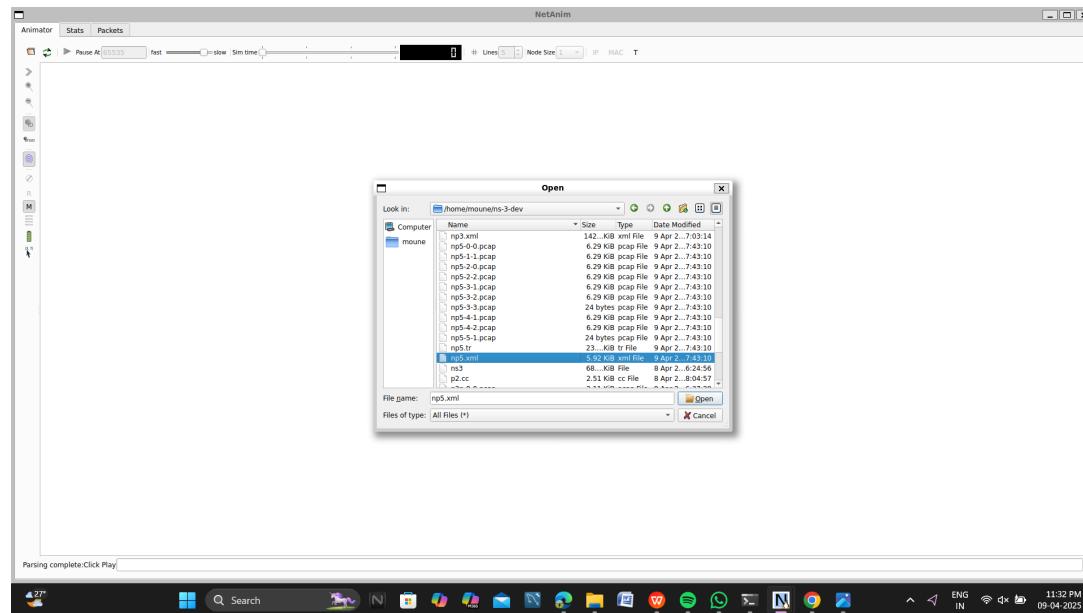
OUTPUT

In our system We have used these commands (Common for all programs)

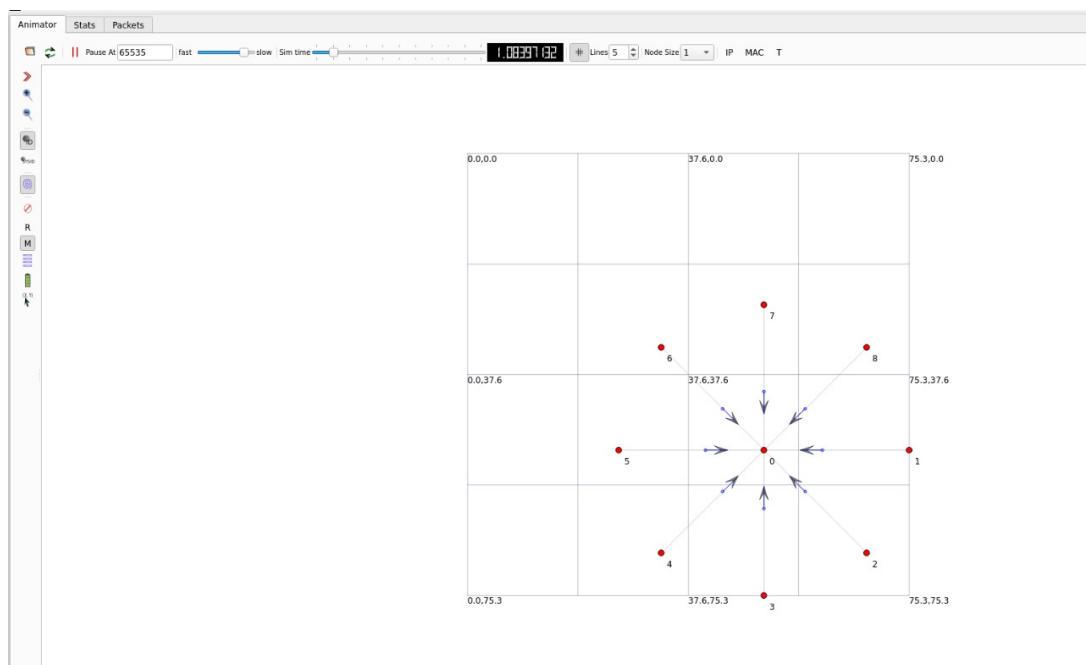
- Open Ubuntu Terminal
- ls
- cd ns-3-dev (File)
- ls
- **To create a file**
- cd scratch
- nano filename.cc
- Type a program
- Ctrl+X (To save)
- Y + Enter
- cd ..
- ./ns3 run scratch/filename.cc (Run Created in scratch)
- *It will start building.*
- **If file not found, use 3 commands below:**
- cd ns-3-dev
- cd build
- cmake ..
- **Then: After running**
- cd ~/netanim/build/bin
- ls
- ./netanim

OUTPUT

```
moune@LAPTOP-UICB0M3J: ~ +  
netanim ns-3-allinone ns-3-dev  
moune@LAPTOP-UICB0M3J: ~$ cd ns-3-dev  
moune@LAPTOP-UICB0M3J:~/ns-3-dev$ ls  
AUTHORS           VERSION      doc          np3-0-3.pcap  np3-5-0.pcap  np5-2-2.pcap  ns3          src  
CHANGES.md        bindings     examples      np3-0-4.pcap  np3-6-0.pcap  np5-3-1.pcap  p2.cc        test.py  
CMakeLists.txt    build        build        np1-0.pcap   np1-7-0.pcap  np5-3-2.pcap  p2p-0-0.pcap  utils  
CONTRIBUTING.md  contrib      contrib      np1-0.pcap   np3-0-6.pcap  np5-3-3.pcap  p2p-1-0.pcap  utils.py  
LICENSE          contrib      contrib      np1.tr       np3-0-7.pcap  np5-4-1.pcap  p3.xml  
MAC.cc            csmra-1.pcap npml.xml     np3-1-0.pcap  np3-8-0.pcap  np5-4-2.pcap  pyproject.toml  
README.md         csmra-2.pcap npml.xml     np3-1-0.pcap  np3-8-0.pcap  np5-5-1.pcap  scratch  
RELEASE-NOTES.md csmra-3.pcap npml.xml     np3-1-3.pcap  np3-8-0.pcap  np5-5-2.pcap  setup.cfg  
TWO.cc            csmra-4.pcap npml.xml     np3-1-4.pcap  np3-8-0.pcap  np5-6-0.pcap  setup.py  
moune@LAPTOP-UICB0M3J:~/ns-3-dev$ run scratch/p2.cc  
AnimationInterface: WARNING: Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:1 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:2 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:3 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:4 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:5 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:6 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:7 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:8 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:9 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:10 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:11 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:12 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:13 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:14 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:15 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:16 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:17 Does not have a mobility model. Use SetConstantPosition if it is stationary  
AnimationInterface: WARNING: Node:18 Does not have a mobility model. Use SetConstantPosition if it is stationary  
moune@LAPTOP-UICB0M3J:~/ns-3-dev$ cd ~/netanim/build/bin  
moune@LAPTOP-UICB0M3J:~/netanim/build/bin$ ls  
netanim  
moune@LAPTOP-UICB0M3J:~/netanim/build/bin$ ./netanim
```



OUTPUT



12 Write a C++ program to implement a bus topology on NS-3

A bus topology connects all devices to a single communication line where each node can directly communicate with any other node on the bus. A bus topology using CSMA (Carrier Sense Multiple Access) in NS-3. It connects nodes using a point-to-point link and a shared CSMA channel.

Network Topology

In a bus topology:

- All nodes share a single channel or communication medium.
- Only one device can transmit at a time to avoid collisions.
- This is typically implemented using CSMA (Carrier Sense Multiple Access) in NS-3.

Simulation Details

Node Creation

Multiple nodes (e.g., five) are created to simulate devices connected to a shared medium. These nodes will communicate with each other using a shared channel.

Channel Configuration

A single `CsmaChannel` is created and shared among all nodes. The CSMA (Carrier Sense Multiple Access) protocol ensures that only one device transmits at a time.

Internet Stack and Addressing

Each node is installed with an Internet stack and given a unique IP address from the same subnet.

Routing

Since all nodes are on the same LAN (Local Area Network), no intermediate routers or routing configuration is required.

Applications

One node can act as a server, and another as a client. A UDP Echo Server is set up on one node, and a UDP Echo Client is set up on another. Communication occurs over the shared CSMA channel.

```

1 #include "ns3/core-module.h"
2 #include "ns3/network-module.h"
3 #include "ns3/csma-module.h"
4 #include "ns3/internet-module.h"
5 #include "ns3/point-to-point-module.h"
6 #include "ns3/applications-module.h"
7 #include "ns3/ipv4-global-routing-helper.h"
8 #include "ns3/netanim-module.h"

9
10 // Default Network Bus Topology
11 //
12 //      10.1.1.0
13 // n0 ----- n1   n2   n3   n4
14 //      point-to-point |   |   |
15 //                           =====
16 //                           LAN 10.2.1.0 - csma
17
18
19 using namespace ns3;
20
21 NS_LOG_COMPONENT_DEFINE ("SecondScriptExample");
22 int
23 main (int argc, char *argv[])
24 {
25     LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO);
26     LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO);
27
28     NodeContainer p2pNodes;
29     p2pNodes.Create (2);
30
31     NodeContainer csmaNodes;
32     csmaNodes.Add (p2pNodes.Get (1));
33     csmaNodes.Create (3);
34
35     PointToPointHelper pointToPoint;
36     pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"));
37     pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
38
39     NetDeviceContainer p2pDevices;
40     p2pDevices = pointToPoint.Install (p2pNodes);
41
42     CsmaHelper csma;
43     csma.SetChannelAttribute ("DataRate", StringValue ("100Mbps"));
44     csma.SetChannelAttribute ("Delay", TimeValue (NanoSeconds (6560)));
45
46     NetDeviceContainer csmaDevices;
47     csmaDevices = csma.Install (csmaNodes);

```

```

48
49     InternetStackHelper stack;
50     stack.Install (p2pNodes.Get(0));
51     stack.Install (csmaNodes);
52
53     Ipv4AddressHelper address;
54     address.SetBase ("10.1.1.0", "255.255.255.0");
55     Ipv4InterfaceContainer p2pInterfaces;
56     p2pInterfaces = address.Assign (p2pDevices);
57
58     address.SetBase ("10.2.1.0", "255.255.255.0");
59     Ipv4InterfaceContainer csmaInterfaces;
60     csmaInterfaces = address.Assign (csmaDevices);
61
62     UdpEchoServerHelper echoServer (9);
63
64     ApplicationContainer serverApps = echoServer.Install (csmaNodes.Get
65         (3));
66     serverApps.Start (Seconds (1.0));
67     serverApps.Stop (Seconds (10.0));
68
69     UdpEchoClientHelper echoClient (csmaInterfaces.GetAddress (3), 9);
70     echoClient.SetAttribute ("MaxPackets", UintegerValue (1));
71     echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
72     echoClient.SetAttribute ("PacketSize", UintegerValue (1024));
73
74     ApplicationContainer clientApps = echoClient.Install (p2pNodes.Get
75         (0));
76     clientApps.Start (Seconds (2.0));
77     clientApps.Stop (Seconds (10.0));
78
79     Ipv4GlobalRoutingHelper::PopulateRoutingTables ();
80
81     pointToPoint.EnablePcapAll ("p2p");
82     csma.EnablePcapAll ("csma");
83
84     AnimationInterface anim("p3.xml");
85     anim.SetConstantPosition(p2pNodes.Get(0),10.0,10.0);
86     anim.SetConstantPosition(csmaNodes.Get(0),20.0,20.0);
87     anim.SetConstantPosition(csmaNodes.Get(1),30.0,30.0);
88     anim.SetConstantPosition(csmaNodes.Get(2),40.0,40.0);
89     anim.SetConstantPosition(csmaNodes.Get(3),50.0,50.0);
90
91     Simulator::Run ();
92     Simulator::Destroy ();
93     return0      ;

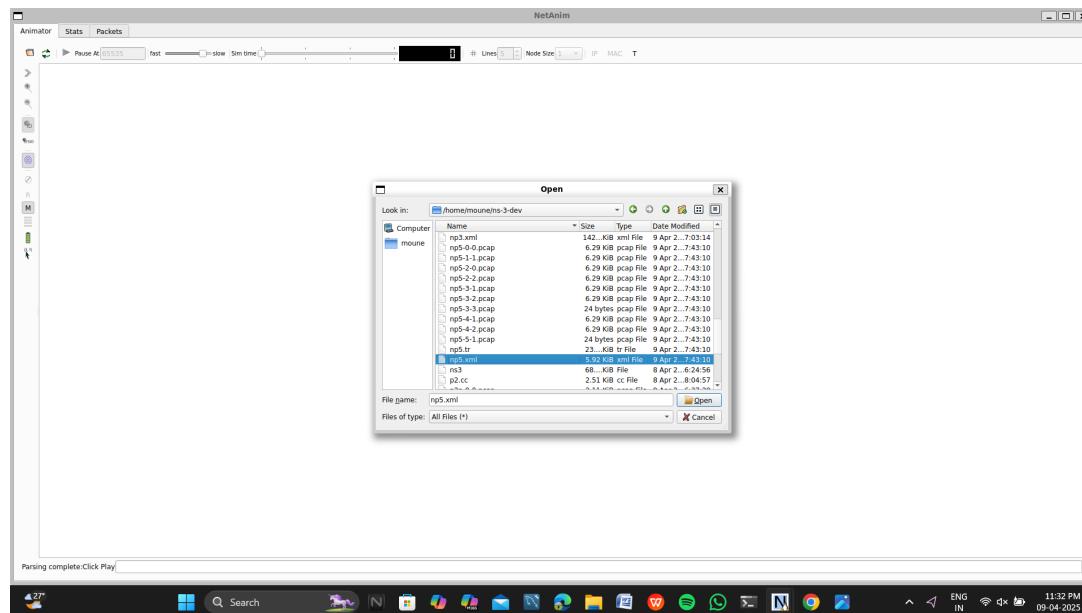
```

Listing 4: Bus Topology in NS-3

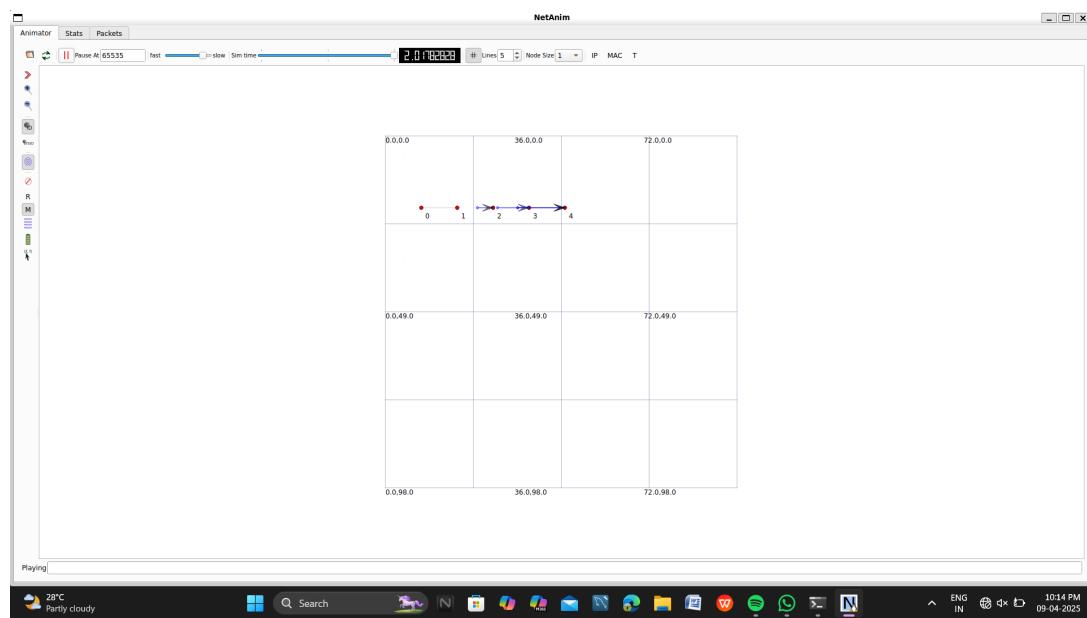
OUTPUT

```
moune@LAPTOP-UICB0MJ3:~$ ls
netanim ns-3 ns-3-allinone ns-3-dev
moune@LAPTOP-UICB0MJ3:~$ cd ns-3-dev
moune@LAPTOP-UICB0MJ3:~/ns-3-dev$ ls
AUTOMATICALLY GENERATED FILES:
    csma-1-0.pcap np3-0-4.pcap np5-0-0.pcap p2p-0-0.pcap
    csma-2-0.pcap np3-0-5.pcap np5-1-1.pcap p2p-1-0.pcap
    csma-3-0.pcap np3-0-6.pcap np5-2-0.pcap p2_xnl
    CONTRIBUTING.md csma-4-0.pcap np3-0-7.pcap np5-2-2.pcap pyproject.toml
    LICENSE doc np3-1-0.pcap np5-3-1.pcap scratch
    MAC.cc examples np3-2-0.pcap np5-3-2.pcap setup.cfg
    README.md np1-0-0.pcap np3-3-0.pcap np5-3-3.pcap setup.py
    RELEASE_NOTES.md np1-1-0.pcap np3-4-0.pcap np5-4-1.pcap src
    TWO.cc np1.tr np3-5-0.pcap np5-4-2.pcap test.py
    VERSION np1.xml np3-6-0.pcap np5-5-1.pcap utils
    bindings np3-0-0.pcap np3-7-0.pcap np5.tr utils.py
    build np3-0-1.pcap np3-8-0.pcap np5.xml
    build-support np3-0-2.pcap np3.tr ns3
    control np3-0-3.pcap np3.xnl p2_cc
    contrib np3-0-4.pcap np3.xnl p2_cc
    config np3-0-5.pcap np3.xnl p2_cc
    core np3-0-6.pcap np3.xnl p2_cc
    Documentation np3-0-7.pcap np3.xnl p2_cc
    DocumentationInterface np3-0-8.pcap np3.xnl p2_cc
    DocumentationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
    DocumentationInterface WARNING:Node:1 Does not have a mobility model. Use SetConstantPosition if it is stationary
    DocumentationInterface WARNING:Node:2 Does not have a mobility model. Use SetConstantPosition if it is stationary
    DocumentationInterface WARNING:Node:3 Does not have a mobility model. Use SetConstantPosition if it is stationary
    DocumentationInterface WARNING:Node:4 Does not have a mobility model. Use SetConstantPosition if it is stationary
    AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
    AnimationInterface WARNING:Node:1 Does not have a mobility model. Use SetConstantPosition if it is stationary
    AnimationInterface WARNING:Node:2 Does not have a mobility model. Use SetConstantPosition if it is stationary
    AnimationInterface WARNING:Node:3 Does not have a mobility model. Use SetConstantPosition if it is stationary
    AnimationInterface WARNING:Node:4 Does not have a mobility model. Use SetConstantPosition if it is stationary
    At time +2s client sent 1024 bytes to 10.2.1.4 port 9
    At time +2.0118s server received 1024 bytes from 10.1.1.1 port 49153
    At time +2.0118s server sent 1024 bytes to 10.1.1.1 port 49153
    At time +2.0220s client received 1024 bytes from 10.2.1.4 port 9
moune@LAPTOP-UICB0MJ3:~/ns-3-dev$ cd ~/netanim/build/bin
moune@LAPTOP-UICB0MJ3:~/netanim/build/bin$ ls
netanim
moune@LAPTOP-UICB0MJ3:~/netanim/build/bin$ ./netanim
```

26°C Mostly cloudy Search N W E S M F G H I J K L P Q R T ENG IN 02:40 AM 10-04-2025



OUTPUT



13 Write a C++ program showing the connection of two nodes and four routers such that the extreme nodes act as client and server on NS-3

Where two nodes are connected through four routers. The nodes at the ends act as client and server, respectively. where two end nodes are connected through four intermediate routers using point-to-point links. The two end nodes act as a client and a server respectively, communicating over a multihop network.

Network Topology

The network consists of:

- Two end nodes: Node0 (Client) and Node1 (Server)
- Four intermediate routers: R1, R2, R3, and R4
- Five point-to-point links connecting the entire path

The structure is illustrated as:

Node0 (Client) → R1 → R2 → R3 → R4 → Node1 (Server)

Each link is defined with a specified bandwidth and delay to emulate real network conditions.

Simulation Details

Node and Router Creation

Two end nodes are created to represent the client and server. Four additional nodes are created to serve as routers. This totals six nodes.

Link Configuration

Five point-to-point connections are configured between:

1. Node0 and Router1
2. Router1 and Router2
3. Router2 and Router3
4. Router3 and Router4
5. Router4 and Node1

Each link is configured with a data rate of 5Mbps and a delay of 2ms.

Internet Stack and Addressing

The Internet stack is installed on all nodes, and each link is assigned a unique IP subnet. This allows routing of packets across the network.

Routing

Global routing is used to automatically populate routing tables across all nodes. This ensures the client can reach the server through the intermediate routers.

Applications

A UDP Echo Server is installed on Node1 (Server) and a UDP Echo Client is installed on Node0 (Client). The client sends a number of packets to the server and receives replies, validating the multihop connection.

```

1 #include "ns3/core-module.h"
2 #include "ns3/network-module.h"
3 #include "ns3/csma-module.h"
4 #include "ns3/internet-module.h"
5 #include "ns3/point-to-point-module.h"
6 #include "ns3/applications-module.h"
7 #include "ns3/ipv4-global-routing-helper.h"
8 #include "ns3/netanim-module.h"
9
10 using namespace ns3;
11
12 NS_LOG_COMPONENT_DEFINE ("SecondScriptExample");
13
14 int main (int argc, char *argv[])
15 {
16     bool verbose = true;
17
18     if (verbose)
19     {
20         LogComponentEnable ("UdpEchoClientApplication", LOG_LEVEL_INFO)
21             ;
22         LogComponentEnable ("UdpEchoServerApplication", LOG_LEVEL_INFO)
23             ;
24     }
25
26     NodeContainer host, router;
27     host.Create (2);           // Two end-hosts: client and server
28     router.Create (4);        // Four intermediate routers
29
30     // Subnet 1: Host 0 to Router 0
31     NodeContainer subnet1;
32     subnet1.Add (host.Get (0));
33     subnet1.Add (router.Get (0));
34
35     PointToPointHelper pointToPoint;
36     pointToPoint.SetDeviceAttribute ("DataRate", StringValue ("5Mbps"))
37         ;
38     pointToPoint.SetChannelAttribute ("Delay", StringValue ("2ms"));
39
40     NetDeviceContainer subnet1Devices = pointToPoint.Install (subnet1);
41
42     InternetStackHelper stack;
43     stack.Install (router);
44     stack.Install (host);
45
46     Ipv4AddressHelper address1;
47     address1.SetBase ("10.1.1.0", "255.255.255.0");

```

```

45     Ipv4InterfaceContainer subnet1Interfaces = address1.Assign (
46         subnet1Devices);
47
48     // Subnet 2: Router 0 to Router 1
49     NodeContainer subnet2;
50     subnet2.Add (router.Get (0));
51     subnet2.Add (router.Get (1));
52     NetDeviceContainer subnet2Devices = pointToPoint.Install (subnet2);
53     Ipv4AddressHelper address2;
54     address2.SetBase ("10.1.2.0", "255.255.255.0");
55     Ipv4InterfaceContainer subnet2Interfaces = address2.Assign (
56         subnet2Devices);
57
58     // Subnet 3: Router 1 to Router 2
59     NodeContainer subnet3;
60     subnet3.Add (router.Get (1));
61     subnet3.Add (router.Get (2));
62     NetDeviceContainer subnet3Devices = pointToPoint.Install (subnet3);
63     Ipv4AddressHelper address3;
64     address3.SetBase ("10.1.3.0", "255.255.255.0");
65     Ipv4InterfaceContainer subnet3Interfaces = address3.Assign (
66         subnet3Devices);
67
68     // Subnet 4: Router 1 to Router 3
69     NodeContainer subnet4;
70     subnet4.Add (router.Get (1));
71     subnet4.Add (router.Get (3));
72     NetDeviceContainer subnet4Devices = pointToPoint.Install (subnet4);
73     Ipv4AddressHelper address4;
74     address4.SetBase ("10.1.4.0", "255.255.255.0");
75     Ipv4InterfaceContainer subnet4Interfaces = address4.Assign (
76         subnet4Devices);
77
78     // Subnet 5: Router 2 to Host 1
79     NodeContainer subnet5;
80     subnet5.Add (router.Get (2));
81     subnet5.Add (host.Get (1));
82     NetDeviceContainer subnet5Devices = pointToPoint.Install (subnet5);
83     Ipv4AddressHelper address5;
84     address5.SetBase ("10.1.5.0", "255.255.255.0");
85     Ipv4InterfaceContainer subnet5Interfaces = address5.Assign (
86         subnet5Devices);
87
88     // Server Application on Host 1
89     UdpEchoServerHelper echoServer (9);
90     ApplicationContainer serverApps = echoServer.Install (subnet5.Get
91         (1));

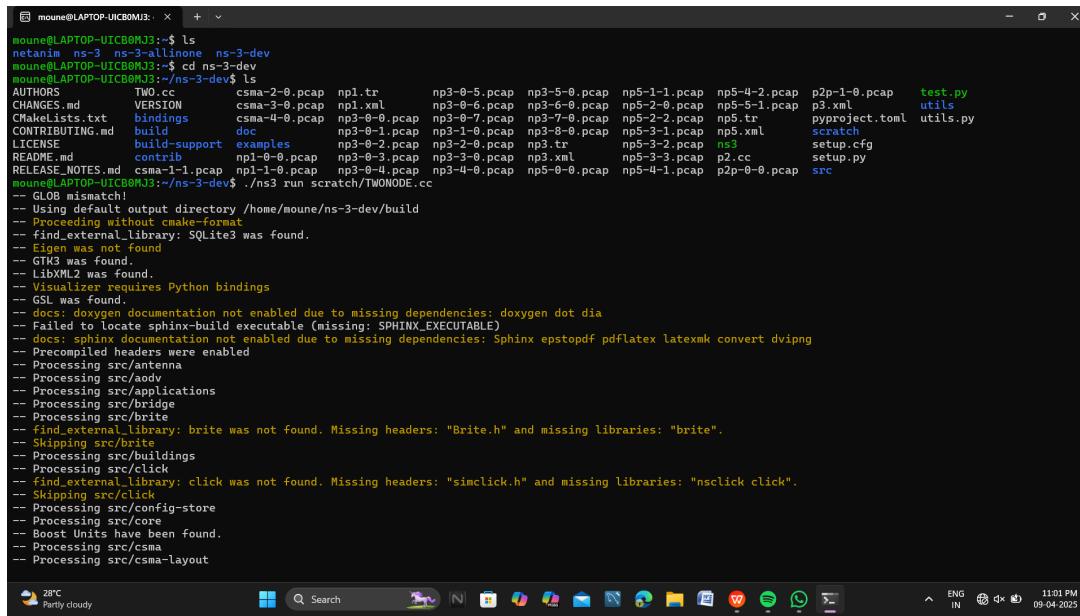
```

```

86     serverApps.Start (Seconds (1.0));
87     serverApps.Stop (Seconds (10.0));
88
89 // Client Application on Host 0
90 UdpEchoClientHelper echoClient (subnet5Interfaces.GetAddress (1),
91     9);
91 echoClient.SetAttribute ("MaxPackets", UintegerValue (3));
92 echoClient.SetAttribute ("Interval", TimeValue (Seconds (1.0)));
93 echoClient.SetAttribute ("PacketSize", UintegerValue (1024));
94 ApplicationContainer clientApps = echoClient.Install (subnet1.Get
95     (0));
95 clientApps.Start (Seconds (1.0));
96 clientApps.Stop (Seconds (10.0));
97
98 Ipv4GlobalRoutingHelper::PopulateRoutingTables ();
99
100 AnimationInterface anim("np5.xml");
101
102 AsciiTraceHelper eventTraces;
103 pointToPoint.EnableAsciiAll(eventTraces.CreateFileStream("np5.tr"))
104     ;
104 pointToPoint.EnablePcapAll ("np5");
105
106 Simulator::Run ();
107 Simulator::Destroy ();
108
109 return 0;
110 }
```

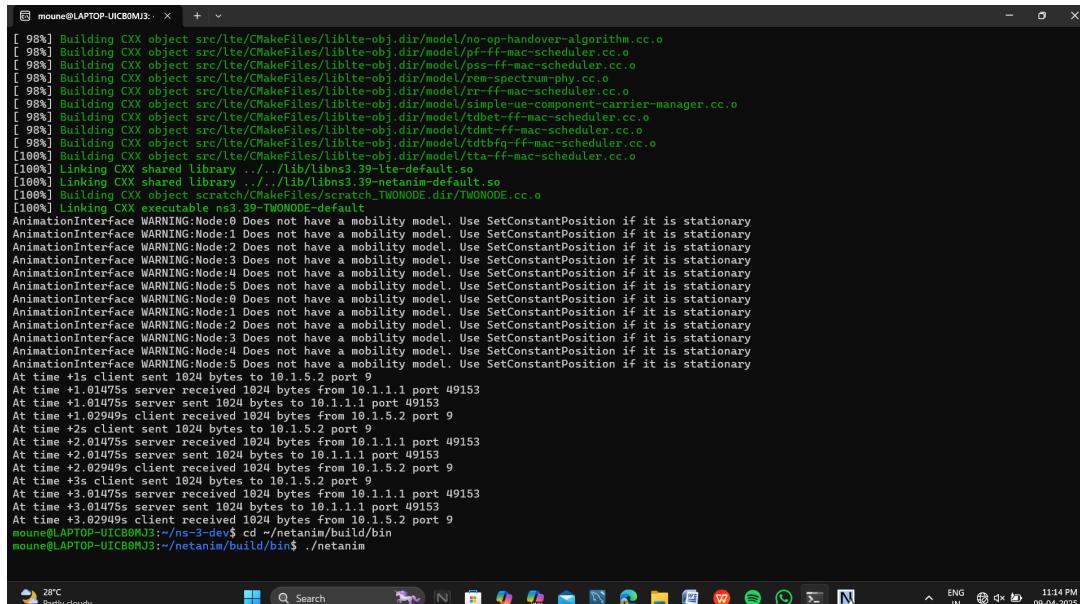
Listing 5: Two Nodes and Four Routers in NS-3

OUTPUT



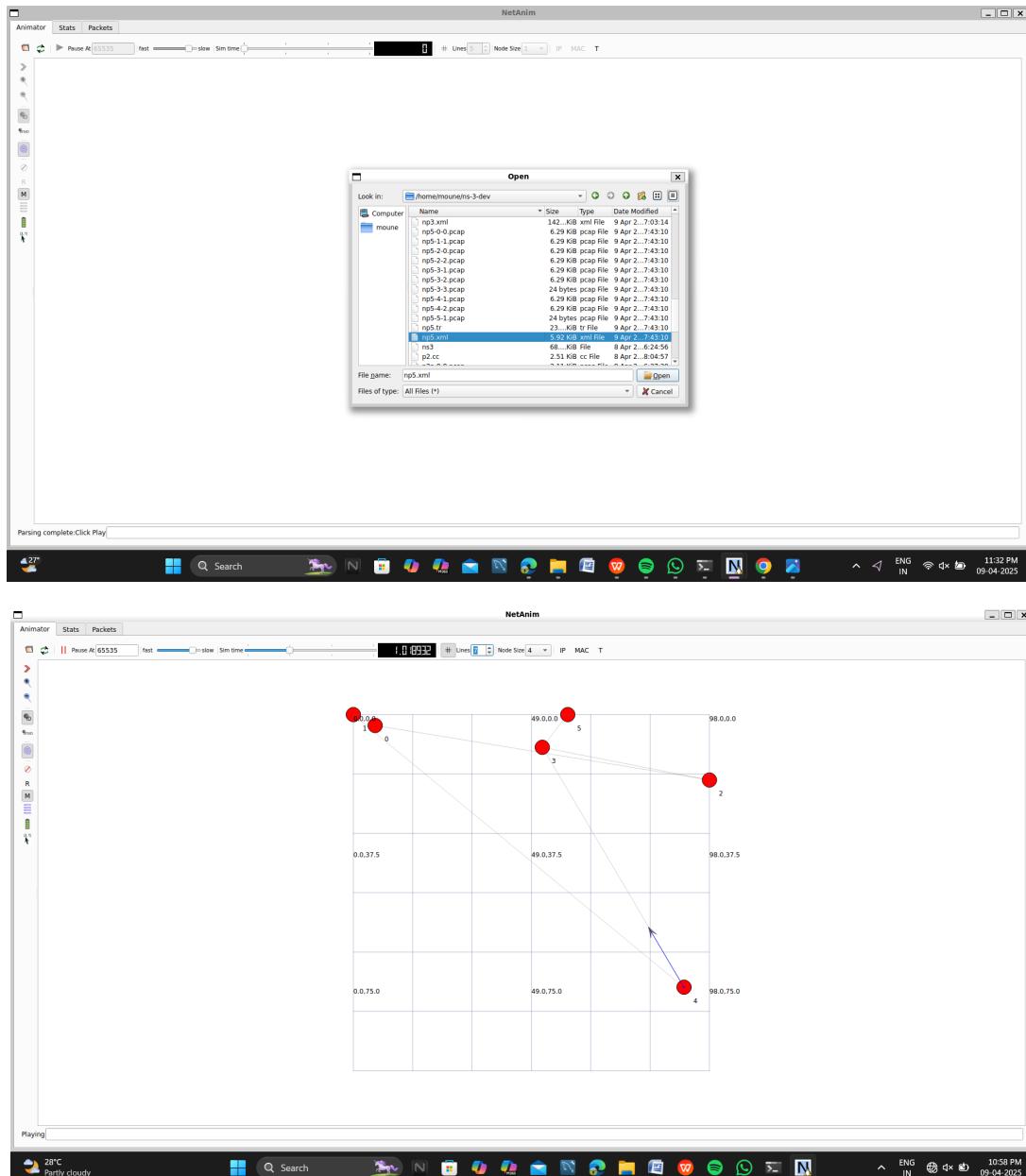
```
moune@LAPTOP-UICB0MJ3:~$ ls
netanim ns-3 ns-3-allinone ns-3-dev
moune@LAPTOP-UICB0MJ3:~$ cd ns-3-dev
moune@LAPTOP-UICB0MJ3:~/ns-3-dev$ ls
AUTHORS      TWO.cc        csma-2-0.pcap  np1.tr      np3-0-5.pcap  np3-5-0.pcap  np5-1-1.pcap  np5-4-2.pcap  p2p-1-0.pcap  test.py
CHANGES.md    VERSION      csma-3-0.pcap  np1.xml    np3-0-6.pcap  np3-6-0.pcap  np5-2-0.pcap  np5-5-1.pcap  p3.xml      utils
CMakeLists.txt bindings     csma-4-0.pcap  np3-0-0.pcap  np3-0-7.pcap  np3-7-0.pcap  np5-2-2.pcap  np5.tr      pyproject.toml  utils.py
CONTRIBUTING.md build       doc          np3-0-1.pcap  np3-1-0.pcap  np3-8-0.pcap  np5-3-1.pcap  np5.xml      scratch
LICENSE       build-support examples   np3-0-2.pcap  np3-2-0.pcap  np3.tr      np5-3-2.pcap  ns3        setup.cfg
README.md     contrib      np1-0-0.pcap  np3-0-3.pcap  np3-3-0.pcap  np3.xml      np5-3-3.pcap  p2.cc      setup.py
RELEASE_NOTES.md csma-1-1.pcap  np1-1-0.pcap  np3-0-4.pcap  np3-4-0.pcap  np5-0-0.pcap  np5-4-1.pcap  p2p-0-0.pcap  src

moune@LAPTOP-UICB0MJ3:~/ns-3-dev$ ./ns3 run scratch/TWONODE.cc
-- GLOB mismatch!
-- Using default output directory /home/moune/ns-3-dev/build
-- Proceeding without cmake-format
-- Found Python library: SQLite3 was found.
-- Eigen was not found.
-- GTK3 was found.
-- LibXML2 was found.
-- Visualizer requires Python bindings
-- GSL was found.
-- docs: doxygen documentation not enabled due to missing dependencies: doxygen dot dia
-- Failed to locate sphinx-build executable (Missing: SPHINX_EXECUTABLE)
-- docs: sphinx documentation not enabled due to missing dependencies: Sphinx epstopdf pdflatex latexmk convert dvipng
-- Precompiled headers were enabled
-- Processing src/antenna
-- Processing src/aodv
-- Processing src/applications
-- Processing src/bridge
-- Processing src/brite
-- find_external_library: brite was not found. Missing headers: "Brite.h" and missing libraries: "brite".
-- Skipping src/brite
-- Processing src/buildings
-- Processing src/click
-- find_external_library: click was not found. Missing headers: "simclick.h" and missing libraries: "nsclick click".
-- Skipping src/click
-- Processing src/config-store
-- Processing src/core
-- Boost Units have been found.
-- Processing src/csma
-- Processing src/csma-layout
```



```
[ 98%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/no-op-handover-algorithm.cc.o
[ 98%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/pf-ff-mac-scheduler.cc.o
[ 98%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/pss-ff-mac-scheduler.cc.o
[ 98%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/rem-spectrum-phy.cc.o
[ 98%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/rri-ff-mac-scheduler.cc.o
[ 98%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/simple-ue-component-carrier-manager.cc.o
[ 98%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/tia-ff-mac-scheduler.cc.o
[ 98%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/tia-ff-mac-scheduler.cc.o
[ 98%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/tddfbfq-ff-mac-scheduler.cc.o
[ 98%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/tia-ff-mac-scheduler.cc.o
[100%] Building CXX object src/lte/CMakeFiles/Liblte-obj.dir/model/tia-ff-mac-scheduler.cc.o
[100%] Linking CXX shared library ../../lib/libns3.39-lte-default.so
[100%] Linking CXX shared library ../../lib/libns3.39-netanim-default.so
[100%] Linking CXX object scratch/CMakeFiles/scratch_TWONODE.dir/TWONODE.cc.o
[100%] Linking CXX executable ns3.39-TWONODE-default
AnimationInterface WARNING:Node:0 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:1 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:2 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:3 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:4 Does not have a mobility model. Use SetConstantPosition if it is stationary
AnimationInterface WARNING:Node:5 Does not have a mobility model. Use SetConstantPosition if it is stationary
At time +1s client sent 1024 bytes to 10.1.5.2 port 9
At time +1.01475s server received 1024 bytes from 10.1.1.1 port 49153
At time +1.01475s server sent 1024 bytes to 10.1.1.1 port 49153
At time +1.02949s client received 1024 bytes from 10.1.5.2 port 9
At time +2s client sent 1024 bytes to 10.1.5.2 port 9
At time +2.01475s server received 1024 bytes from 10.1.1.1 port 49153
At time +2.01475s server sent 1024 bytes to 10.1.1.1 port 49153
At time +2.02949s client received 1024 bytes from 10.1.5.2 port 9
At time +3s client sent 1024 bytes to 10.1.5.2 port 9
At time +3.01475s server received 1024 bytes from 10.1.1.1 port 49153
At time +3.01475s server sent 1024 bytes to 10.1.1.1 port 49153
At time +3.02949s client received 1024 bytes from 10.1.5.2 port 9
moune@LAPTOP-UICB0MJ3:~/ns-3-dev$ cd ~/netanim/build/bin
moune@LAPTOP-UICB0MJ3:~/netanim/build/bin$ ./netanim
```

OUTPUT



14 Implement and study the performance of a typical GSM network on NS-3 (using MAC layer).

The Global System for Mobile Communications (GSM) is a standard developed to describe protocols for second generation (2G) digital cellular networks. While NS-3 (Network Simulator 3) does not include a full GSM model, it can be adapted to simulate basic GSM-like behaviors at the MAC layer using CSMA or LTE primitives.

This project aims to implement a simplified GSM-like network using the CSMA protocol to emulate MAC-layer communication, and study the performance of such a system in terms of throughput, latency, and packet delivery.

NS-3

NS-3 is a discrete-event network simulator, popular in academic and research settings. It supports network models for internet stacks, routing, WiFi, LTE, and more.

GSM

GSM architecture consists of mobile devices, base transceiver stations (BTS), and a core network. The MAC layer in GSM handles time slot scheduling, retransmissions, and control signaling.

Simulation Design

Topology

- 1 Base Station (BTS)
- 3 User Equipment (UE) nodes
- CSMA channel for communication

Assumptions

- GSM behavior is emulated via CSMA MAC channel
- UDP echo application simulates voice or message transfer
- Basic metrics like delay and throughput are analyzed

Implementation in C++ (NS-3)

Simulation Code

```

1 #include "ns3/core-module.h"
2 #include "ns3/network-module.h"
3 #include "ns3/csma-module.h"
4 #include "ns3/internet-module.h"
5 #include "ns3/applications-module.h"
6 #include "ns3/netanim-module.h"
7 using namespace ns3;
8
9 NS_LOG_COMPONENT_DEFINE("GsmMacSim");
10
11 int main(int argc, char *argv[]) {
12     NodeContainer nodes;
13     nodes.Create(4);
14
15     CsmaHelper csma;
16     csma.SetChannelAttribute("DataRate", StringValue("1Mbps"));
17     csma.SetChannelAttribute("Delay", TimeValue(NanoSeconds(6560)));
18
19     NetDeviceContainer devices = csma.Install(nodes);
20
21     InternetStackHelper stack;
22     stack.Install(nodes);
23
24     Ipv4AddressHelper address;
25     address.SetBase("10.1.1.0", "255.255.255.0");
26
27     Ipv4InterfaceContainer interfaces = address.Assign(devices);
28
29     UdpEchoServerHelper echoServer(9);
30     ApplicationContainer serverApps = echoServer.Install(nodes.
31         Get(0));
32     serverApps.Start(Seconds(1.0));
33     serverApps.Stop(Seconds(10.0));
34
35     for (int i = 1; i < 4; ++i) {

```

```
35     UdpEchoClientHelper echoClient(interfaces.GetAddress(0),
36                                     9);
37     echoClient.SetAttribute("MaxPackets", UintegerValue(3));
38     echoClient.SetAttribute("Interval", TimeValue(Seconds
39                               (1.0)));
40     echoClient.SetAttribute("PacketSize", UintegerValue(1024)
41                               );
42
43 }
44
45 csma.EnablePcapAll("gsm-mac");
46 AnimationInterface anim("gm.xml");
47 Simulator::Run();
48 Simulator::Destroy();
49 return0      ;
50 }
```

Listing 6: Simplified GSM MAC Simulation

OUTPUT

