K.S.R COLLEGE OF ENGINEERING

(AUTONOMOUS)

THOKKAVADI POST, TIRUCHENGODE-637215

NAMAKKAL DISTRICT, TAMILNADU

RECORD NOTE BOOK

REGISTER NO:

Certified that this is the bonafide record of work done by Selvan / Selvi _____ of the _____Semester _____branch during the year_____in the _____laboratory.

Staff-In-Charge Head of The Department

Submitted for the University practical Examination on_____

Internal examiner External Examiner

K.S.R. COLLEGE OF ENGINEERING (Autonomous) –TIRUCHENGODE.

Vision of the Institution

- We envision to achieve status as an excellent educational institution in the global knowledge hub, making self-learners, experts, ethical and responsible engineers, technologists, scientists, managers, administrators and entrepreneurs who will significantly contribute to research and environment friendly sustainable growth of the nation and the world.

Mission of the Institution

- To inculcate in the students self-learning abilities that enable them to become competitive and considerate engineers, technologists, scientists, managers, administrators and entrepreneurs by diligently imparting the best of education, nurturing environmental and social needs.
- To foster and maintain a mutually beneficial partnership with global industries and Institutions through knowledge sharing, collaborative research and innovation.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision of the Department

- To create ever green professionals for software industry, academicians for knowledge cultivation and researchers for contemporary society modernization**.**

Mission of the Department

- To produce proficient design, code and system engineers for software development.
- To keep updated contemporary technology and fore coming challenges for welfare of the society.

Programme Educational Objectives (PEOs)

PEO1: Figure out, formulate, analyse typical problems and develop effective solutions by imparting the idea and principles of science, mathematics, engineering fundamentals and computing.

PEO2: Competent professionally and successful in their chosen career through life-long learning.

PEO3: Excel individually or as member of a team in carrying out projects and exhibit social needs and follow professional ethics

A.	Program Outcomes (POs)Engineering Graduates will be able to :
PO1	**Engineering knowledge:** Ability to exhibit the knowledge of mathematics, science, engineering fundamentals and programming skills to solve problems in computer science.
PO2	**Problem analysis:** Talent to identify, formulate, analyse and solve complex engineering problems with the knowledge of computer science. .
PO3	**Design/development of solutions:** Capability to design, implement, and evaluate a computer based system, process, component or program to meet desired needs.
PO4	**Conduct investigations of complex problems:** Potential to conduct investigation of complex problems by methods that include appropriate experiments, analysis and synthesis of information in order to reach valid conclusions.
PO5	**Modern tool Usage:** Ability to create, select, and apply appropriate techniques, resources and modern engineering tools to solve complex engineering problems.
PO6	**The engineer and society:** Skill to acquire the broad education necessary to understand the impact of engineering solutions on a global economic, environmental, social, political, ethical, health and safety.
PO7	**Environmental and sustainability:** Ability to understand the impact of the professional engineering solutions in societal and Environmental contexts and demonstrate the knowledge of, and need for sustainable development.
PO8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibility and norms of the engineering practices.
PO9	**Individual and team work:** Ability to function individually as well as on multi-disciplinary teams.
PO10	**Communication:** Ability to communicate effectively in both verbal and written mode to excel in the career.
PO11	**Project management and finance:** Ability to integrate the knowledge of engineering and management principles to work as a member and leader in a team on diverse projects.
PO12	**Life-long learning:** Ability to recognize the need of technological change by independent and life-long learning.
B. Program Specific Outcomes (PSOs)	
PSO1	Develop and Implement computer solutions that accomplish goals to the industry, government or research by exploring new technologies.
PSO2	Grow intellectually and professionally in the chosen field

K.S.R. COLLEGE OF ENGINEERING (Autonomous)					
SEMESTER – V					
18CS523	**COMPUTER NETWORKS LABORATORY**	L	T	P	C
		0	0	3	1

Prerequisite: *Basic knowledge of java Programming.*

Objectives:

- *To acquire practical exposure on basic networking concepts.*
- *To gain experience on error control and data link layer protocols.*
- *To identify the functions of various routing algorithms.*
- *To comprehend the role of cryptography techniques and communication using UDP and TCP.*

List of Experiments:

1. Study of Network topology configuration and Network Devices in detail.
2. Connect the computers in Local Area Network.
3. Simulation of error detecting code using CRC.
4. Simulation of Stop and wait protocol.
5. Simulation of Go back-N and selective repeat protocols.
6. Simulation of Distance vector routing algorithm.
7. Simulation of Link state routing algorithm.
8. Apply Caesar cipher security algorithm for network security.
9. Apply TCP program for date/time server.
10. Simple UDP socket program for echo server client chat.
11. Develop a program for congestion control using Leaky bucket algorithm.
12. Study the simulation of Network Simulator.

Total: 45 Periods

Course Outcomes: On Completion of this course, the student will be able to

CO1: Demonstrate the various network topologies.
CO2: Implement the performance of error control and data link layer protocols.
CO3: Create and analyse the routing algorithms and congestion control mechanism.
CO4: Apply TCP and UDP to Infer network security and communication.
CO5: Be aware of the simulation of Network simulator.

	18CS523 - COMPUTER NETWORKS LABORATORY													
CO	Course Outcomes	Programme Outcomes												
		PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
C405.1	Demonstrate the various network topologies.	2	2	3	3	3	-	1	-	-	3	2	-	3	3
C405.2	Implement the performance of error control and data link layer protocols.	2	3	3	3	2	-	1	-	-	3	2	-	3	3
C405.3	Create and analyze the routing algorithms and congestion control mechanism.	2	2	3	3	3	-	1	-	-	3	2	-	3	3
C405.4	Apply TCP and UDP to Infer network security and communication	2	2	3	3	3	-	1	-	-	3	2	-	3	3
C405.5	Be aware of the simulation of Network simulator.	2	2	3	3	3	-	1	-	-	3	2	-	3	3
Maximum Appeared value		2	2	3	3	3	-	1	-	-	3	2	-	3	3

LIST OF EXPERIMENTS

EX.NO	DATE	NAME OF EXPERIMENT	PAGE NO	MARKS	STAFF'S SIGNATURE
1		STUDY OF NETWORK TOPOLOGY CONFIGURATION AND NETWORK DEVICES IN DETAIL.			
2		CONNECT THE COMPUTERS IN LOCAL AREA NETWORK			
3		SIMULATION OF ERROR DETECTING CODE USING CRC.			
4		SIMULATION OF STOP AND WAIT PROTOCOL			
5.A		SIMULATION OF GO BACK AND N-PROTOCOL			
5.B		SIMULATION OF SELECTIVE REPEAT PROTOCOL			
6		SIMULATION OF DISTANCE VECTOR ROUTING ALGORITHM.			
7		SIMULATION OF LINK STATE ROUTING ALGORITHM.			
8		APPLY CAESAR CIPHER SECURITY ALGORITHM FOR NETWORK SECURITY.			
9		APPLY TCP PROGRAM FOR DATE/TIME SERVER.			
10		SIMPLE UDP SOCKET PROGRAM FOR ECHO SERVER CHAT.			
11		DEVELOP A PROGRAM FOR CONGESTION CONTROL USING LEAKY BUCKET ALGORITHM			
12		STUDY THE SIMULATION OF NETWORK SIMULATOR			
		AVERAGE:			

EX.NO: 1	STUDY OF NETWORK TOPOLOGY CONFIGURATION AND NETWORK DEVICES IN DETAIL.
DATE:	

AIM:

To study of Network Topology configuration and Network Devices in detail.

PROCEDURE:

NETWORK TOPOLOGY

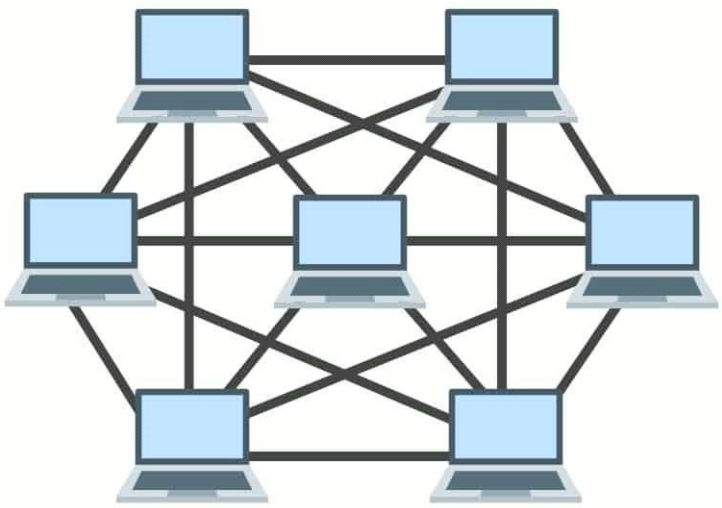
Network topology is the arrangement of the elements of a communication network. Network topology can be used to define or describe the arrangement of various types of telecommunication networks, including command and control radio networks, industrial field busses and computer networks.

Geometric representation of how the computers is connected to each other is known as topology.

There are five types of topology in computer networks:

1. Mesh Topology
2. Star Topology
3. Bus Topology
4. Ring Topology
5. Hybrid Topology

Mesh Topology



In mesh topology each device is connected to every other device on the network through a dedicated point-to-point link. When we say dedicated, it means that the link only carries data for the two connected devices only. Let's say we have n devices in the network then each device must be connected with (n-1) devices of the network. Number of links in a mesh topology of n devices would be n(n-1)/2.

Advantages of Mesh topology

1. No data traffic issues as there is a dedicated link between two devices which means the link is only available for those two devices.

2. Mesh topology is reliable and robust as failure of one link doesn't affect other links and the communication between other devices on the network.

3. Mesh topology is secure because there is a point-to-point link thus unauthorized access is not possible.

4. Fault detection is easy.

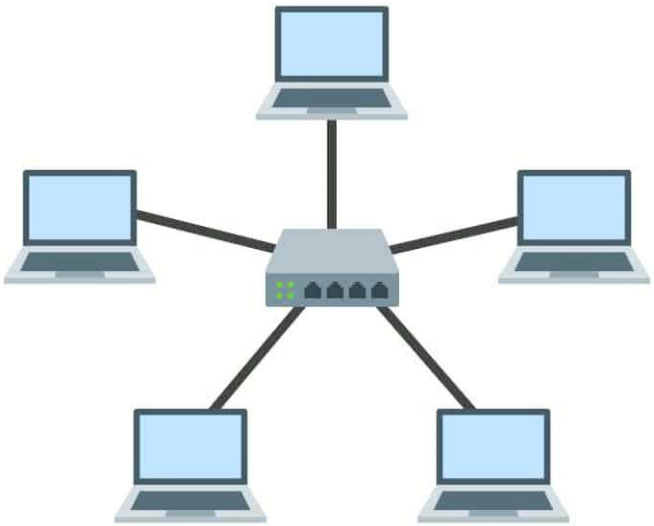
Disadvantages of Mesh topology

1. Number of wires required to connected each system is tedious and headache.

2. Since each device needs to be connected with other devices, number of I/O ports required must be huge.

3. Scalability issues because a device cannot be connected with large number of devices with a dedicated point to point link.

Star Topology



In star topology each device in the network is connected to a central device called hub. Unlike Mesh topology, star topology doesn't allow direct communication between devices, a device must have to communicate through hub. If one device wants to send

data to other device, it has to first send the data to hub and then the hub transmits that data to the designated device.

Advantages of Star topology

1. Less expensive because each device only needs one I/O port and needs to be connected with hub with one link.

2. Easier to install

3. Less number of cables required because each device needs to be connected with the hub only.

4. Robust, if one link fails, other links will work just fine.

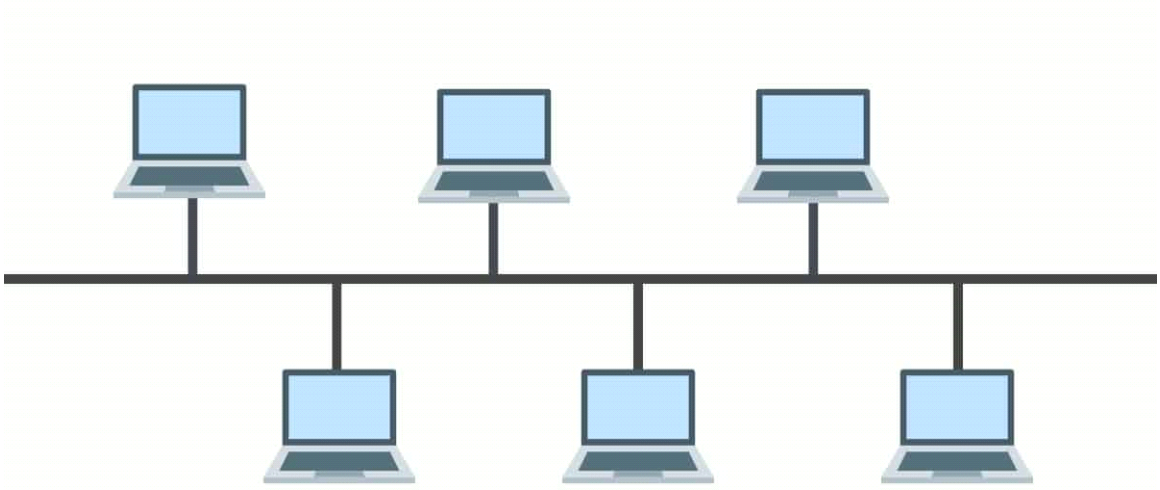
5. Easy fault detection because the link can be easily identified.

Disadvantages of Star topology

1. If hub goes down everything goes down, none of the devices can work without hub.

2. Hub requires more resources and regular maintenance because it is the central system of star topology.

Bus Topology



In bus topology there is a main cable and all the devices are connected to this main cable through drop lines. There is a device called tap that connects the drop line to the main cable. Since all the data is transmitted over the main cable, there is a limit of drop lines and the distance a main cable can have.

Advantages of bus topology

1. Easy installation, each cable needs to be connected with backbone cable.

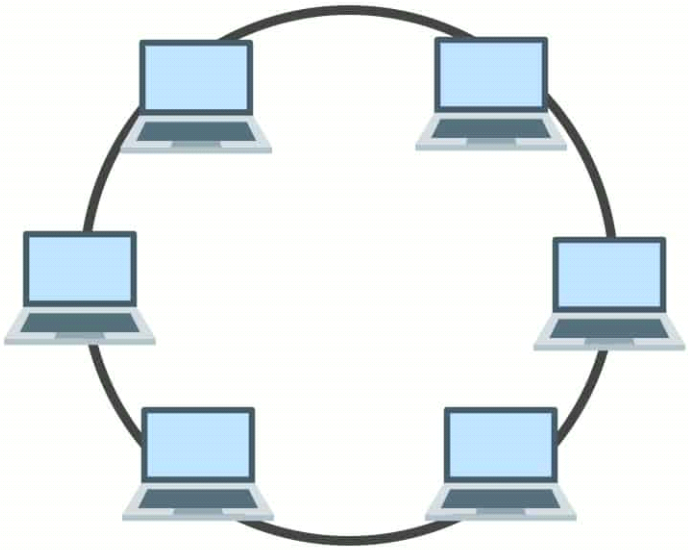
2. Less cables required than Mesh and star topology

Disadvantages of bus topology

1. Difficultly in fault detection.

2. Not scalable as there is a limit of how many nodes you can connect with backbone cable.

Ring Topology



In ring topology each device is connected with the two devices on either side of it. There are two dedicated point to point links a device has with the devices on the either side of it. This structure forms a ring thus it is known as ring topology. If a device wants to send data to another device, then it sends the data in one direction, each device in ring topology has a repeater, if the received data is intended for other device, then repeater forwards this data until the intended device receives it.

Advantages of Ring Topology

1. Easy to install.

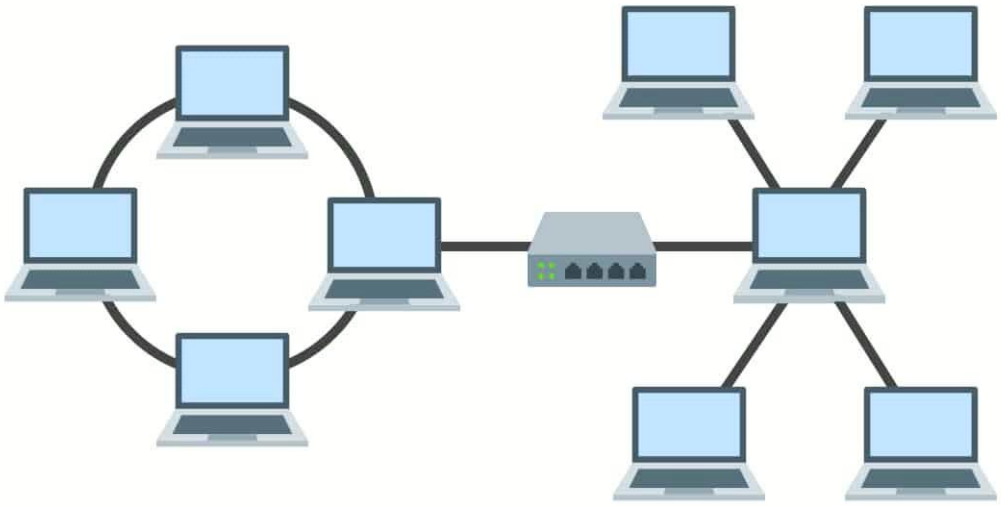
2. Managing is easier as to add or remove a device from the topology only two links are required to be changed.

Disadvantages of Ring Topology

1. A link failure can fail the entire network as the signal will not travel forward due to failure.

2. Data traffic issues, since all the data is circulating in a ring.

Hybrid topology



A combination of two or more topology is known as hybrid topology. For example, a combination of star and mesh topology is known as hybrid topology.

Advantages of Hybrid topology

1. We can choose the topology based on the requirement for example, scalability is our concern then we can use star topology instead of bus technology.

2. Scalable as we can further connect other computer networks with the existing networks with different topologies.

Disadvantages of Hybrid topology

1. Fault detection is difficult.

2. Installation is difficult.

3. Design is complex so maintenance is high thus expensive

NETWORK DEVICES

1. Repeater – A repeater operates at the physical layer. Its job is to regenerate the signal over the same network before the signal becomes too weak or corrupted so as to extend the length to which the signal can be transmitted over the same network. An important point to be noted about repeaters is that they do not amplify the signal. When the signal becomes weak, they copy the signal bit by bit and regenerate it at the original strength. It is a 2-port device.

2. Hub – A hub is basically a multiport repeater. A hub connects multiple wires coming from different branches, for example, the connector in star topology which connects different stations. Hubs cannot filter data, so data packets are sent to all connected devices. In other words, the collision domain of all hosts connected through Hub remains one. Also, they do not have the intelligence to find out the best path for data packets which leads to inefficiencies and wastage.

Types of Hubs:

Active Hub: - These are the hubs that have their own power supply and can clean, boost, and relay the signal along with the network. It serves both as a repeater as well as a wiring center. These are used to extend the maximum distance between nodes.

Passive Hub: - These are the hubs that collect wiring from nodes and power supply from the active hub. These hubs relay signals onto the network without cleaning and boosting them and can't be used to extend the distance between nodes.

Intelligent Hub: - It works like active hubs and includes remote management capabilities. They also provide flexible data rates to network devices. It also enables an administrator to monitor the traffic passing through the hub and to configure each port in the hub.

3. Bridge – A bridge operates at the data link layer. A bridge is a repeater, with add on the functionality of filtering content by reading the MAC addresses of source and destination. It is also used for interconnecting two LANs working on the same protocol. It has a single input and single output port, thus making it a 2-port device.

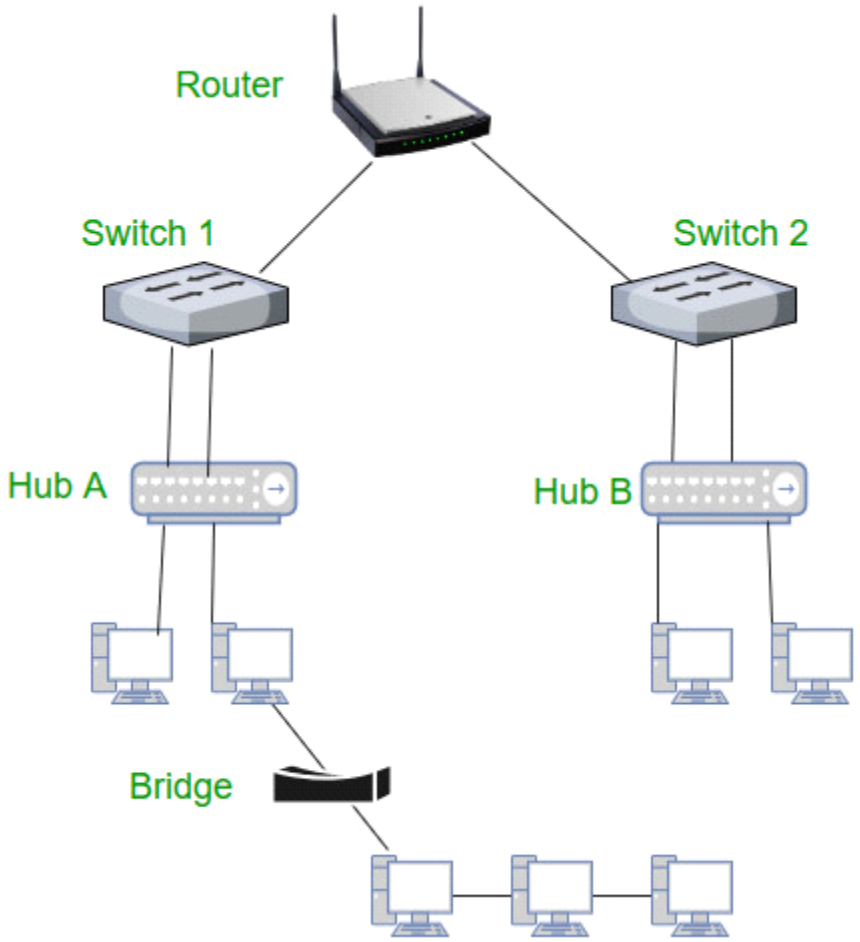
Types of Bridges:

Transparent Bridges: - These are the bridge in which the stations are completely unaware of the bridge's existence i.e., whether or not a bridge is added or deleted from the network, reconfiguration of the stations is unnecessary. These bridges make use of two processes i.e., bridge forwarding and bridge learning.

Source Routing Bridges: - In these bridges, routing operation is performed by the source station and the frame specifies which route to follow. The host can discover the frame by sending a special frame called the discovery frame, which spreads through the entire network using all possible paths to the destination.

4. Switch – A switch is a multiport bridge with a buffer and a design that can boost its efficiency (a large number of ports imply less traffic) and performance. A switch is a data link layer device. The switch can perform error checking before forwarding data, which makes it very efficient as it does not forward packets that have errors and forward good packets selectively to the correct port only. In other words, the switch divides the collision domain of hosts, but broadcast domain remains the same.

5. Routers – A router is a device like a switch that routes data packets based on their IP addresses. The router is mainly a Network Layer device. Routers normally connect LANs and WANs together and have a dynamically updating routing table based on which they make decisions on routing the data packets. Router divide broadcast domains of hosts connected through it.



6. Gateway – A gateway, as the name suggests, is a passage to connect two networks together that may work upon different networking models. They basically work as the messenger agents that take data from one system, interpret it, and transfer it to another system. Gateways are also called protocol converters and can operate at any network layer. Gateways are generally more complex than switches or routers. Gateway is also called a protocol converter.

7. Brouter – It is also known as the bridging router is a device that combines features of both bridge and router. It can work either at the data link layer or a network layer. Working as a router, it is capable of routing packets across networks, and working as the bridge, it is capable of filtering local area network traffic.

8. NIC – NIC or network interface card is a network adapter that is used to connect the computer to the network. It is installed in the computer to establish a LAN. It has a unique id that is written on the chip, and it has a connector to connect the cable to it. The cable acts as an interface between the computer and router or modem. NIC card is a layer 2 device which means that it works on both physical and data link layer of the network model.

RESULT:

The Network Topology configuration and Network devices has been studied successfully.

EX.NO: 2	CONNECT THE COMPUTERS IN LOCAL AREA NETWORK
DATE:	

AIM:

To create scenario and study the performance of Ethernet LAN protocol.
a. Study of Number of Transmitting Nodes vs. Throughput (%) (NetSim).

b. Real time file transfer and study of application and network performance. Compare throughput (%) for 1 Client 1 Server and for 2 Clients 1 Server. (LAN Trainer)

Hardware / Software Requirements
1. NetSim Academic Version simulation software (for simulation)
2. LAN trainer (Model LT 01) (for real time)
3. LT Soft – Operational software for LAN Trainer (for real time)

Introduction
Ethernet is a LAN (Local area Network) protocol operating at the MAC (Medium AccessControl) layer. Ethernet has been standardized as per IEEE 802.3. The underlying protocol in Ethernet is known as the CSMA / CD – Carrier Sense Multiple Access / Collision Detection. The working of the Ethernet protocol is as explained below,

- A node which has data to transmit senses the channel,
- If the channel is idle then, the data is transmitted.
- If the channel is busy then, the station defers transmission until the channel is sensed to be idle and then immediately transmitted.
- If more than one node starts data transmission at the same time, the data collides. This collision is heard by the transmitting nodes which enter into contention phase.
- The contending nodes resolve contention using an algorithm called truncated binary exponential back off.

Performance Metrics
Some of the important performance metrics which are recorded during simulation are given below.

Throughput (%)
Fraction of link's capacity devoted to carrying frames.
Formula:Throughput (%) = (((Pay Load Delivered + Overheads) * Byte Time) / Simulation
End Time) * 100.0.

Normalized Throughput (%) Also called as Goodput (%)
Fraction of link's capacity devoted to carrying non-retransmitted frames excluding bytes due to protocol overhead, collision and retransmission.
Formula:Normalized Throughput (%) = (((Pay Load Delivered) * Byte Time) / SimulationEnd Time) * 100.0.

Mean delay (Micro Second/Frame)
Mean time a frame waits at a station before being successfully transmitted (queuing time and medium access time) and the transmission time per frame.

Formula:**Mean Delay** (Micro Second/Frame) = (Queuing Time + Medium Access Time +Transmission Time) / (Pay Load Delivered / Maximum Data Size per Frame).

Response Time (Micro Second / Frame) Also called as Service Time
Sum of medium access time and transmission time per frame.

Formula:
Response Time (MMicro Second/Frame) = (Medium Access Time + TransmissionTime) / (Pay Load Delivered / Maximum Data Size per Frame).

Collision count.
Total number of collisions in the network.
The definitions of all parameters / variables are available in NetSim.

Description
To begin with the experiment, run LAN Trainer.The systems required for file transmission should have LAN connection. Move the cursor over the icon to check for LAN connection as shown above. If the required system is not connected then go to Start Menu>>Right click at My Network places>>Select Properties>>Click the icon to Enable LAN.Select BUS topology to perform this experiment. For Bus topology, click on in the file panel. Connect the cable from the system to the required topology in the kit. Turn on the LT Kit by turning the manual switch to the Bus topology.User should select one system/computer as the Server and one system as the Client. Transfer the file between the two systems through the LAN Trainer and save the received metrics. **At the Server**, 1500 as value for Packet Size (Bytes) and click Send. This is the standard size of Ethernet Payload.

At the Client:
• Enter the Server IP Address.
• Request for File: Enter the location where the file exists in the Server system (Mention the name of the file along with the path. Eg: C:\test.txt)
• Save in (Path): Click browse and select the file path where you want to save (Do not give a file name. The file will be stored in the original name as it was in the server)
• Inter Packet Gap (ms): Select 0 (Zero)

• Select the correct adapter and click Receive. (If an incorrect adapter is select, an error message will be shown)

Results 2.b

Client-Server	Throughput
1 Server 1 Client	
1 Server 2 Client	

First perform the experiment when one pair of system transmits and receives. Then perform the experiment with two pairs of client-servers simultaneously sending and receiving files.

Inference

The number of collisions when 1 client 1 server operates is zero. When there are multiple pairs of client-servers operating increase in collisions are observed. This is because the file transfers occur in a shared medium. The number of collisions depends on the size of the files being transferred. This directly decreases throughput (%) of the network.

Result:

Thus, the performance of Ethernet LAN protocol is studied.

EX.NO: 03	SIMULATION OF ERROR DETECTING CODE USING CRC.
DATE:	

Aim:
To simulate error detecting code using CRC.

ALGORITHM:
Step 1: Start
Step 2: Open the editor and type the program for error detection
Step 3: Get the input in the form of bits.
Step 4: Append the redundancy bits.
Step 5: Divide the appended data using a divisor polynomial.
Step 6: The resulting data should be transmitted to the receiver.
Step 7: At the receiver the received data is entered.
Step 8: The same process is repeated at the receiver.
Step 9: If the remainder is zero there is no error otherwise there is some error in the received bits.
Step 10: Run the program.
Step 11: Stop

Program:

```
import java.util.*;
class CRC {
public static void main(String args[]) {
Scanner scan = new Scanner(System.in);
int n;
//Accept the input
System.out.println("Enter the size of the data:");
n = scan.nextInt();
int data[] = new int[n];
System.out.println("Enter the data, bit by bit:");

for(int i=0 ; i < n ; i++) {
System.out.println("Enter bit number " + (n-i) + ":");
data[i] = scan.nextInt();
}
// Accept the divisor
System.out.println("Enter the size of the divisor:");
n = scan.nextInt();
int divisor[] = new int[n];
System.out.println("Enter the divisor, bit by bit:");
for(int i=0 ; i < n ; i++) {
```

```
System.out.println("Enter bit number " + (n-i) + ":");
divisor[i] = scan.nextInt();
}
// Divide the inputted data by the inputted divisor
// Store the remainder that is returned by the method
int remainder[] = divide(data, divisor);
for(int i=0 ; i < remainder.length-1 ; i++) {
System.out.print(remainder[i]);
}
System.out.println("\nThe CRC code generated is:");
for(int i=0 ; i < data.length ; i++) {
System.out.print(data[i]);
}
for(int i=0 ; i < remainder.length-1 ; i++) {
System.out.print(remainder[i]);

}
System.out.println();
// Create a new array
// It will have the remainder generated by the above method appended
// to the inputted data
int sent_data[] = new int[data.length + remainder.length - 1];
System.out.println("Enter the data to be sent:");
for(int i=0 ; i < sent_data.length ; i++) {
System.out.println("Enter bit number " + (sent_data.length-i)
+ ":");
sent_data[i] = scan.nextInt();
}
receive(sent_data, divisor);
}
static int[] divide(int old_data[], int divisor[]) {
int remainder[] , i;
int data[] = new int[old_data.length + divisor.length];
System.arraycopy(old_data, 0, data, 0, old_data.length);
remainder = new int[divisor.length];
System.arraycopy(data, 0, remainder, 0, divisor.length);
for(i=0 ; i < old_data.length ; i++) {
System.out.println((i+1) + ".) First data bit is : "
+ remainder[0]);
System.out.print("Remainder : ");
if(remainder[0] == 1) {

for(int j=1 ; j < divisor.length ; j++) {
remainder[j-1] = exor(remainder[j], divisor[j]);
System.out.print(remainder[j-1]);
}
```

```
}
else {
for(int j=1 ; j < divisor.length ; j++) {
remainder[j-1] = exor(remainder[j], 0);
System.out.print(remainder[j-1]);
}
}
remainder[divisor.length-1] = data[i+divisor.length];
System.out.println(remainder[divisor.length-1]);
}
return remainder;
}
static int exor(int a, int b) {
if(a == b) {
return 0;
}
return 1;
}
static void receive(int data[], int divisor[]) {

nt remainder[] = divide(data, divisor);
for(int i=0 ; i < remainder.length ; i++) {
if(remainder[i] != 0) {
System.out.println("There is an error in received data...");
return;
}
}
//Otherwise there is no error in the received data
System.out.println("Data was received without any error.");
}
}
```

Output:

```
Enter the size of the data:
7
Enter the data, bit by bit:
Enter bit number 7:
1
Enter bit number 6:
0
Enter bit number 5:
0
Enter bit number 4:
1
```

```
Enter bit number 3:
1
Enter bit number 2:
0
Enter bit number 1:
1
Enter the size of the divisor:
4
Enter the divisor, bit by bit:
Enter bit number 4:
1
Enter bit number 3:
0

Enter bit number 2:
1
Enter bit number 1:
1
1.) First data bit is : 1
Remainder : 0101
2.) First data bit is : 0
Remainder : 1010
3.) First data bit is : 1
Remainder : 0011
4.) First data bit is : 0
Remainder : 0110
5.) First data bit is : 0
Remainder : 1100
6.) First data bit is : 1
Remainder : 1110
7.) First data bit is : 1
Remainder : 1010
101
The CRC code generated is:
1001101101
Enter the data to be sent:
Enter bit number 10:
1
Enter bit number 9:
0
Enter bit number 8:
0
Enter bit number 7:
1
Enter bit number 6:
```

1
Enter bit number 5:
0
Enter bit number 4:
1
Enter bit number 3:
1
Enter bit number 2:
0
Enter bit number 1:
1
1.) First data bit is : 1
Remainder : 0101
2.) First data bit is : 0
Remainder : 1010
3.) First data bit is : 1
Remainder : 0011
4.) First data bit is : 0
Remainder : 0111

5.) First data bit is : 0
Remainder : 1110
6.) First data bit is : 1
Remainder : 1011
7.) First data bit is : 1
Remainder : 0000
8.) First data bit is : 0
Remainder : 0000
9.) First data bit is : 0
Remainder : 0000
10.) First data bit is : 0
Remainder : 0000
Data was received without any error.

RESULT:

Thus, the program is executed successfully and the output is verified.

EX.NO: 04	SIMULATION OF STOP AND WAIT PROTOCOL
DATE:	

AIM

To implement the stop and wait protocol using java programming language.

SENDER-SIDE ALGORITHM:

Step 1: Start
Step 2: sequence ß 0 .
Step 3: Accept new packet and assign sequence to it.
Step 4: Send packet sequence with sequence number sequence.
Step 5: Set timer for recently sent packets.
Step 6: If error free acknowledgment from receiver and NextFrameExpected -> sequence then sequenceß NextFrameExpected.
Step 7: If time out then go to step3.
Step 8: Stop.

RECEIVER-SIDE ALGORITHM:

Step 1: Start.
Step 2: NextFrameExpectedß 0, repeat steps 3 forever.
Step 3: If error-free frame received and sequence= NextFrameExpected, then pass packet to higher layer and NextFrameExpectedßNextFrameExpected+1(modulo 2).
Step 4: Stop.

SERVER-SIDE CODE:

```
package protocal;

import java.io.*;
import java.net.*;
import java.util.Scanner;
class stopwaitsender
{
 public static void main(String args[]) throws Exception
 {
  stopwaitsender sws = new stopwaitsender();
  sws.run();
 }
 public void run() throws Exception
 {
  Scanner sc=new Scanner(System.in);
```

```
  System.out.println("Enter no of frames to be sent:");
  int n=sc.nextInt();
  Socket myskt=new Socket("localhost",9999);
  PrintStream myps=new PrintStream(myskt.getOutputStream());
  for(int i=0;i<=n;)
  {
   if(i==n)
   {
    myps.println("exit");
        break;
   }
   System.out.println("Frame no "+i+" is sent");
   myps.println(i);
   BufferedReader bf=new BufferedReader(new
InputStreamReader(myskt.getInputStream()));
         String ack=bf.readLine();
   if(ack!=null)
   {
    System.out.println("Acknowledgement was Received from receiver");
        i++;
        Thread.sleep(4000);
   }
   else
   {
     myps.println(i);
   }
  }
 }
}
```

OUTPUT:

```
Acknowledgement was Received from receiver
Frame no 2 is sent
Acknowledgement was Received from receiver
Frame no 3 is sent
Acknowledgement was Received from receiver
Frame no 4 is sent
Acknowledgement was Received from receiver
Frame no 5 is sent
Acknowledgement was Received from receiver
Frame no 6 is sent
Acknowledgement was Received from receiver
Frame no 7 is sent
Acknowledgement was Received from receiver
```

RECEIVER SIDE PROGRAM:

```
package protocal;

import java.io.*;
import java.net.*;
class stopwaitreceive
{
 public static void main(String args[])throws Exception
 {
  stopwaitreceive swr = new stopwaitreceive();
  swr.run();
 }
  public void run() throws Exception
 {
  String temp="any message",str="exit";
  ServerSocket myss=new ServerSocket(9999);
  Socket ss_accept=myss.accept();
  BufferedReader ss_bf=new BufferedReader(new
InputStreamReader(ss_accept.getInputStream()));

  PrintStream myps=new PrintStream(ss_accept.getOutputStream());
  while(temp.compareTo(str)!=0)
  {
   Thread.sleep(1000);
   temp=ss_bf.readLine();
   if(temp.compareTo(str)==0)
   { break;}
   System.out.println("Frame "+temp+" was received");
   Thread.sleep(500);
   myps.println("Received");
  }
   System.out.println("ALL FRAMES  WERE RECEIVED SUCCESSFULLY");
 }
}
```

OUTPUT:

Frame 0 was received
Frame 1 was received
Frame 2 was received
Frame 3 was received
Frame 4 was received
Frame 5 was received
Frame 6 was received
Frame 7 was received
ALL FRAMES WERE RECEIVED SUCCESSFULLY

RESULT:
Thus, the program is executed successfully and the output is verified.

EX.NO: 5.A	SIMULATION OF GO BACK AND N- PROTOCOL
DATE:	

AIM:

To implement Go back and N Protocol programming language.

ALGORITHM:

Step 1: Initialize server socket

Step 2: Display waiting for connection

Step 3: Initialize the socket and accept the client message

Step 4: Display connected with client

Step 5: Initialize i/p stream

Step 6: Read message length

Step 7: Display message length

Step 8: Read message from error occurred bit

Step 9: Display message received from error bit

Step 10: Close all objects

Step 11: Stop

CODE:

```
import java.io.*;
public class GoBackN {
public static void main(String args[]) throws IOException {
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
System.out.println("Please enter the Window Size: ");
int window = Integer.parseInt(br.readLine());
 boolean loop = true;
 int sent = 0;
while(loop) {
for(int i = 0; i < window; i++) {
```

```
System.out.println("Frame " + sent + " has been transmitted.");
 sent++;
 if(sent == window) break;
 }
System.out.println("Please enter the last Acknowledgement received.");
int ack = Integer.parseInt(br.readLine());
 if(ack == window) loop = false;
else sent = ack;
 }
 }
}
```

OUTPUT:

```
Please enter the Window Size:
7
Frame 0 has been transmitted.
Frame 1 has been transmitted.
Frame 2 has been transmitted.
Frame 3 has been transmitted.
Frame 4 has been transmitted.
Frame 5 has been transmitted.
Frame 6 has been transmitted.
Please enter the last Acknowledgement received.
3
Frame 3 has been transmitted.
 Frame 4 has been transmitted.
Frame 5 has been transmitted.
Frame 6 has been transmitted.
```

Please enter the last Acknowledgement received.

5

Frame 5 has been transmitted.

Frame 6 has been transmitted.

Please enter the last Acknowledgement received.

7

RESULT:

Thus, the program is executed successfully and the output is verified.

EX.NO: 5. B	SIMULATION OF SELECTIVE REPEAT PROTOCOL
DATE:	

AIM:

To implement Go back and N Protocol programming language.

SERVER-SIDE ALGORITHM:

Step 1: Initialize server socket

Step 2: Display waiting for connection

Step 3: Initialize the socket and accept the client message

Step 4: Display connected with client

Step 5: Initialize i/p stream

Step 6: Read message length

Step 7: Display message length

Step 8: Read message from error occurred bit

Step 9: Display message received from error bit

Step 10: Close all objects

Step 11: Stop

CLIENT-SIDE ALGORITHM:

Step 1: Open socket with input address, port

Step 2: Display the message server connected

Step 3: Initialize o/p stream

Step 4: Write message

Step 5: assign transmission message and error bit

Step 6: Convert string to character array for bit by bit transmission

Step 7: Write message

Step 8: Display the message Retransmitting message from error bit

Step 9: Check the condition

Step 10: Display the array

Step 11: Close all objects

Step 12: Stop

CLIENT-SIDE CODE:

```
package selective;
import java.lang.System;
import java.net.ConnectException;
import java.net.InetAddress;
import java.net.Socket;
import java.net.SocketException;
import java.io.*;
import java.text.*;
import java.util.Random;
import java.util.*;
public class client {
static Socket connection;
public static void main(String a[]) throws SocketException {
try {
int v[] = new int[10];
int n = 0;
Random rands = new Random();
int rand = 0;
InetAddress addr = InetAddress.getByName("Localhost");
System.out.println(addr);
connection = new Socket(addr, 8011);
DataOutputStream out = new DataOutputStream(
connection.getOutputStream());
DataInputStream in = new DataInputStream(
connection.getInputStream());
```

```
int p = in.read();
System.out.println("No of frame is:" + p);
for (int i = 0; i < p; i++) {
v[i] = in.read();
System.out.println(v[i]);
//g[i] = v[i];
}
rand = rands.nextInt(p);//FRAME NO. IS RANDOMLY GENERATED
v[rand] = -1;
for (int i = 0; i < p; i++)
 {
System.out.println("Received frame is: " + v[i]);
}
for (int i = 0; i < p; i++)
if (v[i] == -1) {
System.out.println("Request to retransmit from"+ (i+1) + " again!!");
n = i;
out.write(n);
out.flush();
}
System.out.println();
v[n] = in.read();
System.out.println("Received frame is: " + v[n]);
System.out.println("quiting");
} catch (Exception e) {
System.out.println(e);
}
}
}
```

OUTPUT:

```
Localhost/127.0.0.1
No of frame is:8
30
40
50
60
70
80
90
100
Received frame is: 30
Received frame is: 40
Received frame is: 50
Received frame is: 60
Received frame is: -1
Received frame is: 80
Received frame is: 90
Received frame is: 100
Request to retransmit from5 again!!

Received frame is: 70
quiting.
```

SERVER-SIDE CODE:

```
package selective;


import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;
import java.net.SocketException;
public class sender
{
static ServerSocket Serversocket;
static DataInputStream dis;
static DataOutputStream dos;
public static void main(String[] args) throws SocketException
{
try
{
int a[] = { 30, 40, 50, 60, 70, 80, 90, 100 };
Serversocket = new ServerSocket(8011);
System.out.println("waiting for connection");
Socket client = Serversocket.accept();
dis = new DataInputStream(client.getInputStream());
dos = new DataOutputStream(client.getOutputStream());
System.out.println("The number of packets sent is:" + a.length);
int y = a.length;
dos.write(y);
dos.flush();
for (int i = 0; i < a.length; i++)
```

```
{
dos.write(a[i]);
dos.flush();
}
int k = dis.read();
dos.write(a[k]);
dos.flush();
}
catch (IOException e)
{
System.out.println(e);
}
finally
{
try
{
dis.close();
dos.close();
}
catch (IOException e)
{
e.printStackTrace();
}
}
}
}
```

OUTPUT:

waiting for connection
The number of packets sent is: 8

RESULT:

Thus, the program is executed successfully and the output is verified.

EX.NO: 6	SIMULATION OF DISTANCE VECTOR ROUTING ALGORITHM.
DATE:	

AIM:

To implement Simulation of Distance vector routing algorithm
using java programming language.

ALGORITHM:

Step 1: Start
Step 2: Open the Cisco Packet Tracer software.
Step 3: Add the router and PCs according to our design.
Step 4: Configure all the routers and PCs.
Step 5: Trace the destination in PC's command prompt.
Step 6: Verify the output.
Step 7: Stop

CODE:

```
import java.io.*;
public class DVR
{
static int graph[][];
static int via[][];
static int rt[][];
static int v;
static int e;
public static void main(String args[]) throws IOException
{
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
System.out.println("Please enter the number of Vertices: ");
v = Integer.parseInt(br.readLine());
System.out.println("Please enter the number of Edges: ");
e = Integer.parseInt(br.readLine());
graph = new int[v][v];
via = new int[v][v];
rt = new int[v][v];
for(int i = 0; i < v; i++)
for(int j = 0; j < v; j++)
{
if(i == j)
```

```
graph[i][j] = 0;
else
graph[i][j] = 9999;
}
for(int i = 0; i < e; i++)
{
System.out.println("Please enter data for Edge " + (i + 1) + ":");
System.out.print("Source: ");
int s = Integer.parseInt(br.readLine());
s--;
System.out.print("Destination: ");
int d = Integer.parseInt(br.readLine());
d--;
System.out.print("Cost: ");
int c = Integer.parseInt(br.readLine());
graph[s][d] = c;
graph[d][s] = c;
}
dvr_calc_disp("The initial Routing Tables are: ");
System.out.print("Please enter the Source Node for the edge whose cost has changed: ");
int s = Integer.parseInt(br.readLine());
s--;
System.out.print("Please enter the Destination Node for the edge whose cost has changed: ");
int d = Integer.parseInt(br.readLine());
d--;
System.out.print("Please enter the new cost: ");
int c = Integer.parseInt(br.readLine());
graph[s][d] = c;
graph[d][s] = c; dvr_calc_disp("The new Routing Tables are: ");
}
static void dvr_calc_disp(String message)
{
System.out.println();
init_tables();
update_tables();
System.out.println(message);
print_tables();
System.out.println();
}
static void update_table(int source)
{
for(int i = 0; i < v; i++)
{
```

```
if(graph[source][i] != 9999)
{
int dist = graph[source][i];
for(int j = 0; j < v; j++)
{
int inter_dist = rt[i][j];
if(via[i][j] == source)
inter_dist = 9999;
if(dist + inter_dist < rt[source][j])
{
rt[source][j] = dist + inter_dist;
via[source][j] = i;
}
}

}
}
}
static void update_tables()
{
int k = 0;
for(int i = 0; i < 4*v; i++)
{
update_table(k);
k++;
if(k == v)
k = 0;
}
}
static void init_tables()
{
for(int i = 0; i < v; i++)
{
for(int j = 0; j < v; j++)
{
if(i == j)
{
rt[i][j] = 0;
via[i][j] = i;
}
else
{
rt[i][j] = 9999;
via[i][j] = 100;
}
```

```
}
}
}
static void print_tables()
{
for(int i = 0; i < v; i++)
{
for(int j = 0; j < v; j++)
{
System.out.print("Dist: " + rt[i][j] + " ");
}
System.out.println();
}
}
}
```

OUTPUT:

```
Please enter the number of Vertices:
4
Please enter the number of Edges:
5
Please enter data for Edge 1:
Source: 1
Destination: 2
Cost: 1
Please enter data for Edge 2:
Source: 1
Destination: 3
Cost: 3
Please enter data for Edge 3:
Source: 2

Destination: 3
Cost: 1
Please enter data for Edge 4:
Source: 2
Destination: 4
Cost: 1
Please enter data for Edge 5:
Source: 3
Destination: 4
Cost: 4
```

The initial Routing Tables are:
Dist: 0 Dist: 1 Dist: 2 Dist: 2
Dist: 1 Dist: 0 Dist: 1 Dist: 1
Dist: 2 Dist: 1 Dist: 0 Dist: 2
Dist: 2 Dist: 1 Dist: 2 Dist: 0
Please enter the Source Node for the edge whose cost has changed: 2
Please enter the Destination Node for the edge whose cost has changed: 4
Please enter the new cost: 10
The new Routing Tables are:
Dist: 0 Dist: 1 Dist: 2 Dist: 6
Dist: 1 Dist: 0 Dist: 1 Dist: 5
Dist: 2 Dist: 1 Dist: 0 Dist: 4
Dist: 6 Dist: 5 Dist: 4 Dist: 0

RESULT:
Thus, the program is executed successfully and the output is verified.

EX.NO: 07	SIMULATION OF LINK STATE ROUTING ALGORITHM.
DATE:	

AIM:

To implement Simulation of Link state routing algorithm
using java programming language.

ALGORITHM:

Step 1: Start
Step 2: Open the Cisco Packet Tracer software.
Step 3: Add the router and PCs according to our design.
Step 4: Configure all the routers and PCs.
Step 5: Trace the destination in PC's command prompt.
Step 6: Verify the output.
Step 7: Stop

CODE:

```
import java.io.BufferedReader;
import java.io.InputStreamReader;

public class OSPF3 {

public static void main(String[] args) throws Exception{
int n=0;
System.out.println(".....WELCOME TO OSPF.....");
int V[]={0,1,2,3,4};
System.out.println("The length of the V:"+V.length);
int c[][]=new int[V.length][V.length];
System.out.println("Enter cost Matrix:");
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
for(int i=0;i<V.length;i++)

for(int j=0;j<V.length;j++)
c[i][j]=Integer.parseInt(br.readLine());
int s[]={V[0]};
System.out.println(s[0]);
int d[]=new int[V.length];
for(int i=0;i<V.length;i++)
{
d[i]=c[0][i];
}
```

```
System.out.println("The cost Matrix which is entered by you:");
for(int i=0;i<d.length;i++)
System.out.print("\t"+d[i]);
int v1[]=new int[V.length-s.length];
System.out.print("the length of v1 is:\t"+v1.length);
int x=0;
for(int i=0;i<5;i++)
{
for(int j=0;j<s.length;j++)
if(V[i]!=s[j])
{
v1[x]=V[i];
x++;
}
}
int w;
System.out.print("\n The v1 values are:");
for(int i=0;i<x;i++)
System.out.print("\t"+(v1[i]));
for(int i=0;i<5;i++)
{
int min=d[v1[0]];
int z=v1[0];
for(int j=1;j<v1.length;j++)
{
if(min>=d[v1[j]])
{
min=d[v1[j]];
z=v1[j];
}
}
w=z;
System.out.println("The w value is:"+w);
System.out.println("The s value at before inc is:\t"+s.length);
s[n+1]=w;
System.out.println("The s value at after inc is:\t"+s.length);
for(int j=0;j<s.length;j++)
System.out.println("The s values are:\t"+s[j]);
int l=0;
for(int k=0;k<V.length;k++)
{
for(int j=0;j<s.length;j++)
if(V[k]!=s[j])
{
```

```
v1[l]=V[i];
l++;
}
}
for(int j=0;j<v1.length;j++)
{
d[v1[j]]=Math.min(d[v1[j]],d[w]+c[w][v1[j]]);
}
}
System.out.println("The minimum cost from A to all:");
for(int i=0;i<V.length;i++)
System.out.println(""+V[i]+"\t"+d[V[i]]);
```

RESULT:

Thus, the program is executed successfully and the output is verified.

EX.NO: 08	APPLY CAESAR CIPHER SECURITY ALGORITHM FOR NETWORK SECURITY.
DATE:	

AIM:

To Apply Caesar cipher security algorithm for network security.

ALGORITHM:

Step 1: Start
Step 2: Read the plain text from the user.
Step 3: Read the key value from the user.
Step 4: If the key is positive then encrypt the text by adding the key with each character in the plain text.
Step 5: Else subtract the key from the plain text.
Step 6: Display the cipher text obtained above
Step 7: Stop

CODE:

```
class CaesarCipher
{
// Encrypts text using a shift od s
public static StringBuffer encrypt(String text, int s)
{
StringBuffer result= new StringBuffer();
for (int i=0; i<text.length(); i++)
{
if (Character.isUpperCase(text.charAt(i)))
{
char ch = (char)(((int)text.charAt(i) +
s - 65) % 26 + 65);

result.append(ch);
}
else
{
char ch = (char)(((int)text.charAt(i) +
s - 97) % 26 + 97);
result.append(ch);
}
}
return result;
```

```
}
// Driver code
public static void main(String[] args)
{
String text = "ATTACKATONCE";
int s = 4;
System.out.println("Text : " + text);
System.out.println("Shift : " + s);
System.out.println("Cipher: " + encrypt(text, s));
}
```

OUPUT:

```
Text : ATTACKATONCE
Shift: 4
Cipher: EXXEGOEXSRGI
```

RESULT:

Thus, the program is executed successfully and the output is verified.

EX.NO: 09	APPLY TCP PROGRAM FOR DATE/TIME SERVER.
DATE:	

AIM:

To Apply TCP program for date/time server.

SERVER-SIDE ALGORITHM:

Step 1: Create a server socket and bind it to port.
Step 2: Listen for new connection and when a connection arrives, accept it.
Step 3: Send server's date and time to the client.
Step 4: Read client's IP address sent by the client.
Step 5: Display the client details.
Step 6: Repeat steps 2-5 until the server is terminated.
Step 7: Close all streams.
Step 8: Close the server socket.
Step 9: Stop.

CLIENT-SIDE ALGORITHM:

Step 1: Create a client socket and connect it to the server's port number.
Step 2: Retrieve its own IP address using built-in function.
Step 3: Send its address to the server.
Step 4: Display the date & time sent by the server.
Step 5: Close the input and output streams.
Step 6: Close the client socket.
Step 7: Stop

SERVER-SIDE CODE:

```
import java.net.*;
import java.io.*;
import java.util.Date;
public class Server_DT { /** * @param args the command line arguments */ public
static void main(String[] args)throws IOException {
//Step 1. Reserve a port number on the Server to offer this service
ServerSocket ss= new ServerSocket(5000);
//(Optional)To confirm Server Reserved specified port or not
```

```
System.out.println("The Server has reserved port No.: "+ss.getLocalPort()+" for this Service"); //Step 2. Now create a Client Socket on Server for Bidirectonal Communication.
//Socket is created only when client communicates with the server

Socket cs=ss.accept();
//To confirm Server communicated through the socket or not
System.out.println("Client with IP Address "+cs.getInetAddress()+" has communicated via port No.: "+cs.getPort());
Date d=new Date();
String s="Current Date & Time on Server is:"+d;
//Send String s to client via client socket
PrintWriter toclient=new PrintWriter(cs.getOutputStream(),true);
toclient.print(s); toclient.close(); cs.close(); ss.close();
  }
 }
```

CLIENT-SIDE CODE:

```
import java.net.*;
import java.io.*;
public class Client_DT {
/** * @param args the command line arguments
*/ public static void main(String[] args) throws UnknownHostException,IOException
{ // TODO code application logic here
//Step 1. Create a client socket to connect to Server
Socket cs= new Socket("LocalHost",5000);
//To confirm Client is communicating through the port
System.out.println("Client "+cs.getInetAddress()+" is communicating from port No.: "+cs.getPort());
//Receive Date Sent by Server
BufferedReader fromserver=new BufferedReader(new InputStreamReader(cs.getInputStream()));
System.out.println(fromserver.readLine());
fromserver.close(); cs.close();
}
}
```

OUPUT:

date/time will be produced at the localhost port.

RESULT:
Thus, the program is executed successfully and the output is verified.

EX.NO: 10	SIMPLE UDP SOCKET PROGRAM FOR ECHO SERVER CHAT.
DATE:	

AIM:

To make a Simple UDP socket program for echo server client chat.

SERVER-SIDE ALGORITHM:

Step 1:Start
Step 2: Declare the variables for the socket
Step 3: Specify the family, protocol, IP address and port number
Step 4: Create a socket using socket() function
Step 5: Bind the IP address and Port number
Step 6: Listen and accept the client's request for the connection
Step 7: Read and Display the client's message
Step 8: Stop

CLIENT-SIDE ALGORITHM:

Step 1: Start
Step 2: Declare the variables for the socket
Step 3: Specify the family, protocol, IP address and port number
Step 4: Create a socket using socket() function
Step 5: Call the connect() function
Step 6: Read the input message
Step 7: Send the input message to the server
Step 8: Display the server's echo
Step 9: Close the socket
Step 10: Stop

CLIENT-SIDE CODE:

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Scanner;
public class udpBaseClient_2 {
public static void main(String args[]) throws IOException
{

Scanner sc = new Scanner(System.in);
// Step 1:Create the socket object for
// carrying the data.
```

```
DatagramSocket ds = new DatagramSocket();
InetAddress ip = InetAddress.getLocalHost();
byte buf[] = null;
// loop while user not enters "bye"
while (true)
{
String inp = sc.nextLine();
// convert the String input into the byte array.
buf = inp.getBytes();
// Step 2 : Create the datagramPacket for sending
// the data.
DatagramPacket DpSend = new DatagramPacket(buf, buf.length, ip, 1234);
// Step 3 : invoke the send call to actually send
// the data.
ds.send(DpSend);
// break the loop if user enters "bye"
if (inp.equals("bye"))
break;
}
}

}
```

SERVER-SIDE CODE:

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.net.SocketException;
public class udpBaseServer_2
{
public static void main(String[] args) throws IOException
{
// Step 1 : Create a socket to listen at port 1234
DatagramSocket ds = new DatagramSocket(1234);
byte[] receive = new byte[65535];
DatagramPacket DpReceive = null;
while (true)
{
// Step 2 : create a DatgramPacket to receive the data.
DpReceive = new DatagramPacket(receive, receive.length);
// Step 3 : revieve the data in byte buffer.
ds.receive(DpReceive);
System.out.println("Client:-" + data(receive));
// Exit the server if the client sends "bye"
```

```
if (data(receive).toString().equals("bye"))
{
System.out.println("Client sent bye.....EXITING");
break;
}
receive = new byte[65535];
}
}
public static StringBuilder data(byte[] a)
{
if (a == null)
return null;
StringBuilder ret = new StringBuilder();
int i = 0;
while (a[i] != 0)
{
ret.append((char) a[i]);
i++;
}
return ret;
}
}
```

OUTPUT:

```
 Hellol am Client....Bye
Client:- byeClient sent bye.....EXITING
```

RESULT:

Thus, the program is executed successfully and the output is verified.

EX.NO: 11	DEVELOP A PROGRAM FOR CONGESTION CONTROL USING LEAKY BUCKET ALGORITHM
DATE:	

AIM:

To Develop a program for congestion control using Leaky bucket algorithm.

ALGORITHM:

Step 1: Start

Step 2: Set the bucket size or the buffer size.

Step 3: Set the output rate.

Step 4: Transmit the packets such that there is no overflow.

Step 5: Repeat the process of transmission until all packets are transmitted. (Reject packets where its size is greater than the bucket size)

Step 6: Stop

CODE:

```
import java.util.Scanner;
import java.util.TreeSet;
class LeakyBucket {
public static void main(String[] args) {
TreeSet<Integer> arrivalTimes = new
TreeSet<Integer>();
Scanner scanner = new Scanner(System.in);
System.out.println("Enter Burstiness in Network");
int L = scanner.nextInt();
System.out.println("Enter Packet Inter Arrival");
int I = scanner.nextInt();
System.out.println("Enter Arrival Times of Packets
(Enter non integer value at end)");
while (scanner.hasNextInt()) {
arrivalTimes.add(scanner.nextInt());
}
int LCT = arrivalTimes.first();
```

```
int X = 0;
for (Integer t : arrivalTimes) {
int temp = X - (t - LCT);
if (temp > L) {
System.out.println("Packet at time " + t + " is Non
Conforming");
} else {

X = temp + I;
LCT = t;
System.out.println("Packet at time " + t
+ " is Conforming");
}
}
}
}
```

OUTPUT:

```
Enter Burstiness in Network
6
Enter Packet Inter Arrival
4
Enter Arrival Times of Packets (Enter
non integer value at end)
1 2 3 5 6 8 11 12 13 15 19 q
Packet at time 1 is Conforming
Packet at time 2 is Conforming
Packet at time 3 is Conforming
Packet at time 5 is Non Conforming
Packet at time 6 is Non Conforming
Packet at time 8 is Conforming
Packet at time 11 is Conforming
Packet at time 12 is Non Conforming
Packet at time 13 is Non Conforming
Packet at time 15 is Conforming

Packet at time 19 is Conforming
```

RESULT:

Thus, the program is executed successfully and the output is verified.

EX.NO: 12	STUDY THE SIMULATION OF NETWORK SIMULATOR
DATE:	

AIM:

To study about the simulation of network simulator.

NETWORK SIMULATOR:

Simulation is a very important technology in modern time. Computer assisted simulation can model hypothetical and real-life objects or activities on a computer to study the well-designed structure. A network simulator is a system of implementing the network on the computer through which the performance of the network is calculated. The computer assisted simulation technologies are applied in the simulation of networking algorithms. The functional network field is narrower than general simulation and it is natural that more specific requirements will be placed on network simulations.

Network simulator allows the researchers to test the scenarios that are difficult or expensive to simulate in real world. Design of various network topologies using nodes, hosts, hubs, bridges, routers and mobile units etc. is possible. The network simulators are of various types which can be compared on the basis of: range (simple to the complex), specification of nodes, links and traffic between the nodes. Specifying about the protocols used to handle traffic in a network, user friendly applications (allow users to easily visualize the simulated environment.), text-based applications (permit more advanced forms of customization) and programming-oriented tools (providing a programming framework that customizes to create an application that simulates the networking environment to be tested).

Concepts in network simulation:

Generally, network simulators try to represent the real world networks and it is a useful technique, given that the activities of a network can be modelled by calculating the interaction between the different network components (they can be end-host or network entities such as routers, packets or physical links) using mathematical formulas. They can also be modelled by actually or virtually capturing and playing back experimental observations from real networks. Upon receipt of the observation data from simulation experiments, the behaviour of the network and protocols supported are analysed in a series of offline test experiments. All types of attributes can also be modified in a controlled manner to assess how the network can behave under different parameter combinations. Another feature of network simulation worth noticing is that the simulation program can be used and analysed together with various strategy, links, applications etc. Typically, users can then adapt

the simulator to fulfil their exact needs. Simulators support the most popular protocols and networks such as WLAN, TCP and WSN.

Simulators:

Most of the commercial simulators are Graphical User Interface (GUI) driven, while some network simulators are Command-Line Interface (CLI). The design of the network describes the state of the network (nodes, routers, switches and links) and the events (data transfer, transmission delay, packet error etc.). The major output of simulation is the trace files which log every packet and event that occurred during simulation and is used for analysis. Also provides other tools to facilitate visual analysis of trends and potential trouble spots. Most of the network simulators are discrete event, in which the list of pending "events" are stored and processed in order. Some events trigger the future events (i.e.) the event of the arrival of a packet at one node triggering the event of the arrival of that packet at a downstream node. Simulation of networks is a very difficult task. For example, if blocking is high, then evaluation of the average occupancy is challenging because of high variance. To evaluate the probability of buffer overflow in a network, the time required for a precise answer can be enormously large. Techniques like "control variants" and “sampling" have been developed to speed simulation.

List of Network Simulators:

There are many both free/open-source and proprietary simulators. Examples of notable simulation software are ordered based on how frequently they are mentioned in research papers.

1. NS2 (Network Simulator 2)
2. NS3 (Network Simulator 3)
3. OPNET
4. OMNeT++.
5. NetSim.
6. QualNet.
7. J-Sim.

Uses of Network Simulators:

Network simulators serve a variety of requirements. Simulators are relatively fast and economical when compared to the cost and time involved in setting up an entire bed containing multiple network computers, data links and routers. They authorize researchers to test scenarios that might be particularly difficult or expensive to emulate using a real hardware - for instance simulating a scenario with several nodes or experimenting with a new protocol in the network. Simulators are mainly useful in

allowing researchers to test new networking protocols or changes to existing protocols in controlled and reproducible surroundings.

A typical simulator encompasses a wide range of networking technologies and can help the users to build complex networks from basic building blocks such as selection of nodes and links. Various types of nodes in Hierarchical networks resembling computers, hubs, bridges, routers, links, switches mobile units etc can be designed with the help of simulators.

Various types of Wide Area Network (WAN) technologies like TCP, ATM, IP etc. and Local Area Network (LAN) technologies like Ethernet, token rings etc., can be imitated with a simulator and the user can examine various standard results apart from devising some novel protocol or routing strategy. Network simulators are widely used to simulate battlefield networks in Network-centric warfare.

Overview of Network Simulators:

Currently there are many network simulators that have different features in different aspects. Short lists of the current network simulators include NS-2, NS-3, OPNET, OMNeT++, NETSIM, QualNet, and J-Sim. These network simulators are selected for discussion regarding their features, advantages and restrictions in this paper.

NS2 :

The Ns2 is a discrete event simulator targeted at packet level networking research and provides substantial support to simulate group of protocols like TCP, UDP, FTP and HTTP. It comprises of two simulation tools. Ns-2 is primarily UNIX based and fully simulates a layered wire or wireless network from the physical radio transmission channel to high-level applications. The simulator is written in C++ and a script language called OTCL.

C++: C++ is fast to run but slower to change, making it suitable for detailed protocol implementation.

OTCL: OTCL runs much slower but can be changed very quickly (and interactively), making it ideal for simulation configuration. Ns provides glue to make objects and variables appear on both languages. NS2 uses an OTcl interpreter by which the user writes an OTcl script that defines the network, (number of nodes and links) the transaction in the network (sources destinations, type of traffic) and the type of protocols used. The outcome of the simulation is a trace file that can be used for data processing (calculate delay, throughput etc). To visualize the simulation, a program called Network Animator (NAM) is used. It visualizes the packets as they propagate throughout the network. The ns2 simulator has numerous features that make it suitable for our simulations. • A network environment for ad-hoc networks,

- Wireless channel modules (e.g.,802.11),

- Routing along multiple paths,
- Mobile hosts for wireless cellular networks.
- Download of ns-2 source code is possible and can be compiled for multiple platforms.

Advantages:

1. NS2 has large number of available models, realistic mobility models, powerful and flexible scripting and simulation setup, large user community and ongoing development.

2. NS2 provides an easy traffic and movement pattern by including an efficient energy model.

3. Complex scenarios can be easily tested.

4. Popular for its modularity.

Limitations:

1. NS2 needs to be recompilation every time if there is a change in the user code.

2. Real system is too complex to model i.e. complicated infrastructure.

NS3:

The ns-3 simulator is a discrete-event network simulator for Internet systems, targeted primarily for research and learning purpose. The ns-3 project, started in 2006, is open-source free software, licensed under the GNU GPLv2 license. It will rely on the current contributions of the community to develop new models, debug or maintain the existing ones, and share the results. Ns3 is mainly used on LINUX systems and not limited to internet-based systems alone.

C++: implementation of simulation and core model. Ns-3 is built as a library which may be statically or dynamically linked to a C++ main program. These libraries describe the beginning of simulation and their topology.

Python: C++ wrapped by Python. Python programs to import an “ns3” module. The features of NS3 simulator are given below.

1. Modular, documented core

2. C++ programs and Python scripting

3. Alignment with real systems

4. Software integrations

5. Virtualization and test bed integration

6. Attribute system

7. Updated models

Advantages:

1. High modularity than its ancestor NS2.

2. Support simulation for TCP, UDP, ICMP, IPv4, multicast routing, P2P and CSMA protocols.

3. Support for ported code should make model validation easier and more credible.

4. Much more flexible than any other simulators.

5. Wide range of use in both optimization and expansion of the existing networks.

Limitations:

1. NS3 still suffers from lack of credibility.

2. NS3 is intended to replicate the successful mode of NS2 in which various organizations contributed to the models and components based on the framework of NS2.

3. NS3 needs a lot of specialized maintainers in order to avail the merits of NS3 as the commercial OPNET network simulators.

4. Active maintainers are required to respond to the user questions, bug reports and help to Test & validate the system.

OPNET: OPNET is extensive and powerful simulation software which enables to simulate heterogeneous networks with a range of protocols. Modeler is a commercial network simulation environment for network modeling and simulation. It allows the users to design communication networks, procedure, protocols, and applications with flexibility and scalability. The network is simulated graphically and the graphical editors mirror the structure of actual networks and their mechanism. An object-oriented system approach is used in the modeler.

C (C++): The programming language in OPNET is C (recent releases support C++ development). The initial configuration (topology setup, parameter setting) is usually achieved using Graphical User Interface (GUI), a set of XML files or through C library calls. Simulation scenarios (e.g., parameter change after some time, topology update, etc.) usually require writing C or C++ code; although in simpler cases one can use special “scenario” parameters (e.g., link fail/restore time).

OPNET’s detailed features include:

1. Fast discrete event simulation engine

2. Lot of component library with source code

3. Object-oriented modeling

4. Hierarchical modeling environment

5. Scalable wireless simulations support

6. 32-bit and 64-bit graphical user interface

7. Customizable wireless modeling

8. Discrete Event, Hybrid, and Analytical simulation

9. 32-bit and 64-bit parallel simulation kernel

10. Grid computing support

11. Integrated, GUI-based debugging and analysis

12. Open interface for integrating external component libraries

Advantages:

1. Leverage three different simulation technologies to efficiently tradeoff simulations detail and speed.

2. Fast discrete event simulation engine.

3. Customizable wireless modeling.

4. Integrated GUI based debugging and analysis .

Limitations:

1. Complex GUI operation.

2. It does not allow much number of nodes within a single connected device.

3. Accuracy of results is limited by the sampling resolution.

4. Simulation is inefficient if nothing happens for long periods

OMNET++: It is a component-based, modular and open architecture discrete event simulator framework. The most common use of OMNeT++ is for simulation of networks, but it is also used for queuing network simulations and other areas as well. It is licensed under its own Academic Public License, which permits GNU Public License like freedom but only in non-commercial settings. It provides component architecture for models.

C++: The C++ class library comprises of simulation kernel and utility classes (for random number generation, statistics collection, topology discovery etc) -- this one is used to create simulation components (simple modules and channels); infrastructure to assemble simulations from these components and configure (NED language, ini files); runtime user interfaces or environments for simulations (Tkenv, Cmdenv); an Eclipse-based simulation IDE for designing, running and evaluating simulations;

extension interfaces for real-time simulation, emulation, MRIP, parallel distributed simulation, database connectivity and so on.

The OMNeT++ components include:

1. Simulation kernel library
2. Compiler for the NED topology description language (nedc)
3. Graphical network editor for NED files (GNED)
4. GUI for simulation execution, links into simulation executable (Tkenv)
5. Command-line user interface for simulation execution (Cmdenv)
6. Graphical output vector plotting tool (Plove)
7. Graphical output scalars visualization tool (Scalars)
8. Model documentation tool (opp_neddoc)
9. Utilities (random number seed generation tool, make file creation tool, etc.)
10. Documentation, sample simulations, etc.

Advantages:

1. Provides a powerful GUI environment.
2. Tracing and debugging are much easier than other simulators.
3. Accurately models most hardware and include the modelling of physical phenomena.

Limitations:

1. It does not offer a great variety of protocols and very few protocols have been implemented, leaving users with significant background work.
2. Poor analysis and management of typical performance.
3. The mobility extension is relatively incomplete.

NETSIM :

NetSim is a discrete event simulator developed by Tetcos in 1997, in association with Indian Institute of Science. It has also been featured with Computer Networks and Internets V edition by Dr. Douglas Comer, published by Prentice Hall. It has an object-oriented system simulating environment to support simulation and analysis of voice and data communication scenarios for High Frequency Global Communication Systems (HFGCS).

Java: It creating fast, platform independent software that could be used in simple, consumer electronic products. Java designed for simple, efficient, platform-

independent program for creating WWW-based programs. Using Java one can create small programs called applets that are entrenched into an HTML document and viewable on any Java-compatible browser. Java applets are compiled into a set of byte-codes, or machine-independent processing instructions.

Advantages:

1. NetSim has a GUI which features drag and drop functionality for devices, links etc. i.e. Modeling in NetSim is simple and user friendly.

2. It has a built in analysis framework that provides intra and inter-protocol [19] performance comparison with graphical options.

3. Data packet and control packet flow can be visual-ized through NetSim built-in packet animator.

4. It is easy to learn all about NetSim.

Limitations:

1. NetSim is a single process discrete event simulator. A single event queue is used for the simulation which at any given time contains one entry for each station on the network. 2.
2. Free version of NetSim is not available.

QualNet: It is a commercial network simulator from Scalable Network Technologies, Inc in 2000-2001. It is an ultra-high fidelity network simulation software that predicts wired or wireless platform network and their device performance. For a large, heterogeneous network and distributed applications such networks are executed.

C++: For implementing new protocols, Qualnet uses C/C++ and follows a procedural paradigm. It uses the parallel simulation environment for the basic operations of complex systems (PARSEC). Hence it can run on distributed machines.

Features:

- QualNet can support real-time speed to enable network developers and designers to run multiple analysis by varying model, network, and traffic parameters in a short time.

- It can model thousands of nodes by taking advantage of the latest hardware and parallel computing techniques. A very powerful simulation tool that can support simulation of 500 to 20,000 nodes.

- QualNet can run on cluster, multi-core, and multi-processor systems to model large networks with high fidelity.

- It provides high fidelity commercial protocol and device models to enable more accurate modelling of real-world networks.

- It enables the designer to design large wired and wireless networks using pre-configured or user-designed models.
- It also facilitates to design new protocol models and to optimize new and existing models.
- QualNet can connect to other hardware and soft-ware applications, such as OTB, real networks, and a third-party graphical software in order to enhance the value of the network model.

Advantages:

1. QualNet supports multiprocessor systems and distributed computing.
2. It can simulate a mixture of both wired and wireless networks.
3. It provides GUI that is convenient and easy-to-use.
4. It facilitates sophisticated animation capabilities.
5. It can run on cluster, multi-core, and multi-processor systems.

Limitations:

1. The simulation tool QualNet is an extension of GloMoSim which is being commercialized.
2. Installation of QualNet on Linux is difficult.
3. The java based user interface provided by this simulation software is slow.

JSIM :

J-Sim has been developed by a team at the Distributed Real-time Computing Laboratory (DRCL). The project has been sponsored by the National Science Foundation (NSF), DARPA's, Air Force Office of Scientific Research's Multidisciplinary University Research Initiative, the Ohio State University and University of Illinois at Urbana-Champaign. J-Sim is free and available with source code.

Java: Java is easy to learn and easy to use. In case of any problems, source texts provided with J-Sim can be used to create fresh code, compiled in the target environment, thus 100% compatible with JVM used. Java provides a class called Thread whose instances run parallel with other such instances. It is an object-oriented language, providing the concepts of classes, instances, encapsulation, inheritance and polymorphism. J-Sim provides basic classes for simulation, process and queue which can be either directly used or extended according to specific user's requirements.

TCL: Scripting is an essential part of J-Sim, used to "glue" all the components and define how the operation of the system takes place. It makes possible to manipulate Java objects in the TCL environment, such as creating an object, invoking a method or accessing a field variable of a Java object.

Advantages:

1. A simple and well-defined component-based soft-ware architecture with object oriented paradigm. This facilitates hierarchical modeling of complex systems

2. Simulation engine is built in the runtime and is transparent to components.

3. It provides Process-oriented modeling and wait () methods, synchronization methods to further ex-tend programming flexibility.

4. It can work with both discrete event simulation and real-time process-based simulation.

5. It will implement a parallel simulation in the autonomous component architecture.

Limitations:

1. Java has some security restrictions. So forth JSIM can be prevented from persistence in data.

2. Operation of JSIM required clear concept of queuing algorithm.

3. The graphical model designer, which has limited capabilities as of now (can only be used to design a model), is intended to be a GUI-based model builder that would do much of the code generation that has to be done manually.

Comparison of Simulators:

MANET simulators exhibit different features and models. The choice of a simulator should be driven by the requirements and the level of details necessary. If high-precision PHY layers are desirable, then ns-2 is the wisest option. On contrary, if the wireless tools have no impact on the targeted protocol, modern simulators (like NAB or Jane) which propose high-level abstractions and polished object-oriented designs will be adapted. These targeted nodes determine the choice of the simulation tool. Sequential simulators should not be anticipated to run more than 1,000 nodes. To end with, most noncommercial simulators suffer from lack of good certification and support. Using a commercial one, may help in case of difficulty. In addition, commercial simulators feature extensive list of supporting protocols, while open-source solutions give complete empowerment. These are listed in the table below.

Name	License	Programming Language	Supported Operating System
NS2	Open source	C++, TCL	GNU/Linux, FreeBSD, Mac OS

			X, Windows XP, Windows Vista and Win. 7.
NS3	Open source	C++, Python	GNU/Linux, FreeBSD, Mac OS X, Windows XP, Windows Vista & Windows 7.
OPNet	Commercial	C	Windows XP, Vista, 7 & Windows NT 4.0.
OMNET++	Open source	C++	Windows XP or Later, Linux, Mac OS X,
NETSIM	Open source	Java	Windows (7, Vista) and windows XP
QUALNET	Commercial	C++	UNIX, Window-MAC, Linux
JSIM	Open source	Java, TCL	Windows XP, Vista & 7, MAC OS X, Linux.

Result:

Thus, the simulation of network simulator is studied.