COMPUTER NETWORKS LABORATORY MANUAL

VI Semester
Course Code: 20CSEL67

As per the Choice Based Credit System- 2020 Scheme

Version 1.1

With effect from: April 2023

Editorial Committee

CN Lab Faculty, Department of CSE

Approved by

H.O.D, Department of CSE

Global Academy of Technology

Department of Computer Science and Engineering
Rajarajeshwari Nagar, Bengaluru – 560 098

DOCUMENT LOG

Name of the document	CN LAB MANUAL
Current version number	1.1
Date of Updation	17/04/2023
Subject code	20CSEL67
Authored by	CN Lab Faculty
Updated by	Prof Rudramurthy V C, Prof Haripriya C, Prof Naveen Kumar C
Verified by	Prof Rudramurthy V C, Prof Haripriya C, Prof Naveen Kumar C
Approved by	Dr. Kumaraswamy S, HOD, CSE

Table of Contents

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Vision of the Department

To achieve academic excellence and strengthen the skills to meet emerging challenges of Computer Science and Engineering.

Mission of the Department

M1: To impart strong theoretical foundations in the field of Computer Science and Engineering accompanied with extensive practical skills.

M2: To inculcate research and innovation spirit through interaction with industry and carry out projects that address societal needs.

M3: Instill professional ethics and values with concern for environment.

Program Educational Objectives (PEOs) of Department

After the course completion, CSE graduates will be able to:

- Succeed in engineering/management positions with professional ethics.
- Engage in improving professional knowledge through certificate/post-graduate programs in engineering or management.
- Establish themselves as entrepreneurs and contribute to the Society.

Program Specific Outcomes (PSOs)

PSO1: Design, implement and test System Software and Application Software to meet the desired needs.

PSO2: Develop solutions in the area of Communication Networks, Database Systems and Computing Systems.

Program Outcomes (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Course Details

Course Name: Computer Networks Laboratory

Course Code: 20CSEL67

Course prerequisite: Data Communication

Course Objectives

Upon completion of this course, students are expected to:

1. Illustrate topology formation and working of stop-and-wait protocol.
2. Examine working of network connections and LAN working.
3. Illustrate the working of error detection and finding the shortest path using Bellman-ford algorithm.
4. Examine RSA algorithm used for security and Congestion Control concepts.
5. Demonstrate socket programming using TCP protocol.

COMPUTER NETWORKS LABORATORY

Subject Code: 20CSEL67	**CIE Marks: 50**
Hours/Week: 02	**SIE Marks: 50**
Duration of SEE (Hours): 03	

Implement the following in NS2:

1. Implement Ring topology and Bus topology operation using NS2.
2. Implement three nodes point – to – point network with duplex links between them. Set the queue size, vary the bandwidth and find the number of packets dropped.
3. Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source /destination.
4. Write a NS2 script to implement the operation of Stop and Wait protocol
5. Implement simple ESS and with transmitting nodes in wire-less LAN by simulation and determine the performance with respect to transmission of packets.

Implement the following in Java:

6. Write a program for error detecting code using CRC-CCITT (16-bits).
7. Write a program to find the shortest path between vertices using Bellman-Ford algorithm.
8. Write a Java program for congestion control using leaky bucket algorithm.
9. Write a Java program to Implement RSA algorithm to encrypt and decrypt the data while sending it and decrypt while receiving.
10. Write a Java program to Implement TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present

Course Outcomes

The students should be able to:

CO1	Demonstrate topology formation and working of stop-and-wait protocol in NS2
CO2	Examine point – to – point network and LAN working (Wired and Wireless) using NS2
CO3	Illustrate the working of error detection using CRC and finding the shortest path using Bellman-ford algorithm in Java
CO4	Examine RSA algorithm used for security and Congestion Control by Leaky bucket in Java
CO5	Demonstrate socket programming using TCP protocol in Java.

Conduction of Practical Examination:

Scheme of Examination (CIE):

- Experiment distribution.
  - For laboratories having only one part: Students are allowed to pick one experiment from the lot with equal opportunity.
  - For laboratories having PART A and PART B: Students are allowed to pick one experiment from PART A and one experiment from PART B, with equal opportunity.
- Change of experiment is allowed only once and marks allotted for procedure to be made zero of the changed part only.
- Marks Distribution *(Coursed to change in accordance with university regulations)*
  - For laboratories having only one part – Procedure + Execution + Viva-Voce: 15+70+15 = 100 Marks
  - For laboratories having PART A and PART B
    - Part A – Procedure + Execution + Viva = 8 + 35 + 7 = 50 Marks
    - Part B – Procedure + Execution + Viva = 8 + 35 + 7 = 50 Marks

CO-PO-PSO MAPPING

CO/PO-PSO	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO10	PO11	PO12	PSO 1	PSO 2
CO1	2	2	-	-	1	-	-	-	1	1	-	1	-	2
CO2	2	2	-	-	1	-	-	-	1	1	-	1	-	2
CO3	2	2	-	-	1	-	-	-	1	1	-	1	-	2
CO4	2	2	-	-	-	-	-	-	1	1	-	1	-	2
CO5	2	2	-	-	-	-	-	-	1	1	-	1	-	2
Average	2	2	-	-	1	-	-	-	1	1	-	1	-	2

High-3: Medium-2: Low

LAB EVALUATION PROCESS

CIE Evaluation

Sl. No	Activity	Marks
1	Write-up	15
2	Conduction	70
3	Viva Voce	15
	Total	**100 reduced to 20**

End of Semester CIE Calculation

Sl. No	Activity	Marks
1	CIE Evaluation	20
2	Average of Weekly Entries	30
	Total	**50**

LAB RUBRICS

Rubrics for Evaluation of Record

Attribute	Max. Marks	Good	Satisfactory	Poor
		4-5	2-3	0-1
Writeup	**05**	Writes the program without errors.	Writes the program with few mistakes.	Unable to write the program.
	Max. Marks	**10-20**	**5-9**	**0-4**
Execution of program	**20**	• Debugs the program independently. • Executed the program for all possible inputs.	• Works with little help from faculty. • All possible input cases not covered.	• Unable to complete the execution of the program within the lab session.
	Max. Marks	**3-5**	**1-2**	**0**
Viva Voce	**05**	• Able to explain the logic of the program. • Answered all questions.	• Partially understood the logic of the program. • Answered few questions.	• Not understood the logic of the program. • Not answering any questions

Rubrics for Evaluation of Internal Test

Attribute	Max Marks	Good	Satisfactory	Poor
		4-5	2-3	0-1
Writeup	**05**	• Completes source code. • No syntax and logical errors. • All possible inputs listed with expected output.	• Completes source code. • Syntax and logical errors exist. • All possible inputs listed with expected output.	• Incomplete source code. • Change of program.
	Max Marks	**21-30**	**6-20**	**1-5**
Execution	**30**	Able to debug the program and get the right set of outputs.	• Program, executed for only few input cases.	Not executed.
	Max Marks	**3-5**	**1-2**	**0**
Viva Voce	**05**	Answering all questions.	Answering few questions.	Not Able to answer basic questions.

CHAPTER 1

1. Introduction to NS-2

NS-2 stands for Network Simulator Version 2. It is an open-source event-driven simulator designed specifically for research in computer communication networks. Network Simulator-2 (NS2) is a popular open source network simulator for carrying out network experimentation. Way back when it was being designed, its primary usage was to analyze the performance of congestion control algorithms implemented in Transmission control protocol (TCP). Even today, it remains the most widely used network simulator for TCP research. Over the period of time, it gained wide acceptance in industry, and now supports simulation of latest wired as well as wireless networking protocols (e.g., routing algorithms, TCP, User Data Protocol (UDP)) and paradigms such as Mobile Ad hoc Networks (MANETs) Vehicular Ad hoc Network (VANETs), etc.

Another simulator called ns-3 has gained a lot of popularity in the recent past. It is not a sequel of NS-2. NS-3 APIs are not compatible with those of NS-2 API. Both are completely different tools.

Features of NS-2:

- It is a discrete event simulator for networking research.
- It provides substantial support to simulate protocols like TCP, FTP, UDP & DSR.
- It simulates wired and wireless network.
- It is primarily UNIX based.
- Uses TCL as its scripting language.
- Otcl: Object oriented support Tcl
- TclCL: Tcl with Classes and OTcl linkage
- Discrete event scheduler

Basic Architecture of NS2

NS-2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl). While the C++ defines the internal mechanism (i.e., a backend) of the simulation objects, the OTcl sets up simulation by assembling and configuring the objects as well as scheduling discrete events. The C++ and the OTcl are linked together using TclCL. The Basic architecture of NS is as shown in Fig 1.

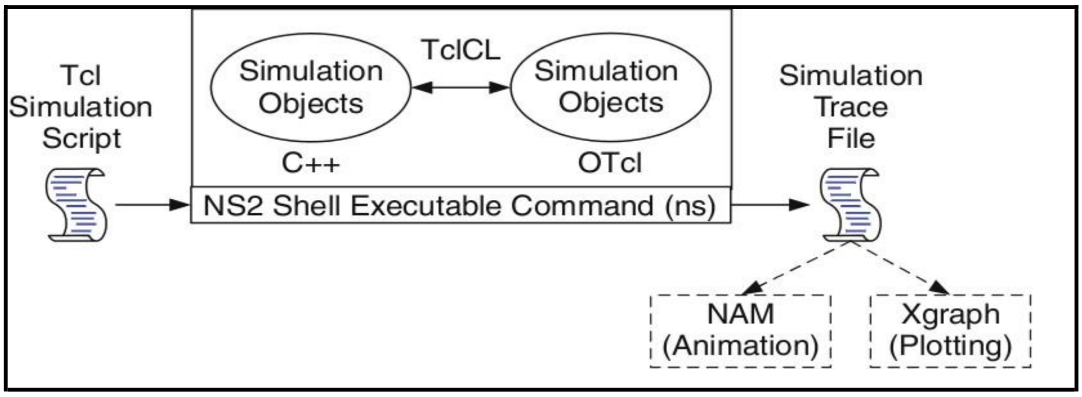


Fig. 1 Basic Architecture of Network Simulator

Why two languages? (TCL and C++)

- NS2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl).
- The C++ defines the internal mechanism (i.e.,a backend) of the simulation objects.
- The OTcl sets up simulation by configuring the objects as well as scheduling discrete events (i.e., a frontend).
- The C++ and the OTcl are linked together using TclCL.
- NS2 uses OTcl to create and configure a network, and uses C++ to run simulation.
- C++ is fast to run but slow to change.
- OTcl, on the other hand, is slow to run but fast to change.
- We write a Tcl simulation script and feeditas an input argument to NS2 when running a simulation (e.g., executing “ns myfirst_ns.tcl”).
- Here, “ns” is a C++ executable file obtained from the compilation.
- myfirst_ns.tcl is an input configuration file specifying system parameters and configuration such as nodes, link, and how they are connected.
- C++ is used for the creation of objects because of speed and efficiency.
- OTcl is used as a front-end to setup the simulator, configure objects and schedule events because of its ease of use.

Tcl scripting

- Tcl is a general purpose scripting language. [Interpreter]
- Tcl runs on most of the platforms such as Unix, Windows, and Mac.
- The strength of Tcl is its simplicity.

- It is not necessary to declare a data type for variable prior to the usage.

Structure of NS-2 Program:

- Creating a Simulator Object
- Setting up files for trace & NAM
- Tracing files using their commands
- Closing trace file and starting NAM
- Creating LINK & NODE topology &Orientation of links

2. Working of NS-2

- NS2 provides users with executable command **ns** which takes an input argument, the name of a **Tcl** simulation scripting file.
- Users are feeding the name of a Tcl simulation script (which sets up a simulation) as an input argument of an NS2 executable command ns.
- In most cases, a simulation trace file is created, and is used to plot graph and/or to create animation.

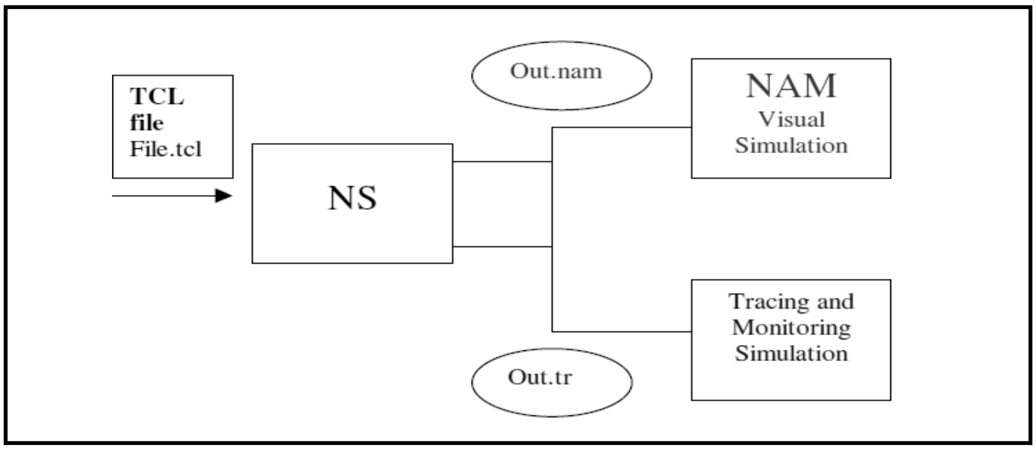


Fig 2. Working of Network Simulator 2

Trace file and NamTrace file:

- Once the simulation is complete, we can see two files: “trace.tr”, and “nam.out”.
- The trace file (trace.tr) is a standard format used by ns2.
- In ns2, each time a packet moves from one node to another, or onto a link, or into a buffer, etc., it gets recorded in this trace file.
- Each row represents one of these events and each column has its own meaning.

- Start nam with the command 'nam <nam-file>' where '<nam-file>' is the name of a nam trace file that was generated by ns.

Network Animator Window

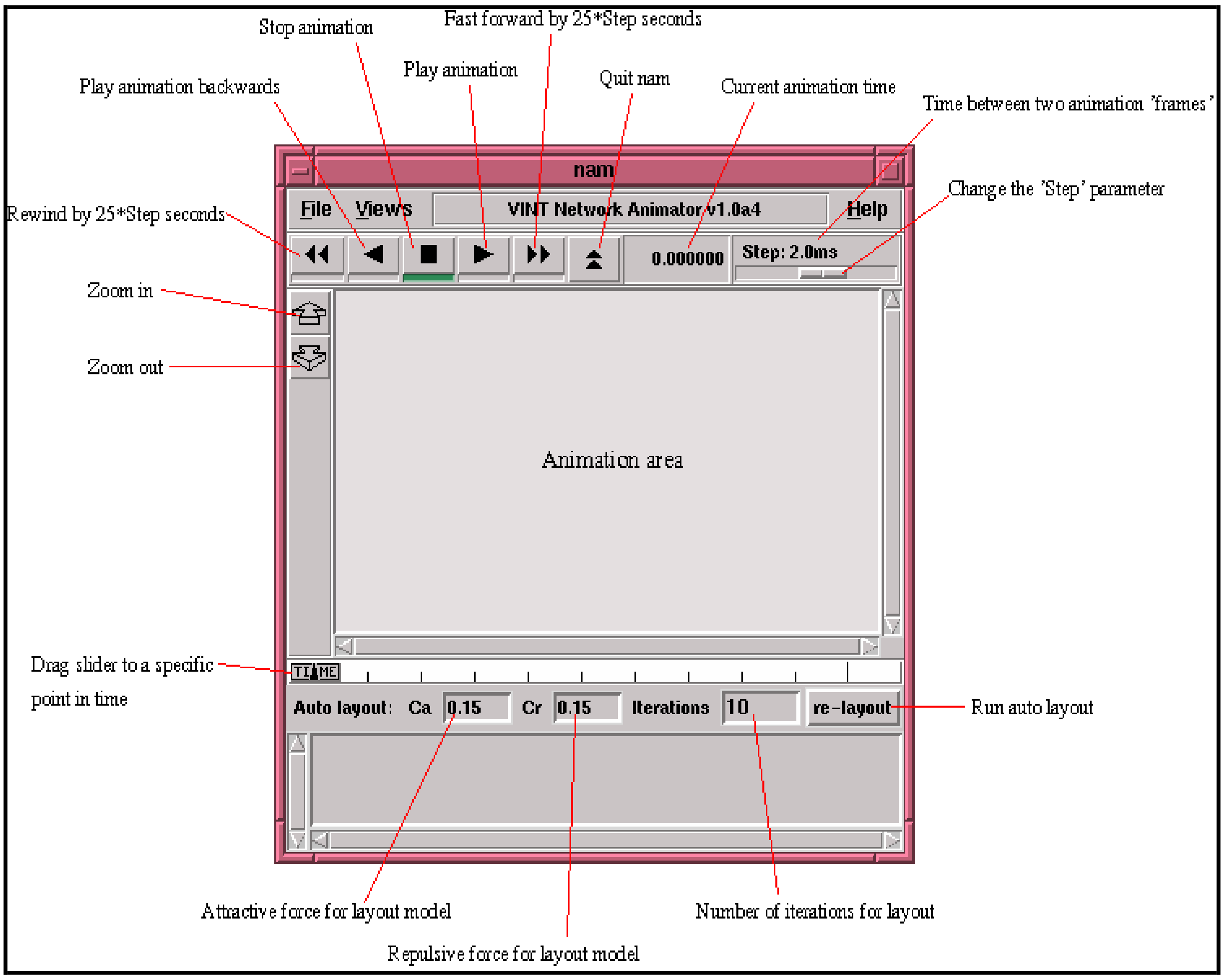

Stop animation
Fast forward by 25*Step seconds
Play animation backwards
Play animation
Quit nam
Current animation time
Time between two animation 'frames'
Change the 'Step' parameter
Rewind by 25*Step seconds
Zoom in
Zoom out
nam
File
Views
VINT Network Animator v1.0a4
Help
0.000000
Step: 2.0ms
Animation area
Drag slider to a specific point in time
TIME
Auto layout:
Ca
0.15
Cr
0.15
Iterations
10
re-layout
Run auto layout
Attractive force for layout model
Number of iterations for layout
Repulsive force for layout model

Fig. 3 Details of NAM Window

Advantages and Disadvantages of NS2:

Advantages

1. Open Source
2. Complex scenarios can be easily tested.
3. Results can be quickly obtained – more ideas can be tested in a smaller time frame.
4. Supported protocols
5. Supported platforms
6. Modularity

Disadvantages

1. Limitation in designing large scale systems
2. May be slow compared to real time network and computationally expensive
3. Does not reflect reality in large and complex networks.
4. Statistical uncertainty in results

Steps to create a scenario file:

Step1: Declare Simulator and setting output file

Step2: Setting Node and Link

Step3: Setting Agent

Step4: Setting

Application

Step5: Setting Simulation time and

schedules Step6: Declare finish.

Step 1: Declare Simulator and setting

$ns [new Simulator]	#first line of tcl script. Creates ns object
get file [open out.tr w] $ns trace-all $file	#open the trace file
get namfile [open out.nam w] $ns namtrace-all $namfile	#open the nam file

Step 2: Setting Node and Link

$ n0 [$ns node]	setting a node
ns duplex-link $n0 $n2 3Mb 5ms DropTail	#bidirectional link between n0 and n2 is declared bandwidth 3Mbps and delay 5ms. DropTail is a waiting queue type.
$ns duplex-link-op $n0 $n2 orient right-down	#Sets positions of node and link for Nam. It does not affect to the result of simulation
$ns queue-limit $n2 $n3 20	#The length of queue on the link from n2 to n3 is 20[packets].
$ns duplex-link-op $n2 $n3 queuePos 0.5	#The position of queue is set for Nam, 0.5 is the angle between link and queue, it equals to (0.5_).

Step 3: Setting Agent

UDP Agent: To use UDP in simulation, the sender sets the Agent as UDP Agent while the receiver sets to Null Agent. Null Agents do nothing except receiving the packets.

set udp [new Agent/UDP] $ns attach-agent $n0 $udp set null [new Agent/Null] $ns attach-agent $n3 $null	#udp and null Agent are set for n0 and n3, respectively
$ns connect $udp $null	Declares the transmission between udp and null.
$udp set fid_ 0	Sets the number for data flow of udp. This number will be recorded to all packets which are sent from udp
$ns color 0 blue	Mark the color to discrete packet for showing result on NAM.

TCP Agent: To use TCP in simulation, the sender sets the Agent as TCP Agent while the receiver sets to TCPSink Agent. When receiving a packet, TCPSink Agent will reply an acknowledgment packet (ACK). Setting Agent for TCP is similar to UDP.

set tcp [new Agent/TCP] $ns attach-agent $n1 $tcp set sink [new Agent/TCPSink] $ns attach-agent $n3 $sink	# tcp and sink Agent are set for n1 and n3, respectively
$ns connect $tcp $sink	#declares the transmission between tcp and sink.
$tcp set fid_ 1	#sets the number for data flow of tcp. This number will be recorded to all packet which are sent from tcp
$ns color 1 red	#mark the color to discrete packet for showing result on Nam.

Step 4: Setting Application

In general, UDP Agent uses CBR Application while TCP Agent uses FTP Application.

set cbr [new Application/Traffic/CBR] $cbr attach-agent $udp
set ftp [new Application/FTP] $ftp attach-agent $tcp

Step 5: Setting time schedule for simulation

Time schedule of a simulation is set as below:

$ns at 1.0 "$cbr start" $ns at 3.5 "$cbr stop"	cbr transmits data from 1.0[sec] to 3.5[sec]
$ns at 1.5 "$ftp start" $ns at 3.0 "$ftp stop"	ftp transmits data from 1.5[sec] to 3.0[sec].

Step 6: Declare finish

After finish setting, declaration of finish is written at the end of file.

$ns at 4.0 "finish"	

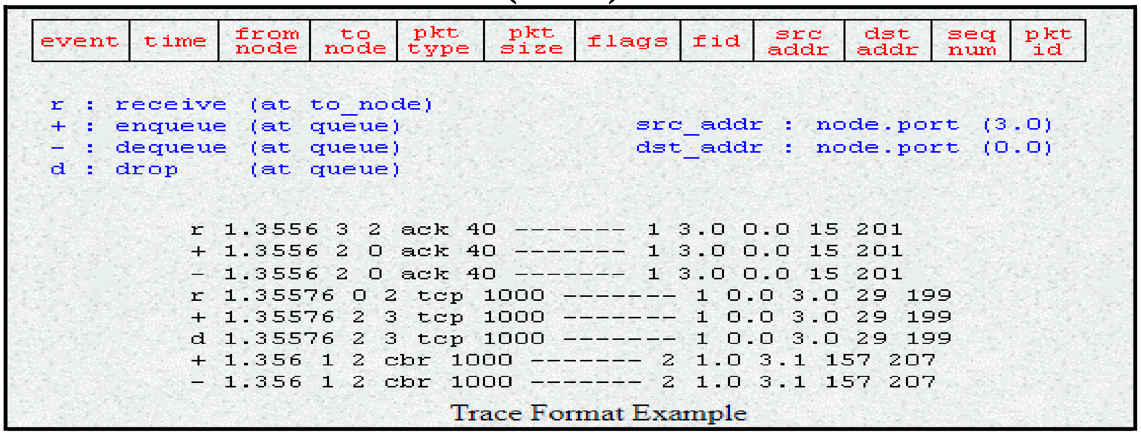
proc finish {} {
global ns file namfile tcpfile
$ns flush-trace
close $file
close $namfile
close $tcpfile
exit 0
}
	The finish function is used to output data file at the end of simulation.

Execute Simulation and start Nam

By executing below command line, simulation will be started and shows the animation of simulation

```
ns sample.tcl
nam out.nam
```

View trace file (out.tr)



event	time	from node	to node	pkt type	pkt size	flags	fid	src addr	dst addr	seq num	pkt id

```
r : receive  (at to_node)
+ : enqueue  (at queue)                src_addr : node.port (3.0)
- : dequeue  (at queue)                dst_addr : node.port (0.0)
d : drop     (at queue)

r 1.3556 3 2 ack 40 -------- 1 3.0 0.0 15 201
+ 1.3556 2 0 ack 40 -------- 1 3.0 0.0 15 201
- 1.3556 2 0 ack 40 -------- 1 3.0 0.0 15 201
r 1.35576 0 2 tcp 1000 -------- 1 0.0 3.0 29 199
+ 1.35576 2 3 tcp 1000 -------- 1 0.0 3.0 29 199
d 1.35576 2 3 tcp 1000 -------- 1 0.0 3.0 29 199
+ 1.356 1 2 cbr 1000 -------- 2 1.0 3.1 157 207
- 1.356 1 2 cbr 1000 -------- 2 1.0 3.1 157 207
```

Trace Format Example

Figure 4 . Details of Trace Window

1. The first field is the event type. It is given by one of four possible symbols r, +, -, d which correspond respectively to receive (at the output of the link), enqueued, dequeued and dropped.
2. The second field gives the time at which the event occurs.
3. Gives the input node of the link at which the event occurs.
4. Gives the output node of the link at which the event occurs.
5. Gives the packet type (eg CBR or TCP)
6. Gives the packet size
7. Some flags
8. This is the flow id (fid) of IPv6 that a user can set for each flow at the input OTcl

script one can further use this field for analysis purposes; it is also used when specifying stream color for the NAM display.

9. This is the source address given in the form of “node.port”.
10. This is the destination address, given in the same form.
11. This is the network layer protocol’s packet sequence number. Even though UDP implementations in a real network do not use sequence number, ns keeps track of UDP packet sequence number for analysis purposes
12. The last field shows the Unique id of the packet.

AWK file:

The basic function of awk is to search files for lines (or other units of text) that contain certain patterns. When a line matches one of the patterns, awk performs specified actions on that line. awk keeps processing input lines in this way until the end of the input files are reached. Programs in awk are different from programs in most other languages, because awk programs are **data-driven;** that is, we describe the data to work with, and then what to do when we find it. Most other languages are **procedural.** When working with procedural languages, it is usually much harder to clearly describe the data of our program will process. For this reason, awk programs are often refreshingly easy to both write and read. When we run awk, we can specify an awk **program** that tells awk what to do. The program consists of a series of **rules**. (It may also contain **function definitions**, an advanced feature which we will ignore for now. Each rule specifies one pattern to search for, and one action to perform when that pattern is found). Syntactically, a rule consists of a pattern followed by an action. The action is enclosed in curly braces to separate it from the pattern. Rules are usually separated by newlines. Therefore, an awk program looks like this:

***pattern* { *action* }** ***pattern* { *action* }**

Since we are dealing with column oriented data, AWK is probably the easiest tool we can use to format our data. AWK is a simple scripting language that scans through a file line by line. It allows to access any column in the current line by using special variables $1, $2, $3, etc. for the first, second and third columns. The definition of each column of the trace file is shown above, so we can use the AWK script to check the value of each column and collect the data we need.

The BEGIN and END sections are only executed once (before and after the file has been processed). The middle section is executed for each line of the file. The AWK script keeps three variables to store the throughput in Mb/s for flow 1, flow 2, and the total. For each line of the trace file, it checks to see if a TCP packet ($5 == “tcp”) is received ($1 == “r”) at node 3 ($4 == “3”), which is our destination node. If so, it increments the count for the appropriate flow, using the size of the particular packet (in bytes) from column 6. After each second, it prints the total while converting bytes to Mb.

BEGIN { print "START" } { print }END { print "STOP" }

To run awk script

awk -f <filename.awk> <input_file> <output_file>

XGRAPH:

- Plotting purposes.
- Comes together with NS2 installation package.
- Running Xgraph

Xgraph <inputfile$_1$>...<inputfile$_n$> -bg <color> -t <graph_title> -x <xtitle> -y <ytitle>

NS Simulation Script

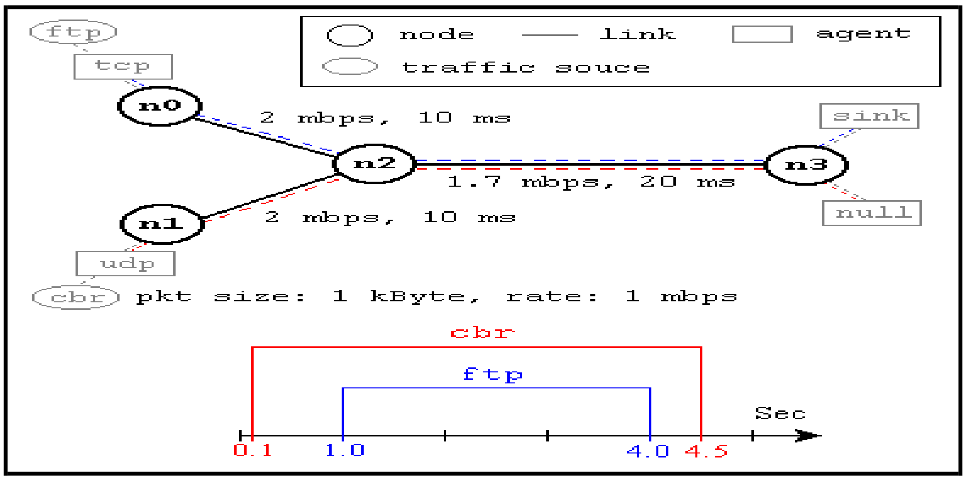


Fig 5. A Simple Network Topology and Simulation Scenario

This network consists of 4 nodes (n0, n1, n2, n3) as shown in Fig. 5. The duplex links between n0 and n2, and n1 and n2 have 2 Mbps of bandwidth and 10 ms of delay. The duplex link between n2 and n3 has 1.7 Mbps of bandwidth and 20 ms of delay. Each node uses a DropTail queue, of which the maximum size is 10. A "tcp" agent is attached to n0, and a connection is established to a tcp "sink" agent attached to n3. As default, the maximum size of a packet that a "tcp" agent can generate is 1KByte. A tcp "sink" agent generates and sends ACK packets to the sender (tcp agent) and frees the received packets. A "udp" agent that is attached to n1 is connected to a "null" agent attached to n3. A "null" agent just frees the packets received. A "ftp" and a "cbr" traffic generator are attached to "tcp" and "udp" agents respectively, and the "cbr" is configured to generate 1KByte packets at the rate of 1 Mbps.

The "cbr" is set to start at 0.1 sec and stop at 4.5 sec, and "ftp" is set to start at 1.0 sec and stop at 4.0 sec.

An Example Simulation Script

```
 #Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish
{} {   global
ns nf
 $ns flush-trace

 #Close the NAM trace file
close $nf

#Execute NAM on the trace file
exec nam out.nam
& exit 0
}

#Create four
nodes set n0 [$ns
node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#Create links between the nodes
```

```
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail

#Set Queue Size of link (n2-n3) to 10
$ns queue-limit $n2 $n3 10

#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right

#Monitor the queue for link (n2-n3). (for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5
#Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ftp set type_ FTP

#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1
$udp set null [new
Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null
$udp set fid_ 2

#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set type_ CBR
$cbr set packet_size_ 1000
$cbr set rate_ 1mb
$cbr set random_ false

#Schedule events for the CBR and FTP agents
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
```

```
$ns at 4.5 "$cbr stop"

#Call the finish procedure after 5 seconds of simulation time
$ns at 5.0 "finish"

#Print CBR packet size and interval
puts "CBR packet size = [$cbr set packet_size_]"
puts "CBR interval = [$cbr set interval_]"

#Run the simulation
$ns run
```

Program No. 1: Demonstrate ring topology and bus topology creation in NS2

```
#Ring Topology
#Create a simulator object
set ns [new Simulator]

#Open the nam trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
   global ns nf
   $ns flush-trace
   #Close the trace file
   close $nf
   #Executenam on the trace file
   exec nam out.nam &
   exit 0
}

#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
$ns duplex-link $n2 $n3 1Mb 10ms DropTail
$ns duplex-link $n3 $n4 1Mb 10ms DropTail
$ns duplex-link $n4 $n5 1Mb 10ms DropTail
$ns duplex-link $n5 $n0 1Mb 10ms DropTail

#Create a TCP agent and attach it to node n0
set tcp0 [new Agent/TCP]
$tcp0 set class_ 1
$ns attach-agent $n1 $tcp0
#Create a TCP Sink agent (a traffic sink) for TCP and
attach it to node n3
set sink0 [new Agent/TCPSink]
$ns attach-agent $n3 $sink0
#Connect the traffic sources with the traffic sink
$ns connect $tcp0 $sink0
```

```
# Create a CBR traffic source and attach it to tcp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.01
$cbr0 attach-agent $tcp0

#Schedule events for the CBR agents
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"


#Call the finish procedure after 5 seconds of simulation
time
$ns at 5.0 "finish"

#Run the simulation
$ns run
```

#Bus Topology

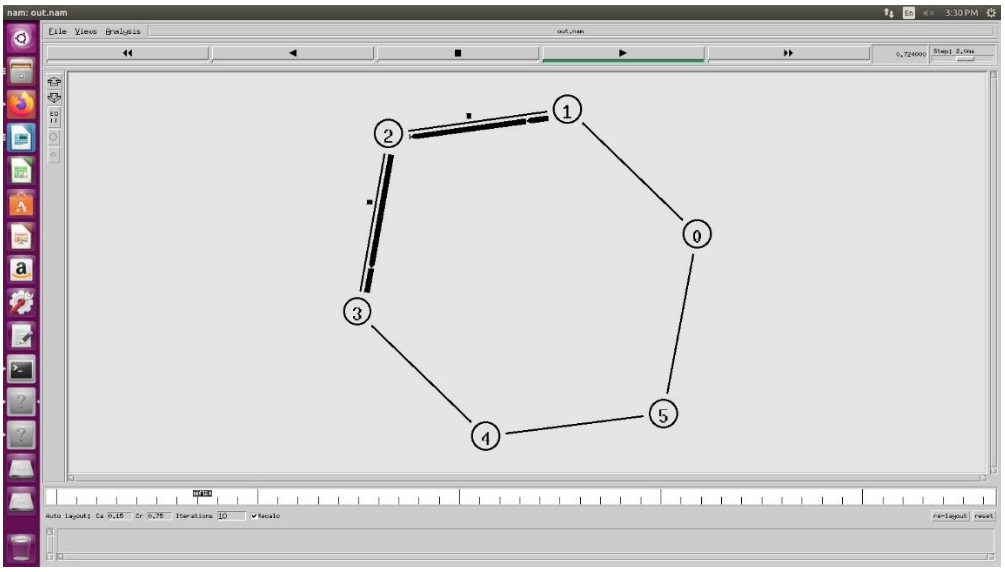
```
#Create a simulator object
set ns [new Simulator]

#Open the nam trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
   global ns nf
   $ns flush-trace
   #Close the trace file
   close $nf
   #Executenam on the trace file
   exec nam out.nam &
   exit 0
}

#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]

#CreateLanbetween the nodes
set lan0 [$ns newLan "$n0 $n1 $n2 $n3 $n4" 0.5Mb 40ms
LL Queue/DropTail MAC/Csma/Cd Channel]
```

```
#Create a TCP agent and attach it to node n0
set tcp0 [new Agent/TCP]
$tcp0 set class_ 1
$ns attach-agent $n1 $tcp0
#Create a TCP Sink agent (a traffic sink) for TCP and
attach it to node n3
set sink0 [new Agent/TCPSink]
$ns attach-agent $n3 $sink0
#Connect the traffic sources with the traffic sink
$ns connect $tcp0 $sink0

# Create a CBR traffic source and attach it to tcp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.01
$cbr0 attach-agent $tcp0

#Schedule events for the CBR agents
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"


#Call the finish procedure after 5 seconds of simulation
time
$ns at 5.0 "finish"

#Run the simulation
$ns run
```

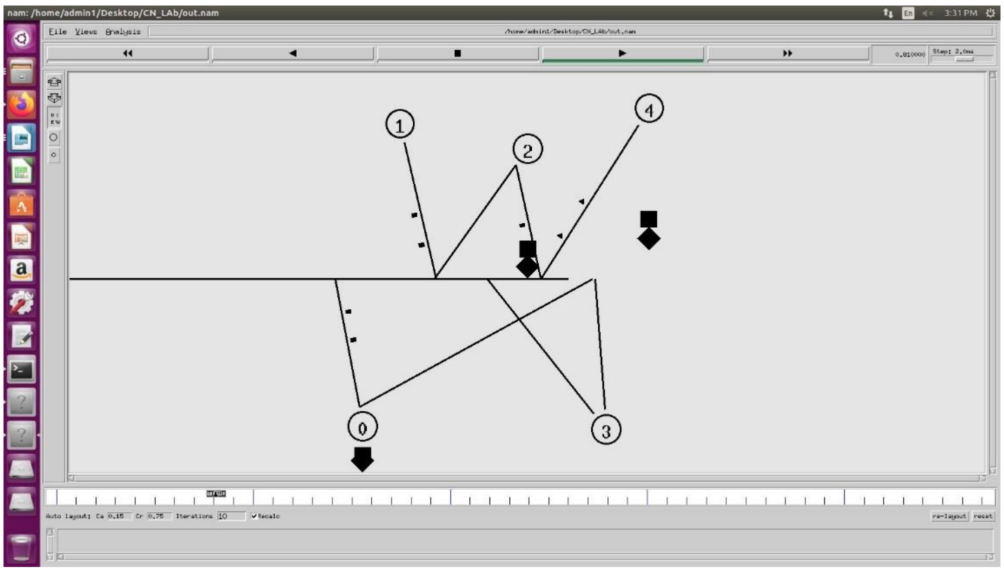
Steps for execution:

- *Open gedit and type program. Program name should have the extension ". tcl "*
  [root@localhost ~] gedit topo1.tcl
- *Save the program.*
- *Open gedit and type **awk** program. Program name should have the extension "**. awk** "*
  [root@localhost ~] gedit topo1.awk
- *Save the program.*
- *Run the simulation program **[root@localhost~] ns topo1.tcl***
- *Here **"ns"** indicates network simulator. We get the topology shown in the snapshot.*
- *Now press the play button in the simulation window and the simulation will begin.*

Snapshot 1



Snapshot 2


nam: /home/admin1/Desktop/CN_LAb/out.nam
File Views Analysis
/home/admin1/Desktop/CN_LAb/out.nam
0.810000
Step: 2.0ms
re-layout
reset
0
1
2
3
4

Program No. 2: Implement three nodes point – to – point network with duplex links between them. Set the queue size, vary the bandwidth and find the number of packets dropped.

Theory:

- Create a simulator object.
- We open a file for writing that is going to be used for the trace data.
- We now attach the agent to the nodes.
- Now we attach the application to run on top of these nodes
- We now connect the agent and the application for its working
- Set the simulation time
- The next step is to add a 'finish' procedure that closes the trace file and starts nam.

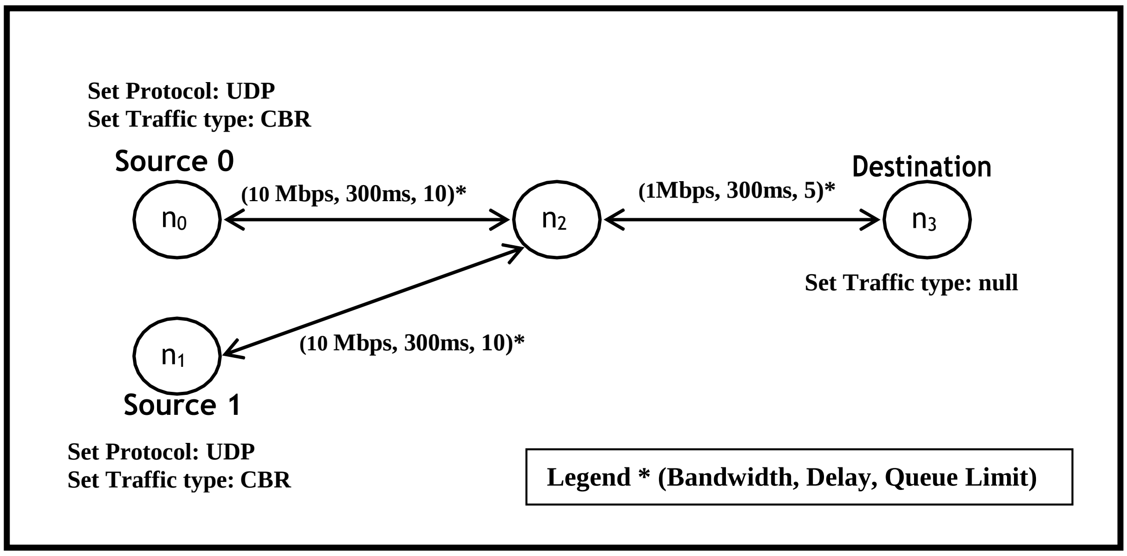


Figure 2 Sample Network for program 2

Program:

```
set ns [ new Simulator ]
set tf [ open lab1.tr w ]
$ns trace-all $tf
set nf [ open lab1.nam w ]
$ns namtrace-all $nf
# The below code is used to create the nodes.
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
#This is used to give color to the packets.
$ns color 1 "red"
$ns color 2 "blue"
$n0 label "Source/udp0"
```

```
$n1 label "Source/udp1"
$n2 label "Router"
$n3 label "Destination/Null"
#Vary the below Bandwidth and see the number of packets
dropped.
$ns duplex-link $n0 $n2 10Mb 300ms DropTail
$ns duplex-link $n1 $n2 10Mb 300ms DropTail
$ns duplex-link $n2 $n3 1Mb 300ms DropTail
#The below code is used to set the queue size b/w the
nodes
$ns set queue-limit $n0 $n2 10
$ns set queue-limit $n1 $n2 10
$ns set queue-limit $n2 $n3 5
#The below code is used to attach an UDP agent to n0,
UDP agent to n1 and null agent to n3.
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
set null [new Agent/Null]
$ns attach-agent $n3 $null
set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
#The below code sets the udp0 packets to red and udp1
packets to blue color
$udp0 set class_ 1
$udp1 set class_ 2
#The below code is used to connect the agents.
$ns connect $udp0 $null
$ns connect $udp1 $null
#The below code is used to set the packet size to 500
$cbr1 set packetSize_ 500Mb
#The below code is used to set the interval of the packets,
i.e., Data rate of the packets.
#if the data rate is high then packets drops are high.
$cbr1 set interval_ 0.005
proc finish { } {
global ns nf tf
$ns flush-trace
exec nam lab1.nam &
close $tf
close $nf
exit 0
}
$ns at 0.1 "$cbr0 start"
$ns at 0.1 "$cbr1 start"
$ns at 10.0 "finish"
```

```
$ns run
```

AWK file:

(Open a new editor using gedit command and write awk file and save with ".awk" extension)

```
BEGIN{
#include<stdio.
h>
count=0;
}
{
if($1=="d") #d stands for the packets
drops.
count++
}
EN
D{
printf("The Total no of Packets Dropped due to Congestion :%d\n\n", count)
}
```

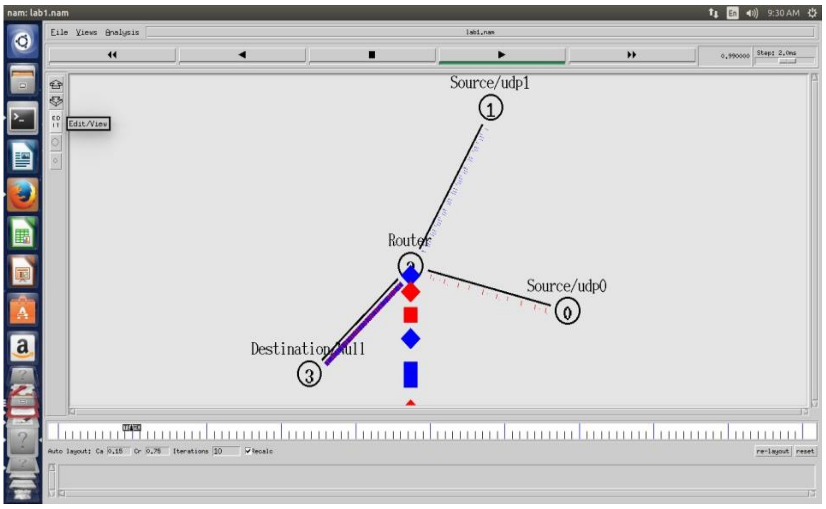
Steps for execution:

- *Open gedit and type program. Program name should have the extension ". tcl "*
  [root@localhost ~] gedit lab1.tcl
- *Save the program.*
- *Open gedit and type* ***awk*** *program. Program name should have the extension "****. awk*** *"*
  [root@localhost ~] gedit lab1.awk
- *Save the program.*
- *Run the simulation program* ***[root@localhost~] ns lab1.tcl***
- *Here* ***"ns"*** *indicates network simulator. We get the topology shown in the snapshot.*
- *Now press the play button in the simulation window and the simulation will begin.*
- *After simulation is completed run* ***awk file*** *to see the output,*
  [root@localhost~] awk –f lab1.awk lab1.tr
- *To see the trace file contents open the file as ,* ***[root@localhost~] gedit lab1.tr***

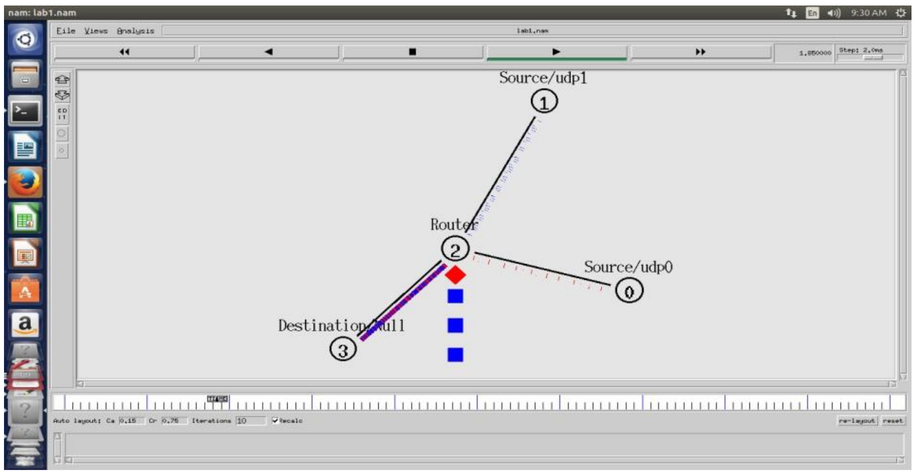
Output:

```
The Total no of packets Dropped due to congestion: 456
```

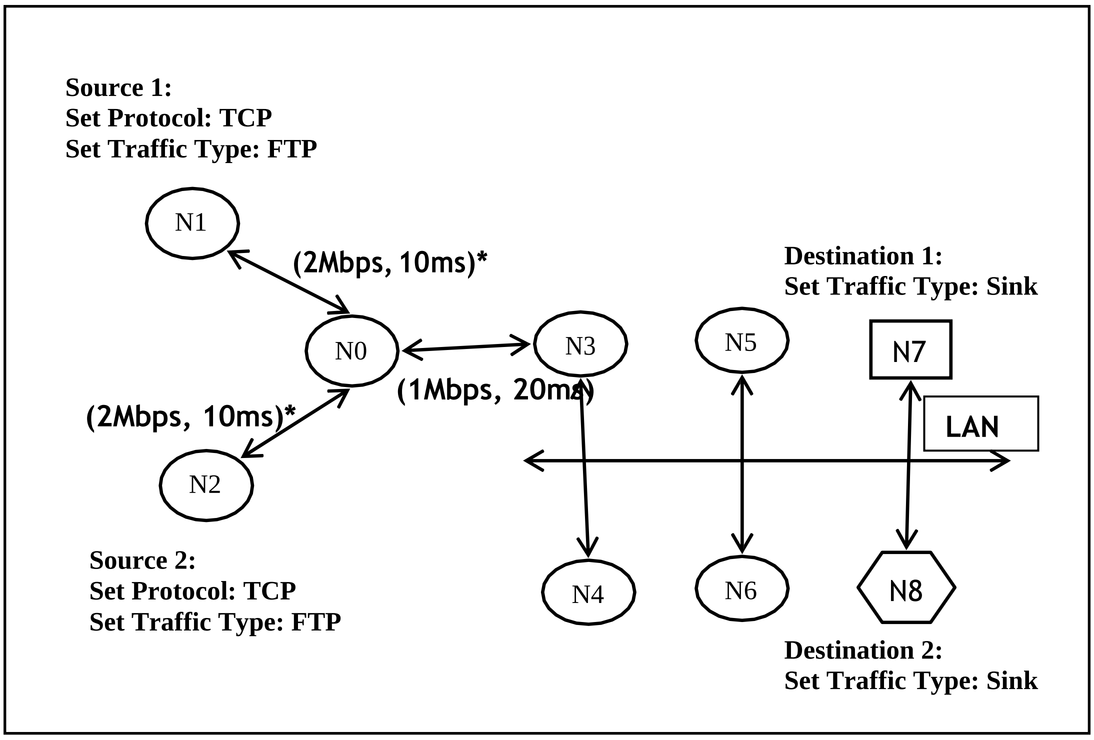
Snapshot 1:



Snapshot 2



Program No. 3: Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination.



Program:

```
set ns [new Simulator]
set nf [open lab3.nam w]
$ns namtrace-all $nf
set nd [open lab3.tr w]
$ns trace-all $nd
$ns color 1 Blue
$ns color 2 Red
proc finish { } {
global ns nf nd
$ns flush-trace
close $nf
close $nd
exec nam lab3.nam &
exit 0
}
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]
set n7 [$ns node]
set n8 [$ns node]
$n7 shape box
$n7 color Blue
$n8 shape hexagon
$n8 color Red
$ns duplex-link $n1 $n0 2Mb 10ms DropTail
$ns duplex-link $n2 $n0 2Mb 10ms DropTail
$ns duplex-link $n0 $n3 1Mb 20ms DropTail
$ns make-lan "$n3 $n4 $n5 $n6 $n7 $n8" 512Kb 40ms LL
Queue/DropTail Mac/802_3
$ns duplex-link-op $n1 $n0 orient right-down
$ns duplex-link-op $n2 $n0 orient right-up
$ns duplex-link-op $n0 $n3 orient right
$ns queue-limit $n0 $n3 20
set tcp1 [new Agent/TCP/Vegas]
$ns attach-agent $n1 $tcp1
set sink1 [new Agent/TCPSink]
$ns attach-agent $n7 $sink1
$ns connect $tcp1 $sink1
$tcp1 set class_ 1
$tcp1 set packetSize_ 55
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
set tfile [open cwnd.tr w]
$tcp1 attach $tfile
$tcp1 trace cwnd_
set tcp2 [new Agent/TCP/Reno]
$ns attach-agent $n2 $tcp2
set sink2 [new Agent/TCPSink]
$ns attach-agent $n8 $sink2
$ns connect $tcp2 $sink2
$tcp2 set class_ 2
$tcp2 set packetSize_ 55
set ftp2 [new Application/FTP]
$ftp2 attach-agent $tcp2
set tfile2 [open cwnd2.tr w]
$tcp2 attach $tfile2
$tcp2 trace cwnd_
$ns at 0.5 "$ftp1 start"
$ns at 1.0 "$ftp2 start"
$ns at 5.0 "$ftp2 stop"
$ns at 5.0 "$ftp1 stop"
$ns at 5.5 "finish"
$ns run
```

<u>AWK File:</u>

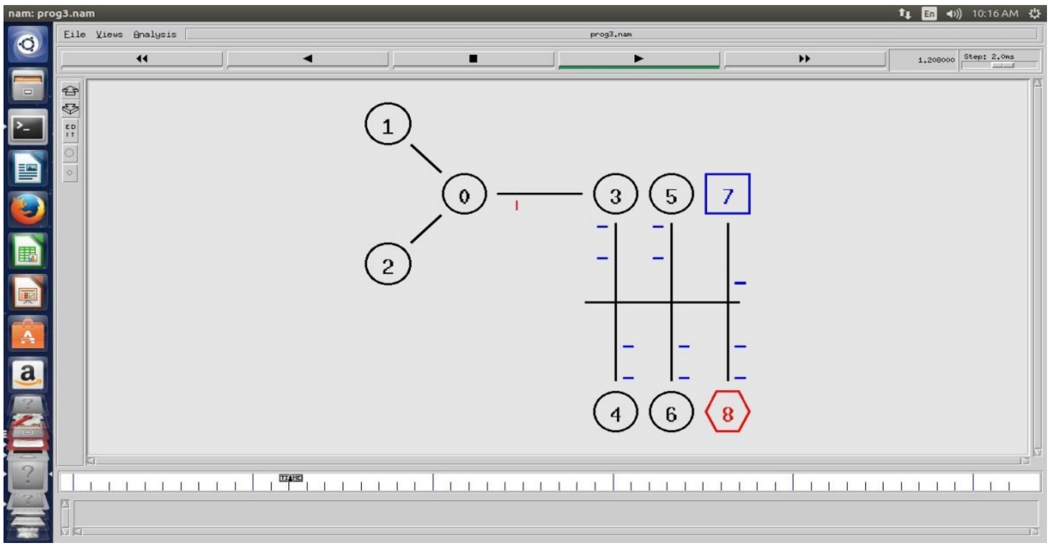
```
BEGIN {
}
```

```
{
if($6=="cwnd_") {
printf("%f\t%f\n",$1,$7);
}
}
END {
}
```

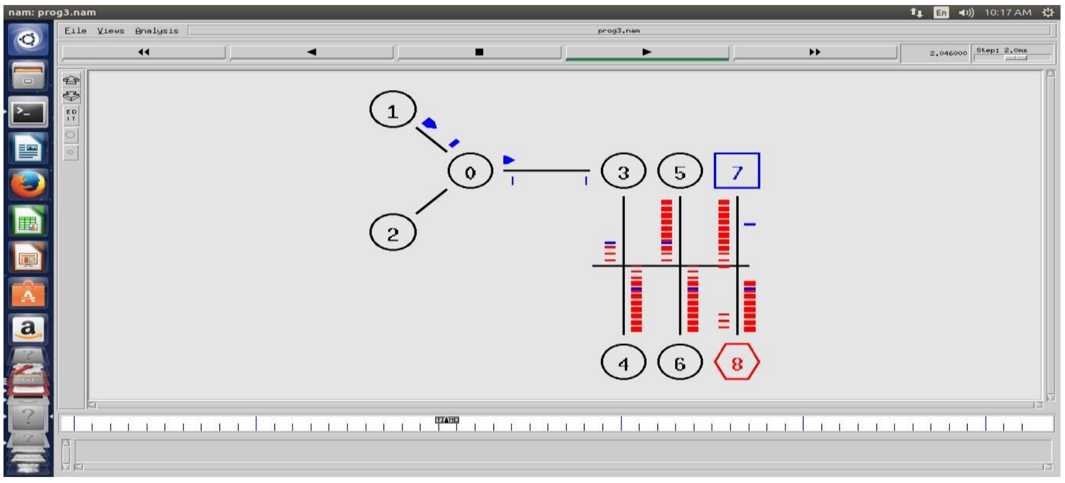
Output Commands:

```
[root@localhost ~]# ns lab3.tcl
[root@localhost ~]# awk -f lab3.awk file1.tr>tcp1
[root@localhost ~]# awk -f lab3.awk file2.tr>tcp2
[root@localhost ~]# xgraph -x “time” -y “convalue” tcp1 tcp2
```

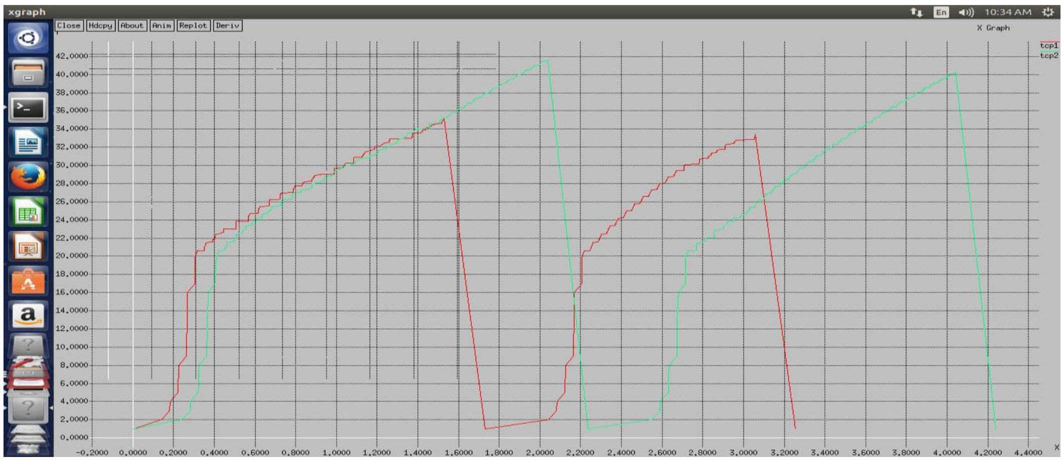
Snapshot 1:



Snapshot 2:



Snapshot 3:


xgraph
Close
Hdcpy
About
Anim
Replot
Deriv
X Graph
tcp1
tcp2
10:34 AM

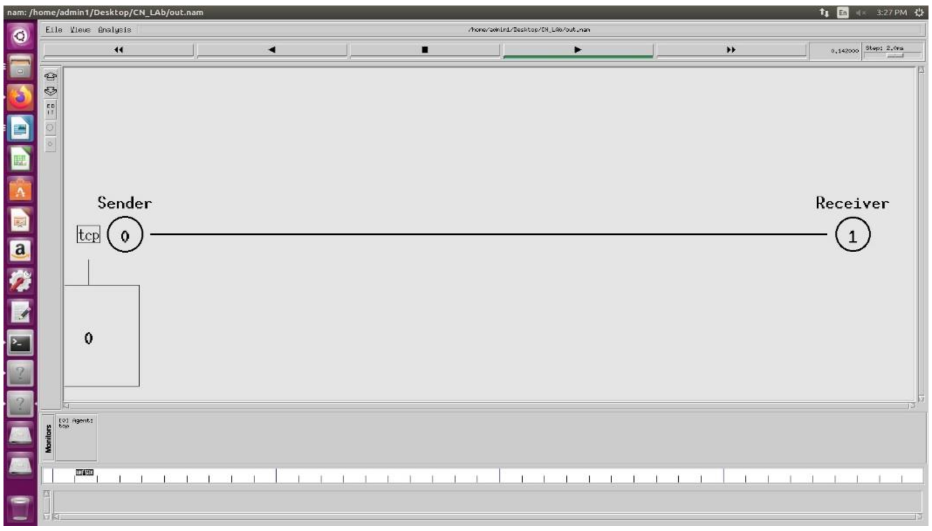
Program No. 4: Write a NS2 script to implement the operation of Stop and Wait Protocol.

Program:

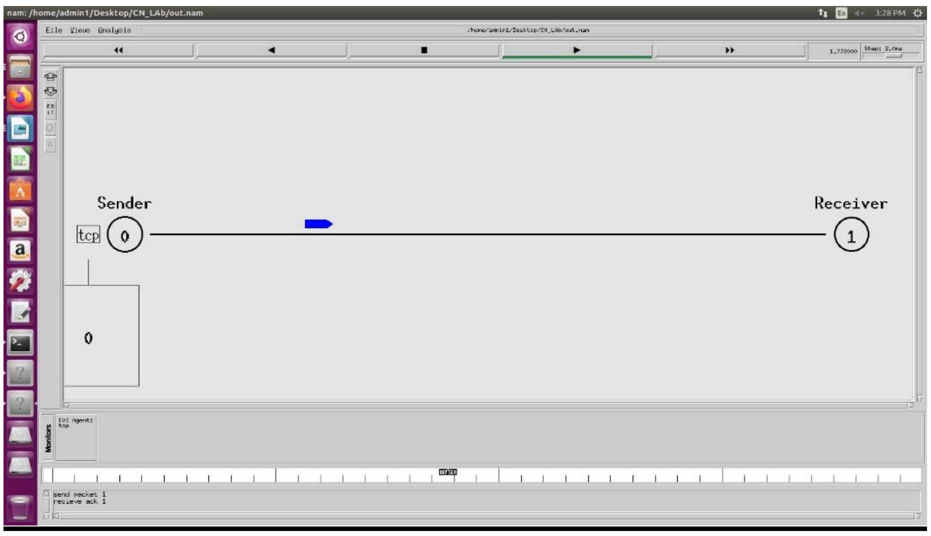
```
set ns [new Simulator]
$ns color 1 Blue
# set nam output file
set nf [open out.nam w]
$ns namtrace-all $nf

# destructor
proc finish {} {
    global ns nf
    $ns flush-trace
    close $nf
    exec nam out.nam &
    exit 0
}

# create two new nodes and create labels for them
set n0 [$ns node]
set n1 [$ns node]
$ns at 0.0 "$n0 label \" Sender \" "
$ns at 0.0 "$n1 label \"Receiver\" "

# set up a new duplex link
$ns duplex-link $n0 $n1 1Mb 200ms DropTail
$ns duplex-link-op $n0 $n1 orient right

# create a new TCP agent
set tcp [new Agent/TCP]
# attach the agent to first node
$ns attach-agent $n0 $tcp
$tcp set fid_ 1
$tcp set window_ 1
$tcp set maxcwnd_ 1
$ns add-agent-trace $tcp tcp
$ns monitor-agent-trace $tcp
set tcpsink [new Agent/TCPSink]
$ns attach-agent $n1 $tcpsink

$ns connect $tcp $tcpsink
set ftp [new Application/FTP]
$ftp attach-agent $tcp

$ns at 0.5 "$ftp start"
$ns at 3.0 "$ns detach-agent $n0 $tcp ; $ns detach-agent $n1
$tcpsink "
$ns at 1.0 "$ns trace-annotate \"send packet 1\""
$ns at 1.4 "$ns trace-annotate \"recieve ack 1\""
$ns at 2.0 "$ns trace-annotate \"send packet 2\""
```

```
$ns at 2.5 "$ns trace-annotate \"receive ack 2\""
$ns at 3.2 "$ns trace-annotate \"send packet 3\""
$ns at 3.5 "$ns trace-annotate \"receive ack 3\""
$ns at 3.8 "$ns trace-annotate \"send packet 4\""
$ns at 4.0 "finish"
$ns run
```

Snapshot 1



Snapshot 2



```
send packet 1
recieve ack 1
```

Program No. 5: Implement simple ESS and with transmitting nodes in wire-less LAN by simulation and determine the performance with respect to transmission of packets.

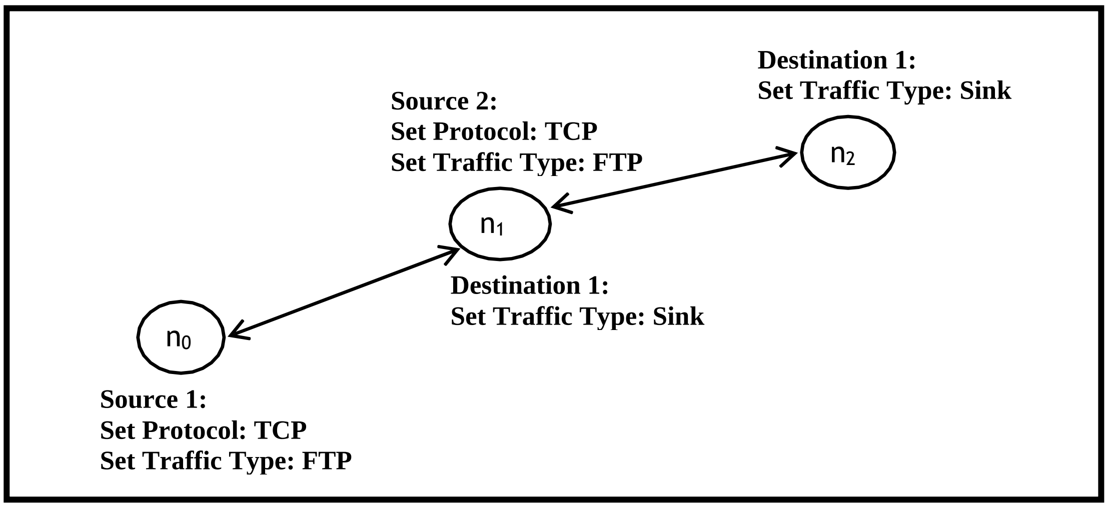


Figure 5: Sample Network for Program 5

Program:

```
set ns [new Simulator]
set val(chan) Channel/WirelessChannel;
set val(prop) Propagation/TwoRayGround;
set val(netif) Phy/WirelessPhy;
set val(mac) Mac/802_11;
set val(ifq) Queue/DropTail/PriQueue;
set val(ll) LL;
set val(ant) Antenna/OmniAntenna;
set val(ifqlen) 50;
set val(nn) 2;
set val(rp) DSDV;
set val(x) 1000.0;
set val(y) 1000.0;
# channel type
# radio-propagation model
# network interface type
# MAC type
# interface queue type
# link layer type
# antenna model
# max packet in ifq
# number of mobilenodes
# routing protocol
set tf [open lab4.tr w]
$ns trace-all $tf
set topo [new Topography]
$topo load_flatgrid 1000 1000
set nf [open lab4.nam w]
$ns namtrace-all-wireless $nf 1000 1000
```

```
$ns node-config -adhocRouting $val(rp) \
-llType $val(ll) \
-macType $val(mac) \
-ifqType $val(ifq) \
-ifqLen $val(ifqlen) \
-antType $val(ant) \
-propType $val(prop) \
-phyType $val(netif) \
-channelType $val(chan) \
-topoInstance $topo \
-agentTrace ON \
-routerTrace ON \
-macTrace ON
create-god $val(nn)
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
$n0 label "tcp0"
$n1 label "sink1/tcp1"
$n2 label "sink2"
$n0 set X_ 250
$n0 set Y_ 250
$n0 set Z_ 0
$n1 set X_ 300
$n1 set Y_ 300
$n1 set Z_ 0
$n2 set X_ 600
$n2 set Y_ 600
$n2 set Z_ 0
$ns at 0.1 "$n0 setdest 250 250 15"
$ns at 0.1 "$n1 setdest 300 300 25"
$ns at 0.1 "$n2 setdest 600 600 25"
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
set sink1 [new Agent/TCPSink]
$ns attach-agent $n1 $sink1
$ns connect $tcp0 $sink1
set tcp1 [new Agent/TCP]
$ns attach-agent $n1 $tcp1
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
set sink2 [new Agent/TCPSink]
$ns attach-agent $n2 $sink2
$ns connect $tcp1 $sink2
$ns at 5 "$ftp0 start"
$ns at 5 "$ftp1 start"
#The below code is used to provide the node movements.
```

```
$ns at 100 "$n1 setdest 550 550 15"
$ns at 190 "$n1 setdest 70 70 15"
proc finish {} {
global ns nf tf
$ns flush-trace
exec nam lab4.nam &
close $tf
exit 0
}
$ns at 250 "finish"
$ns run
```

AWK FILE:

```
BEGIN {
#include<stdio.h>
count1=count2=pack1=pack2= time1=time2=0
}
{
if($1 == "r"&&$3 == "_1_"&&$4 == "AGT")
{
count1++
pack1=pack1+$8
time1=$2
}
if($1=="r"&&$3=="_2_"&&$4=="AGT")
{
count2++
pack2 = pack2+$8
time2=$2
}
}
END{
printf("The Throughput from n0 to n1: %f Mbps\n",((count1 * pack1 *8)/(time1*1000000)));
printf("The Throughput from n1 to n2: %f Mbps", ((count2 * pack2 * 8) /(time2*1000000)));
}
```

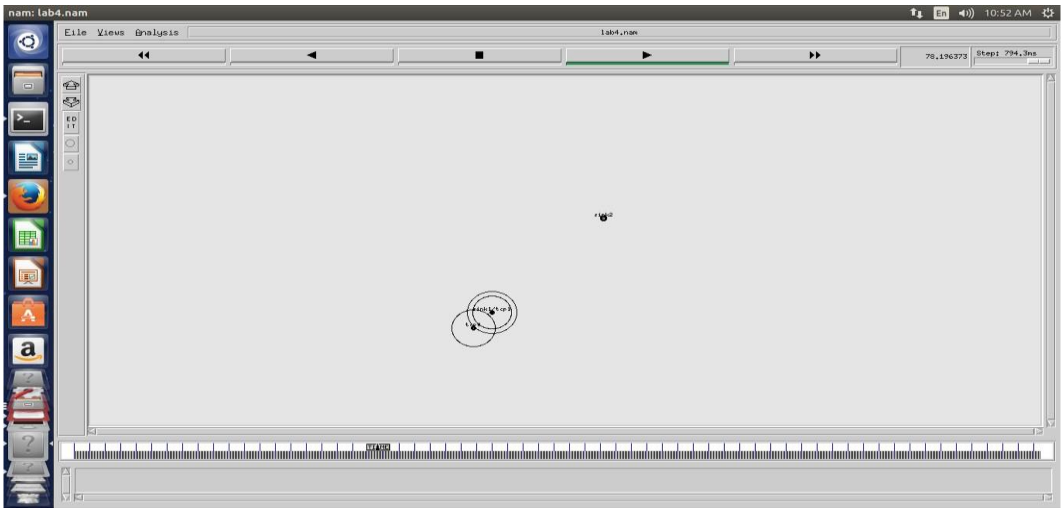
Output Commands:

```
[root@localhost ~]# ns prg4.tcl
[root@localhost ~]# awk –f prg4.awk lab4.tr
```

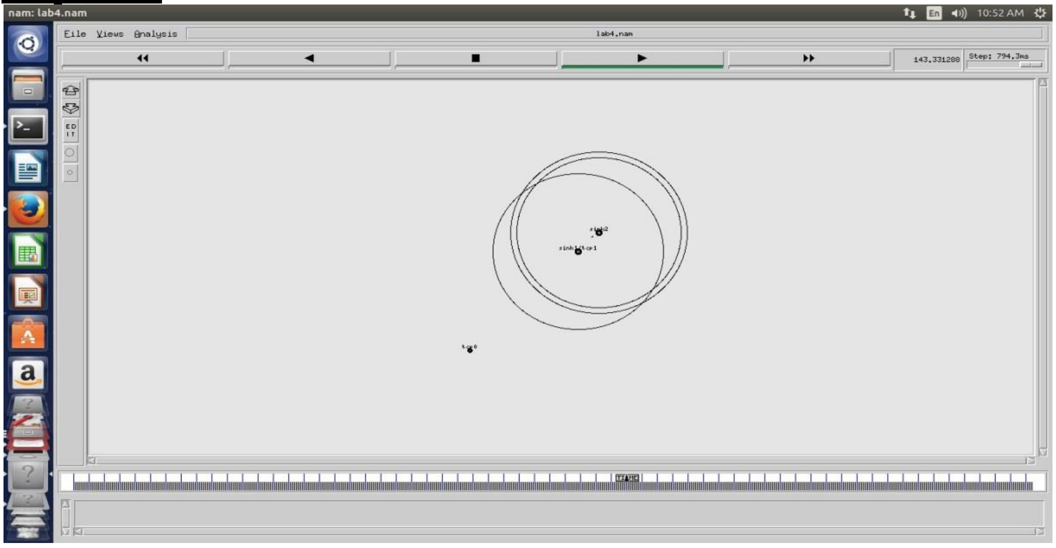
Output:

```
The Throughput from n0 to n1: 5444Mbps
The Throughput from n1 to n2: 345Mbps
```

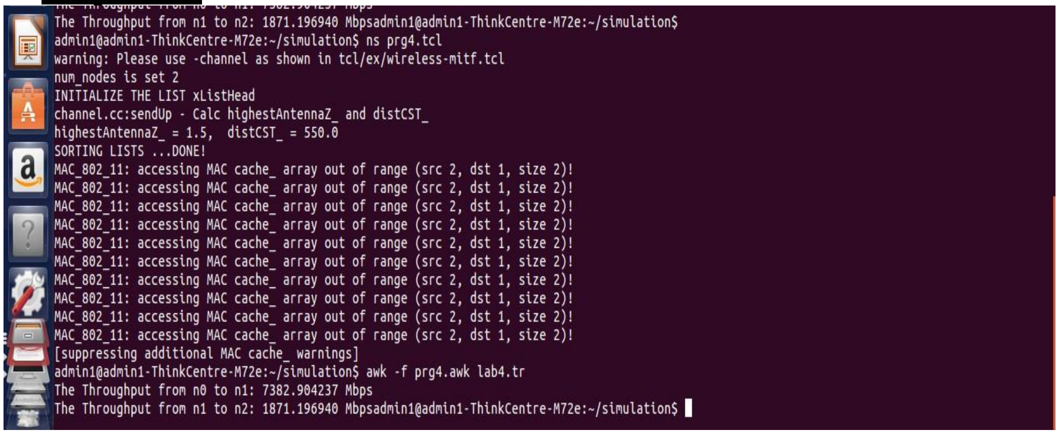
Snapshot 1:



Snapshot 2:



Snapshot 3:



```
The Throughput from n1 to n2: 1871.196940 Mbpsadmin1@admin1-ThinkCentre-M72e:~/simulation$
admin1@admin1-ThinkCentre-M72e:~/simulation$ ns prg4.tcl
warning: Please use -channel as shown in tcl/ex/wireless-mitf.tcl
num_nodes is set 2
INITIALIZE THE LIST xListHead
channel.cc:sendUp - Calc highestAntennaZ_ and distCST_
highestAntennaZ_ = 1.5,  distCST_ = 550.0
SORTING LISTS ...DONE!
MAC_802_11: accessing MAC cache_ array out of range (src 2, dst 1, size 2)!
MAC_802_11: accessing MAC cache_ array out of range (src 2, dst 1, size 2)!
MAC_802_11: accessing MAC cache_ array out of range (src 2, dst 1, size 2)!
MAC_802_11: accessing MAC cache_ array out of range (src 2, dst 1, size 2)!
MAC_802_11: accessing MAC cache_ array out of range (src 2, dst 1, size 2)!
MAC_802_11: accessing MAC cache_ array out of range (src 2, dst 1, size 2)!
MAC_802_11: accessing MAC cache_ array out of range (src 2, dst 1, size 2)!
MAC_802_11: accessing MAC cache_ array out of range (src 2, dst 1, size 2)!
MAC_802_11: accessing MAC cache_ array out of range (src 2, dst 1, size 2)!
MAC_802_11: accessing MAC cache_ array out of range (src 2, dst 1, size 2)!
[suppressing additional MAC cache_ warnings]
admin1@admin1-ThinkCentre-M72e:~/simulation$ awk -f prg4.awk lab4.tr
The Throughput from n0 to n1: 7382.904237 Mbps
The Throughput from n1 to n2: 1871.196940 Mbpsadmin1@admin1-ThinkCentre-M72e:~/simulation$
```

<u>Sample Java Program to find the IP ADDRESS</u>

AIM: To write a java program to find the IP address of the system.

ALGORITHM:

```
1. Start
2. Declare a variable 'ip' as a static InetAddress.
3. Using the function getLocalHost() to find the address of the system.
4. Get the name of the system by using the getHostName() function.
5. By specifying the system name,find out the IP address of the system using
the function getByName().
6. Stop.
```

SOURCE CODE:

```
import java.io.*;
import java.net.*;
class address
{
public InetAddress ip;
public static void main(String args[])throws UnknownHostException
{
InetAddress ip=InetAddress.getLocalHost();
System.out.println("\n IP address is :"+ip);
String s1=ip.getHostName();
System.out.println("system number is:"+s1);
InetAddress ip1=InetAddress.getByName("system 10");
System.out.println("\n name of other system is :"+ip1);
}
}
```

<u>Output:</u>

```
IP address is: LAB1-89/192.168.6.189
system number is: LAB1-89
```

<u>Sample Java Program to display Fibonacci Series</u>

```
class FibonacciExample1
{
public static void main(String args[])
{
 int n1=0,n2=1,n3,i,count=10;
 System.out.print(n1+" "+n2);//printing 0 and 1
 for(i=2;i<count;++i)//loop starts from 2 because 0 and 1 are already printed
 {
  n3=n1+n2;
 System.out.print(" "+n3);
 n1=n2;
 n2=n3;
}
}
}
```

Output:

```
0 1 1 2 3 5 8
```

Program No. 6: Write a Java program for error detecting code using CRC-CCITT (16- bits).

The idea behind CRC calculation is to look at the data as one large binary number. This number is divided by generator and the remainder of the calculation is called the CRC.

All of the CRC formulas you will encounter are based on modulo-2 binary division where we ignore carry bits and in effect the subtraction will be equal to an *exclusive or* operation. Though some differences exist in the specifics across different CRC formulas, the basic mathematical process is always the same:

- The message bits are appended with *k-1* zero bits; this *augmented message* is the dividend
- A predetermined *k*-bit binary sequence, called the *generator polynomial*, is the divisor
- The checksum is the *c*-bit remainder that results from the division operation

Table 1 lists some of the most commonly used generator polynomials for 16- and 32-bit CRCs. Remember that the width of the divisor is always one bit wider than the remainder. So, for example, you'd use a 17-bit generator polynomial whenever a 16-bit checksum is required.

	CRC-CCITT	**CRC-16**	**CRC-32**
Checksum Width	16 bits	16 bits	32 bits
Generator Polynomial	$x^{16}+x^{12}+x^{5}+1$	$x^{16}+x^{15}+x^{2}+1$	$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1$
Generator Polynomial in bits	10001000000100001	11000000000000101	100000100110000010001110110110111

Table 1. International Standard CRC Polynomials

Error detection with CRC

Consider a message represented by the polynomial M(x)

Consider a *generating polynomial* G(x).

This is used to generate a CRC = C(x) to be appended to M(x).

Note this G(x) is prime.

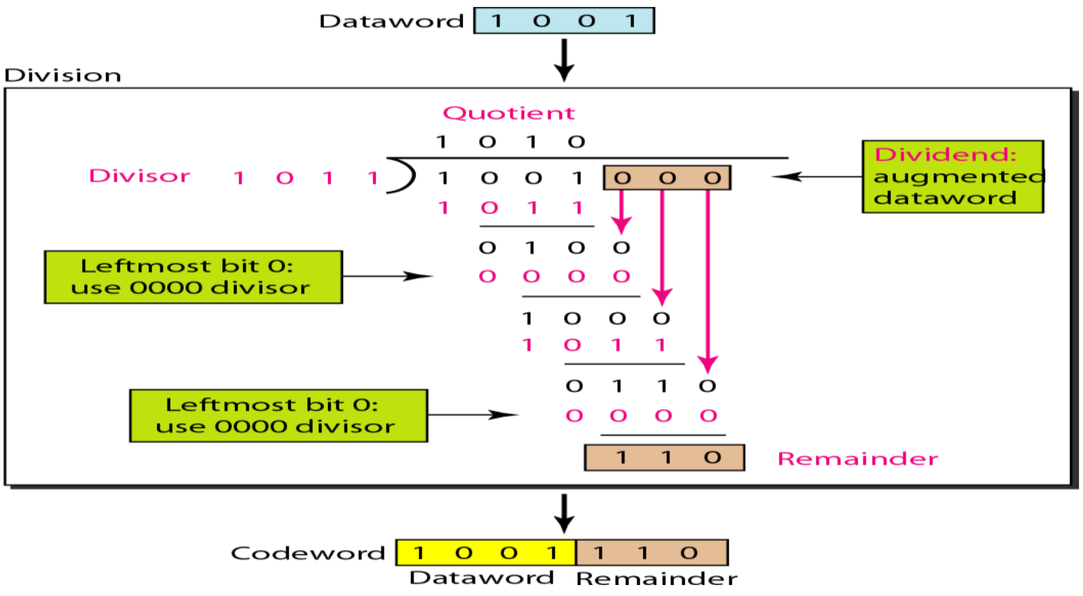
Steps:

1. Multiply M(x) by highest power in G(x). i.e. Add So much zeros to M(x).
2. Divide the result by G(x). The remainder = C(x).

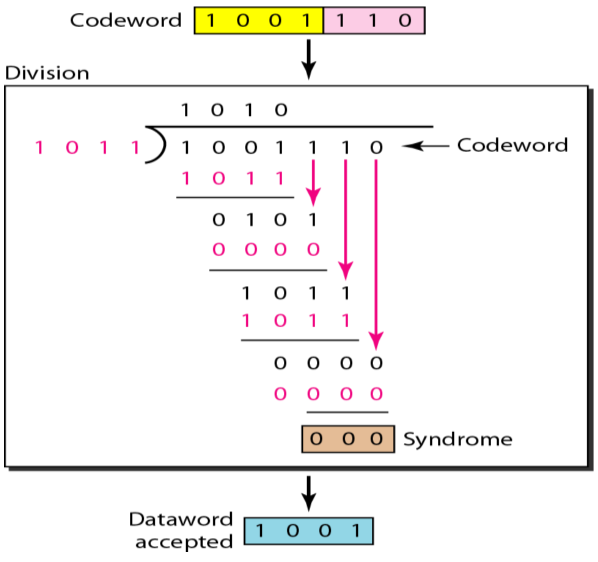
   Special case: This won't work if bitstring =all zeros. We don't allow such an M(x).But M(x) bitstring = 1 will work, for example. Can divide 1101 into 1000.
3. Transmit: T(x) = M(x) + C(x)
4. Receiver end: Receive T(x). Divide by G(x), should have remainder 0.

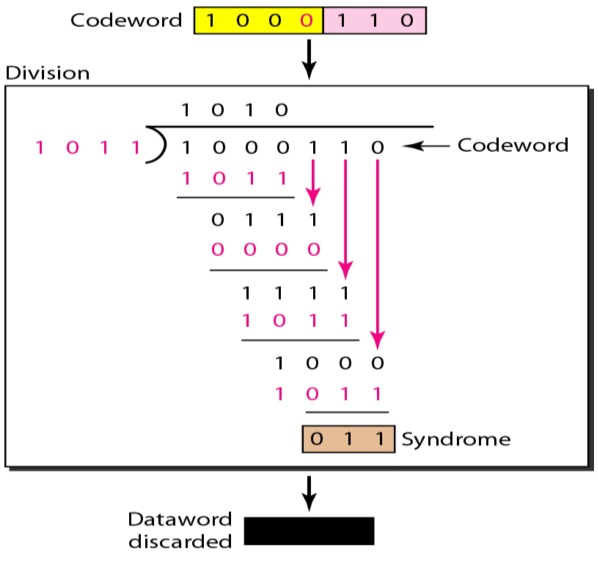
Note if G(x) has order n - highest power is x^n, then G(x) will cover (n+1) bits and the remainder will cover n bits. i.e. Add n bits (Zeros) to message.

Division at the Sender site:



Division at the Receiver site for two cases:





Some CRC polynomials that are actually used

- CRC-8:
  x^8+x^2+x+1
  - Used in: 802.16 (along with error *correction*).
- CRC-12: $x^{12} + x^{11} + x^3 + x^2 + x + 1$
- CRC-CCITT (Cyclic Redundancy Check- Consultative Committee for International Telegraphy and Telephony): $x^{16}+x^{12}+x^5+1$
  - Used in: HDLC, SDLC, PPP default
- IBM-CRC-16 (ANSI):
  $x^{16}+x^{15}+x^2+1$
- CRC-32 used in IEEE 802.3 Ethernet Standard:
  $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$

Program:

```
import java.util.Scanner;
class CRC
{
   static String datastream;
   static String generator= "10001000000100001";
   public static void main(String args[]){
   Scanner sc = new Scanner(System.in);
   System.out.println("---At the Sender---\n Enter data stream: ");
   String datastream = sc.nextLine();
   int datalen=datastream.length();
   int genlen =generator.length();
   int data[] = new int[datalen + genlen - 1];
   int codeword[] = new int[datalen + genlen - 1];
   int div[] = new int[generator.length()];
   for(int i=0;i<datastream.length();i++)
            data[i] = Integer.parseInt(datastream.charAt(i)+"");
   for(int i=0;i<generator.length();i++)
            div[i] = Integer.parseInt(generator.charAt(i)+"");
   codeword = calculateCrc(data,div,datalen);
   System.out.println("The CRC(Final Codeword) code is: "); //Display CRC-final codeword
   for(int i=0;i<datastream.length();i++)
            codeword[i] = Integer.parseInt(datastream.charAt(i)+"");
   for(int i=0;i<data.length;i++)
            System.out.print(codeword[i]);
   System.out.println("\n");
   System.out.println("---At the Receiver---\n Enter Received codeword: ");
//Check for input CRC code
   datastream = sc.nextLine();
   data = new int[datastream.length() + generator.length() - 1];
   for(int i=0;i<datastream.length();i++)
            data[i] = Integer.parseInt(datastream.charAt(i)+"");
   codeword = calculateCrc(data,div,datalen);
   boolean valid = true; //Checking remainder is zero or not
   for(int i=0;i<codeword.length;i++)
            if(codeword[i]==1){
                     valid = false;
                     break;
            }
   if(valid==true)
            System.out.println("Data stream is valid. No error occured");
   else
            System.out.println("Data stream is invalid. CRC error occured.");
   sc.close();
   }

public static int[] calculateCrc(int[] divrem,int[] divisor,int len){
//Calculation of CRC
   for(int i=0;i<len;i++)
   {
         if(divrem[i]==1)
            for(int j=0;j<divisor.length;j++)
                     divrem[i+j] ^= divisor[j];
   }
   return divrem;
   }
}
```

Output Sample 1:

```
---At the Sender---
Enter data stream:
110101
The CRC(Final Codeword) code is:
1101010110011011110110

---At the Receiver---
 Enter Received codeword:
1101010110011011110110
Data stream is valid. No error occured
```

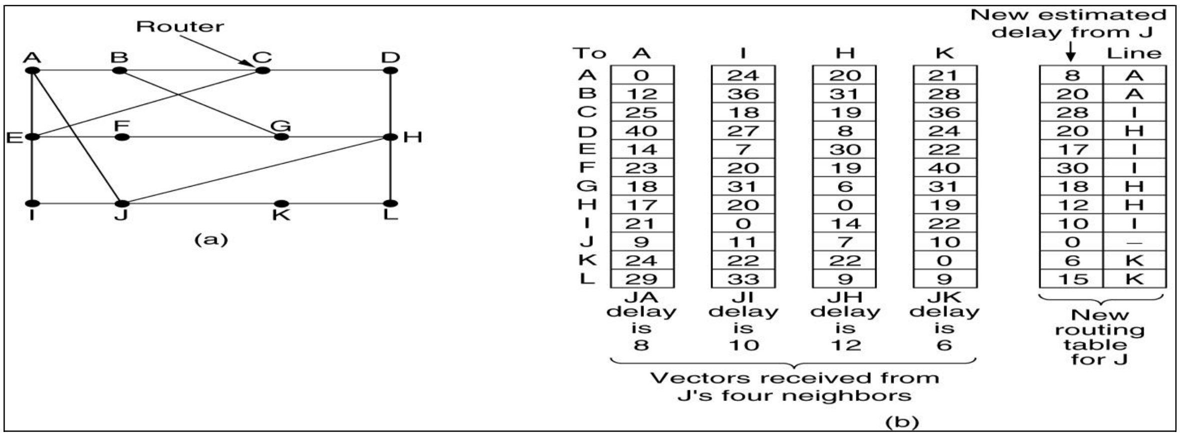
Output Sample 2:

```
---At the Sender---
Enter data stream:
110101
The CRC(Final Codeword) code is:
1101010110011011110110
---At the Receiver---
 Enter Received codeword:
1001111001100101010101
Data stream is invalid. CRC error occured.
```

Program No. 7: Write a Java program to find the shortest path between vertices using Bellman- Ford algorithm.

Distance Vector Algorithm is a decentralized routing algorithm that requires that each router simply inform its neighbors of its routing table. For each network path, the receiving routers pick the neighbor advertising the lowest cost, then add this entry into its routing table for re-advertisement. To find the shortest path, Distance Vector Algorithm is based on one of two basic algorithms: The Bellman-Ford and the Dijkstra algorithms. Routers that use this algorithm have to maintain the distance tables (which is a one- dimension array -- "a vector"), which tell the dist and shortest path to sending packets to each node in the network. The information in the distance table is always up date by exchanging information with the neighboring nodes. The number of data in the table equals to that of all nodes in networks (excluded itself). The columns of table represent the directly attached neighbors whereas the rows represent all destinations in the network. Each data contains the path for sending packets to each destination in the network and distance/or time to transmit on that path (we call this as "cost"). The measurements in this algorithm are the number of hops, latency, the number of outgoing packets, etc.

The Bellman–Ford algorithm is an algorithm that computes shortest paths from a single source v to all of the other vertices in a weighted digraph. It is slower than Dijkstra's algorithm for the same problem, but more versatile, as it is capable of handling graphs in which some of the edge weights are negative numbers. Negative edge weights are found in various applications of graphs, hence the usefulness of this algorithm. If a graph contains a "negative cycle" (i.e. a cycle whose edges sum to a negative value) that is reachable from the source, then there is no cheapest path: any path that has a point on the negative cycle can be made cheaper by one more walk around the negative cycle. In such a case, the Bellman–Ford algorithm can detect negative cycles and report their existence.



To	A	I	H	K	New estimated delay from J	Line
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	—
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA delay is 8	JI delay is 10	JH delay is 12	JK delay is 6	New routing table for J	

Vectors received from J's four neighbors

Figure (a) A subnet. (b) Input from A, I, H, K, and the new routing table for J.

Implementation Algorithm:

1. send my routing table to all my neighbors whenever my link table changes
2. when I get a routing table from a neighbor on port P with link metric M:
   a. add L to each of the neighbor's metrics
   b. for each entry (D, P', M') in the updated neighbor's table:
      i. if I do not have an entry for D, add (D, P, M') to my routing table
      ii. if I have an entry for D with metric M", add (D, P, M') to my routing table if M' < M"
3. if my routing table has changed, send all the new entries to all my neighbors

Program:

```
import java.util.Scanner;
public class BellmanFord
{
    private int distances[];
    private int numberofvertices;
    public static final int MAX_VALUE = 999;
    public BellmanFord(int numberofvertices)
    {
        this.numberofvertices = numberofvertices;
        distances = new int[numberofvertices + 1];
    }
    public void BellmanFordEvaluation(int source, int adjacencymatrix[][])
    {
        for (int node = 1; node <= numberofvertices; node++)
        {
            distances[node] = MAX_VALUE;
        }
        distances[source] = 0;
        for (int node = 1; node <= numberofvertices - 1; node++)
        {
            for (int sourcenode = 1; sourcenode <= numberofvertices; sourcenode++)
            {
```

```
            for (int destinationnode = 1; destinationnode <= numberofvertices; destinationnode++)
            {
               if (adjacencymatrix[sourcenode][destinationnode] != MAX_VALUE)
               {

                  if (distances[destinationnode] > distances[sourcenode]
                        + adjacencymatrix[sourcenode][destinationnode])
                     distances[destinationnode] = distances[sourcenode]
                        + adjacencymatrix[sourcenode][destinationnode];

               }
            }
      }
     }
     for (int sourcenode = 1; sourcenode <= numberofvertices; sourcenode++)
     {
        for (int destinationnode = 1; destinationnode <= numberofvertices; destinationnode++)
        {
           if (adjacencymatrix[sourcenode][destinationnode] != MAX_VALUE)
           {
              if (distances[destinationnode] > distances[sourcenode]
                    + adjacencymatrix[sourcenode][destinationnode])
                 System.out.println("The Graph contains negative egde cycle");
           }
        }
     }
     for (int vertex = 1; vertex <= numberofvertices; vertex++)
     {
        System.out.println("distance of source  " + source + " to " + vertex + " is " + distances[vertex]);
     }
  }
  public static void main(String... arg)
  {
     int numberofvertices = 0;
     int source;
     Scanner scanner = new Scanner(System.in);
     System.out.println("Enter the number of vertices");
     numberofvertices = scanner.nextInt()
     int adjacencymatrix[][] = new int[numberofvertices + 1][numberofvertices + 1];
     System.out.println("Enter the adjacency matrix");
     for (int sourcenode = 1; sourcenode <= numberofvertices; sourcenode++)
     {
        for (int destinationnode = 1; destinationnode <= numberofvertices; destinationnode++)
        {
           adjacencymatrix[sourcenode][destinationnode] = scanner.nextInt();
           if (sourcenode == destinationnode)
           {
              adjacencymatrix[sourcenode][destinationnode] = 0;
              continue;
```

```
                }
                if (adjacencymatrix[sourcenode][destinationnode] == 0)
                {
                    adjacencymatrix[sourcenode][destinationnode] = MAX_VALUE;
                }
            }
        }
        System.out.println("Enter the source vertex");
        source = scanner.nextInt();
        BellmanFord bellmanford = new BellmanFord(numberofvertices);
        bellmanford.BellmanFordEvaluation(source, adjacencymatrix);
        scanner.close();
    }
}
```

Output Sample 1:

```
Enter the number of vertices
4
Enter the adjacency matrix
0 5 1 4
5 0 6 2
1 6 0 3
4 2 3 0
Enter the source vertex 1
After (N-1)th Iteration
distance of source  1 to 1 is 0
distance of source  1 to 2 is 5
distance of source  1 to 3 is 1
distance of source 1 to 4 is 4
After Nth Iteration
distance of source  1 to 1 is 0
distance of source  1 to 2 is 5
distance of source  1 to 3 is 1
distance of source  1 to 4 is 4
```

Output Sample 2:

```
Enter the number of vertices
6
Enter the adjacency matrix
0 4 1 999 999 999
999 0 999 999 1 2
999 999 0 3 999 999
999 999 999 0 999 999
-6 999 2 4 0 999
999 999 999 5 999 0
Enter the source vertex 1
After (N-1)th Iteration
distance of source 1 to 1 is -5
distance of source 1 to 2 is 0
distance of source 1 to 3 is -3
distance of source  1 to 4 is 0
distance of source  1 to 5 is 1
distance of source  1 to 6 is 2
The Graph contains negative edge cycle
The Graph contains negative edge cycle
The Graph contains negative edge cycle
The Graph contains negative edge cycle
The Graph contains negative edge cycle
The Graph contains negative edge cycle
After Nth Iteration
distance of source  1 to 1 is -6
distance of source  1 to 2 is -1
distance of source  1 to 3 is -4
distance of source  1 to 4 is -1
distance of source  1 to 5 is 0
distance of source  1 to 6 is 1
```

Program No. 8: Write a Java program for congestion control using leaky bucket algorithm.

The main concept of the leaky bucket algorithm is that the output data flow remains constant despite the variant input traffic, such as the water flow in a bucket with a small hole at the bottom. In case the bucket contains water (or packets) then the output flow follows a constant rate, while if the bucket is full any additional load will be lost because of spillover. In a similar way if the bucket is empty the output will be zero. From network perspective, leaky bucket consists of a finite queue (bucket) where all the incoming packets are stored in case there is space in the queue, otherwise the packets are discarded. In order to regulate the output flow, leaky bucket transmits one packet from the queue in a fixed time (e.g. at every clock tick). In the following figure we can notice the main rationale of leaky bucket algorithm, for both the two approaches (e.g. leaky bucket with water (a) and with packets (b)).

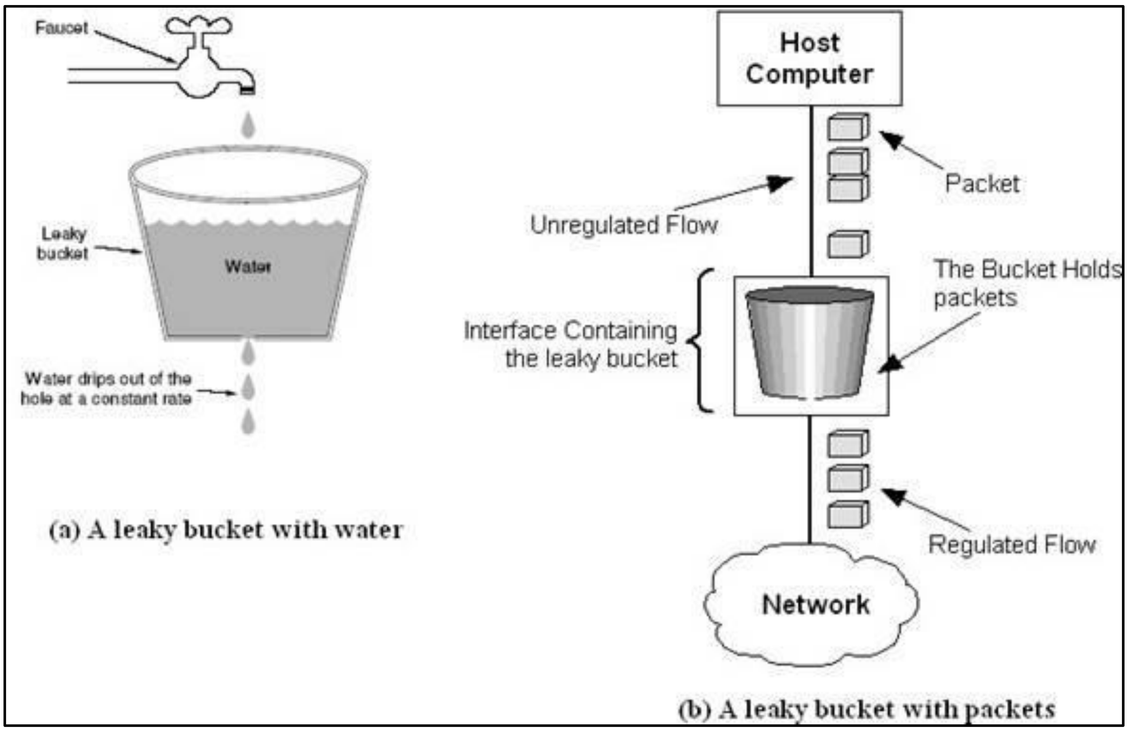

Faucet
Leaky bucket
Water
Water drips out of the hole at a constant rate
(a) A leaky bucket with water
Host Computer
Packet
Unregulated Flow
The Bucket Holds packets
Interface Containing the leaky bucket
Regulated Flow
Network
(b) A leaky bucket with packets

Figure: The leaky bucket traffic shaping algorithm

While leaky bucket eliminates completely bursty traffic by regulating the incoming data flow its main drawback is that it drops packets if the bucket is full. Also, it doesn't take into account the idle process of the sender which means that if the host doesn't transmit data for some time the bucket becomes empty without permitting the transmission of any packet.

The leaky-bucket algorithm:
The algorithm can be conceptually understood as follows:

- Consider a bucket with a hole in the bottom.
- The empty space of the bucket represents an amount of credit available measured in bytes.
- The size of the bucket is *b* bytes. This means that if the bucket is empty, *b* bytes of credit is available.
- If a packet arrives and its size is less than the available credit, the packet can be forwarded. Otherwise, it is discarded or queued depending on the application.
- The bucket leaks through the hole in its bottom at a constant rate of *r* bytes per second, this indicates credit accumulation

Program:

```
import java.util.Scanner;
public class LeakyBucket
{
public static void main(String args[])
{
  Scanner sc = new Scanner(System.in);
  int incoming, outgoing, buck_size, n, time = 1, store = 0;
  System.out.println("Enter bucket size, outgoing rate and Number of Packets:");
  buck_size = sc.nextInt();
  outgoing = sc.nextInt();
  n = sc.nextInt();
  while (n != 0)
  {
          System.out.println("Enter the incoming packet size at Time:" + (time++) );
          incoming=sc.nextInt();
          System.out.println("Incoming packet size is " + incoming);
          if (incoming <= (buck_size - store))
          {
                  store += incoming;
                  System.out.println("Bucket buffer size is " + store + " out of " + buck_size);
          }
          else
          {
                  int pktdrop = incoming - (buck_size - store);
                  System.out.println("Dropped " + pktdrop + " no of packets");
                  System.out.println("Bucket buffer size is 10 out of "+ buck_size);
                  store = buck_size;
          }
  store = store - outgoing;
  if(store < 0)
  {
          store=0;
          System.out.println("Empty Buffer");
  }
  System.out.println("After outgoing: "+ store +" packets left out of " + buck_size + " inbuffer\n");
  n--;
  }
  sc.close();
  }
}
```

Output 1

```
Enter bucket size, outgoing rate and Number of Packets:
10 5 3
Enter the incoming packet size at Time: 1
16
Incoming packet size is 16
Dropped 6 no of packets
Bucket buffer size is 10 out of 10
After outgoing: 5 packets left out of 10 in buffer
Enter the incoming packet size at Time: 2
6
Incoming packet size is 6
Dropped 1 no of packets
Bucket buffer size is 10 out of 10
After outgoing: 5 packets left out of 10 in buffer
Enter the incoming packet size at Time: 3
4
Incoming packet size is 4
Bucket buffer size is 9 out of 10
After outgoing: 4 packets left out of 10 in buffer
```

Output 2

```
Enter bucket size, outgoing rate and Number of Packets:
8 2 4
Enter the incoming packet size at Time:1
6
Incoming packet size is 6
Bucket buffer size is 6 out of 8
After outgoing: 4 packets left out of 8 in buffer
Enter the incoming packet size at Time:2
10
Incoming packet size is 10
Dropped 6 no of packets Bucket
buffer size is 10 out of 8
After outgoing: 6 packets left out of 8 in buffer
Enter the incoming packet size at Time:3 12
Incoming packet size is 12
Dropped 10 no of packets
Bucket buffer size is 10 out of 8
After outgoing: 6 packets left out of 8 in buffer
Enter the incoming packet size at Time:4 12
Incoming packet size is 12
Dropped 10 no of packets
Bucket buffer size is 10 out of 8
After outgoing: 6 packets left out of 8 in buffer
```

Program No. 9: Write a Java program to implement RSA algorithm, to encrypt the data while sending it and decrypt while receiving

RSA is algorithm used by modern computers to encrypt and decrypt messages. It is an asymmetric cryptographic algorithm. Asymmetric means that there are two different keys. This is also called *public key cryptography*, because one of them can be given to everyone. The other key must be kept private. It is based on the fact that finding the factors of an integer is hard (the factoring problem). RSA stands for **Ron Rivest, Adi Shamir** and **Leonard Adleman**, who first publicly described it in 1978. A user of RSA creates and then publishes the product of two large prime numbers, along with an auxiliary value, as their public key. The prime factors must be kept secret. Anyone can use the public key to encrypt a message, but with currently published methods, if the public key is large enough, only someone with knowledge of the prime factors can feasibly decode the message.

RSA involves a public key and private key. The public key can be known to everyone; it is used to encrypt messages. Messages encrypted using the public key can only be decrypted with the private key. The RSA algorithm can be used for both public key encryption and digital signatures. Its security is based on the difficulty of factoring large integers. The RSA algorithm's efficiency requires a fast method for performing the modular exponentiation operation. A less efficient, conventional method includes raising a number (the input) to a power (the secret or public key of the algorithm, denoted e and d, respectively) and taking the remainder of the division with N. A straight-forward implementation performs these two steps of the operation sequentially: first, raise it to the power and second, apply modulo. Basically RSA is cryptographic algorithm which is meant to encrypt the data, generally used in network security applications while we are sending the data from one source to destination. The concept of RSA algorithm starts with a two key concepts, it uses two keys (asymmetric keys) one is considered as the public key and another is a private key.

It was developed because using the symmetric encryption algorithm is easy but the key distribution is difficult, so the concept of two key concept appears to be more efficient. The whole algorithm depends on the fact that "It is not possible to judge another key when attacker gets one key" Here in two keys, one key is taken as the public key another as a private key. Public key is available for everyone to access, so whenever sender want to send the data to receiver, he uses the public key of receiver (as it is available for use to all) and encrypts the data using the key, this encrypted data is called cipher text, when receiver receives the cipher text, he can decrypt the data using his private key. Here even if the attacker knows the encryption algorithm, he can’t do anything until the keys are available.

A very simple example of RSA encryption

1. Select primes p = 11, q = 3.
2. n = p*q = 11*3 = 33

   phi = (p-1)(q-1) = 10*2 = 20
3. Choose e=3

   Check gcd(e, p-1) = gcd(3, 10) = 1 (i.e. 3 and 10 have no common factors except 1), and check gcd(e, q-1) = gcd(3, 2) = 1

   therefore gcd(e, phi) = gcd(e, (p-1)(q-1)) = gcd(3, 20) = 1
4. Compute d such that ed ≡ 1 (mod phi)

   i.e. compute d = $e^{\wedge -1}$ mod phi = $3^{\wedge -1}$ mod 20

   i.e. find a value for d such that phi divides (ed-1)

   i.e. find d such that 20 divides 3d-1.

   Simple testing (d = 1, 2, ...) gives d = 7

   Check: ed-1 = 3*7 - 1 = 20, which is divisible by phi.
5. Public key = (n, e) = (33, 3)

   Private key = (n, d) = (33, 7).

Now say we want to encrypt the message m = 7,

c = $m^{\wedge e}$ mod n = $7^{\wedge 3}$ mod 33 = 343 mod 33 = 13.

Hence the ciphertext c = 13.

To check decryption we compute

m' = $c^{\wedge d}$ mod n = $13^{\wedge 7}$ mod 33 = 7.

Note that we don't have to calculate the full value of 13 to the power 7 here. We can make use of the fact that a = bc mod n = (b mod n).(c mod n) mod n so we can break down a potentially large number into its components and combine the results of easier, smaller calculations to calculate the final value.

One way of calculating m' is as follows: -

m' = $13^{\wedge 7}$ mod 33 = $13^{\wedge(3+3+1)}$ mod 33 = $13^{\wedge 3}$.$13^{\wedge 3}$.13 mod 33

= ($13^{\wedge 3}$ mod 33).($13^{\wedge 3}$ mod 33).(13 mod 33) mod 33

= (2197 mod 33).(2197 mod 33).(13 mod 33) mod 33

= 19.19.13 mod 33 = 4693 mod 33

= 7.

Key Generation Algorithm

1. Generate two large random primes, p and q, of approximately equal size such that their product n = pq is of the required bit length, e.g. 1024 bits.
2. Compute n = pq and (φ) phi = (p-1)(q-1).
3. Choose an integer e, 1 < e < phi, such that gcd(e, phi) = 1.
4. Compute the secret exponent d, 1 < d < phi, such that
   ed ≡ 1 (mod phi).
5. The public key is (n, e) and the private key is (n, d). The values of p, q, and phi should also be kept secret.

- n is known as the modulus.
- e is known as the public exponent or encryption exponent.
- d is known as the secret exponent or decryption exponent.

Note: It is possible to find a smaller d by using lcm(p-1,q-1) instead of phi, lcm(p-1,q-1) = phi /gcd(p-1,q-1)).

Encryption

Sender A does the following: -

1. Obtains the recipient B's public key (n, e).
2. Represents the plaintext message as a positive integer m.
3. Computes the ciphertext $c = m^{\wedge e} \bmod n$.
4. Sends the ciphertext c to B.

Decryption

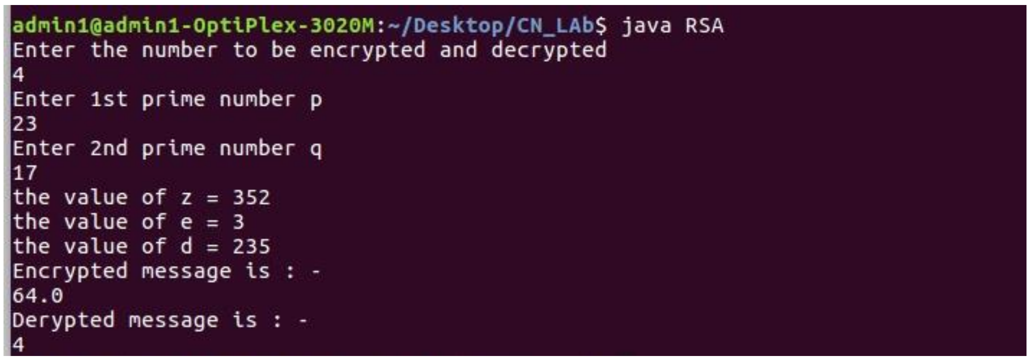
Recipient B does the following: -

1. Uses his private key (n, d) to compute $m = c^{\wedge d} \bmod n$.
2. Extracts the plaintext from the integer representative m.

Program:

```
 import java.util.*;
 import java.math.*;
 class RSA
 {
   public static void main(String args[])
   {
            Scanner sc=new Scanner(System.in);
            int p,q,n,z,d=0,e,i;
            System.out.println("Enter the number to be encrypted and decrypted");
            int msg=sc.nextInt();
            double c;
            BigInteger msgback;
            System.out.println("Enter 1st prime number p");
            p=sc.nextInt();
            System.out.println("Enter 2nd prime number q");
            q=sc.nextInt();

            n=p*q;
            z=(p-1)*(q-1);
            System.out.println("the value of z = "+z);

            for(e=2;e<z;e++)
            {
                        if(gcd(e,z)==1)             // e is for public key exponent
                        {
                                    break;
                        }
            }
            System.out.println("the value of e = "+e);
            for(i=0;i<=9;i++)
            {
                        int x=1+(i*z);
                        if(x%e==0)       //d is for private key exponent
                        {
                                    d=x/e;
                                    break;
                        }
            }
            System.out.println("the value of d = "+d);
            c=(Math.pow(msg,e))%n;
            System.out.println("Encrypted message is : -");
            System.out.println(c);
             //converting int value of n to BigInteger
            BigInteger N = BigInteger.valueOf(n);
            //converting float value of c to BigInteger
            BigInteger C = BigDecimal.valueOf(c).toBigInteger();
            msgback = (C.pow(d)).mod(N);
            System.out.println("Derypted message is : -");
            System.out.println(msgback);
  }
  static int gcd(int e, int z)
  {
            if(e==0)
                        return z;
            else
                        return gcd(z%e,e);
  }
}
```

Output 1



```
admin1@admin1-OptiPlex-3020M:~/Desktop/CN_LAb$ java RSA
Enter the number to be encrypted and decrypted
4
Enter 1st prime number p
23
Enter 2nd prime number q
17
the value of z = 352
the value of e = 3
the value of d = 235
Encrypted message is : -
64.0
Derypted message is : -
4
```

Program No. 10: Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present.

The term *network programming* refers to writing programs that execute across multiple devices (computers), in which the devices are all connected to each other using a network.

The java.net package of the J2SE APIs contains a collection of classes and interfaces that provide the low-level communication details, allowing you to write programs that focus on solving the problem at hand.

The java.net package provides support for the two common network protocols −

- **TCP** − TCP stands for Transmission Control Protocol, which allows for reliable communication between two applications. TCP is typically used over the Internet Protocol, which is referred to as TCP/IP.
- **UDP** − UDP stands for User Datagram Protocol, a connection-less protocol that allows for packets of data to be transmitted between applications.

Sockets are a protocol independent method of creating a connection between processes.
Sockets can be either

- ➢ *Connection based or connectionless*: Is a connection established before communication or does each packet describe the destination?
- ➢ *Packet based or streams based*: Are there message boundaries or is it one stream?
- ➢ *Reliable or unreliable:* Can messages be lost, duplicated, reordered, or corrupted?

Socket characteristics

Sockets are characterized by their domain, type and transport protocol. Common domains are:

- ➢ AF_UNIX: address format is UNIX pathname
- ➢ AF_INET: address format is host and port number

Common types are:

- ➢ *virtual circuit*: received in order transmitted and reliably
- ➢ *datagram*: arbitrary order, unreliable

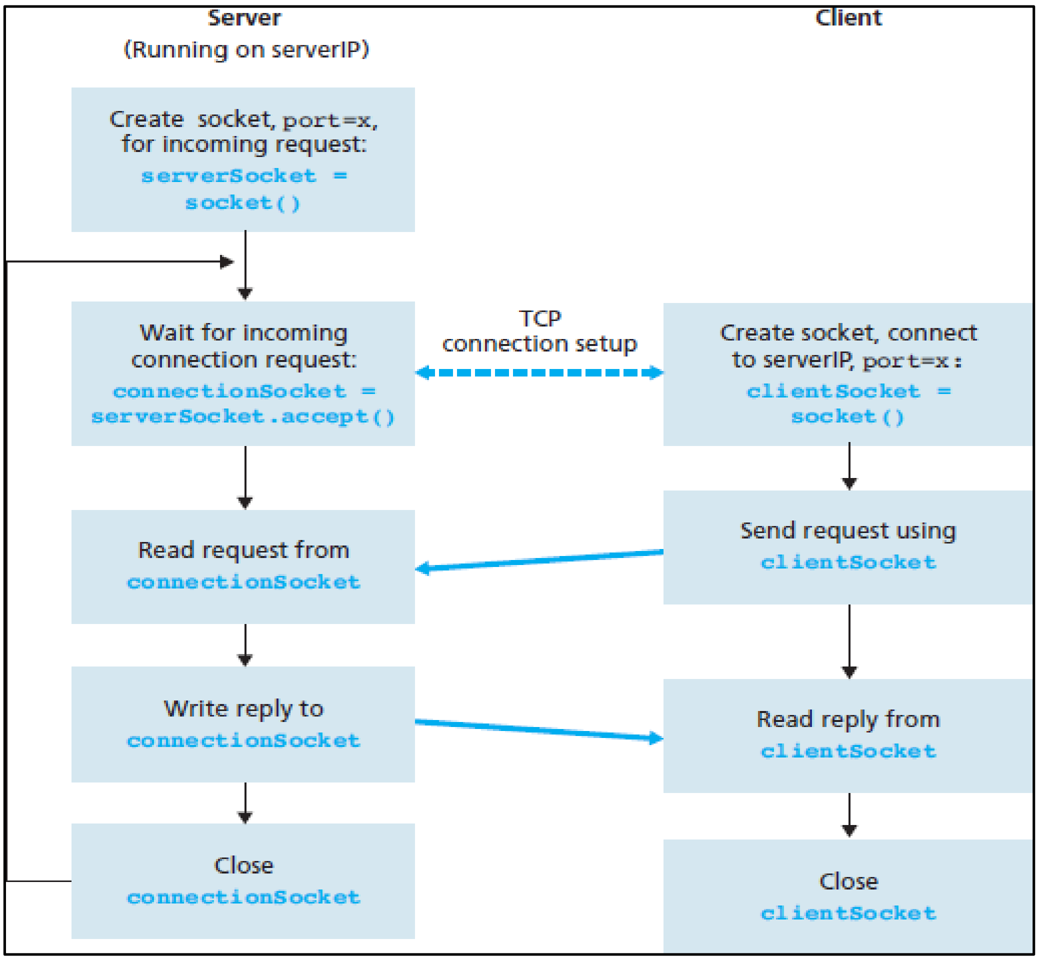
Each socket type has one or more protocols. Ex:

- ➢ TCP/IP (virtual circuits)
- ➢ UDP (datagram)

Use of sockets:

- Connection–based sockets communicate client-server: the server waits for a connection from the client
- Connectionless sockets are peer-to-peer: each process is symmetric.

Socket is an interface which enables the client and the server to communicate and pass on information from one another. Sockets provide the communication mechanism between two computers using TCP. A client program creates a socket on its end of the communication and attempts to connect that socket to a server. When the connection is made, the server creates a socket object on its end of the communication. The client and the server can now communicate by writing to and reading from the socket. The java.net.Socket class represents a socket, and the java.net.ServerSocket class provides a mechanism for the server program to listen for clients and establish connections with them.


Server
(Running on serverIP)
Client
Create socket, port=x, for incoming request:
serverSocket = socket()
Wait for incoming connection request:
connectionSocket = serverSocket.accept()
TCP connection setup
Create socket, connect to serverIP, port=x:
clientSocket = socket()
Read request from connectionSocket
Send request using clientSocket
Write reply to connectionSocket
Read reply from clientSocket
Close connectionSocket
Close clientSocket
Fig. 1 The client-server application using TCP

As shown in the Fig. 1, the following steps occur when establishing a TCP connection between two computers using sockets –

- The server instantiates a ServerSocket object, denoting which port number communication is to occur on.
- The server invokes the accept() method of the ServerSocket class. This method waits until a client connects to the server on the given port.
- After the server is waiting, a client instantiates a Socket object, specifying the server name and the port number to connect to.
- The constructor of the Socket class attempts to connect the client to the specified server and the port number. If communication is established, the client now has a Socket object capable of communicating with the server.
- On the server side, the accept() method returns a reference to a new socket on the server that is connected to the client's socket.

Program:

Server SideProgram:

```
import java.io.*;
import java.net.*;
public class FileServer
{
public static void main(String[] args)
{
new FileServer();
}
public FileServer()
{
try
{
  ServerSocket serversocket=new ServerSocket(8000);
  System.out.println("Server Started....");
  System.out.println("----------------------------------------------");
  Socket socket=serversocket.accept();
  DataInputStream input=new DataInputStream(socket.getInputStream());
  DataOutputStream output=new DataOutputStream(socket.getOutputStream());
  String str=input.readUTF();
  System.out.println("Requested File Name:"+str);
  System.out.println("----------------------------------------------");
  String everything;
try
{
  InputStream in = new FileInputStream(str);
  BufferedReader reader = new BufferedReader(new InputStreamReader(in));
  StringBuilder out = new StringBuilder();
  String line;
  System.out.println("Reading Contents of the File...");
  System.out.println("----------------------------------------------");
while ((line = reader.readLine()) != null)
{
  out.append(line+"\n");
}
  everything= out.toString();
  System.out.println("File Contents sent to client...");
  System.out.println("----------------------------------------------");
}
catch(Exception ex)
{
  everything="File Not Found!";
}
  output.writeUTF(everything);
}
catch(Exception ex)
{
  ex.printStackTrace();
}
}
}
```

Client Side Program:

```
import java.io.*;
import java.net.*;
public class FileClient
{
public static void main(String[] args)
{
new FileClient();
}
public FileClient()
{
BufferedReader bufReader=new BufferedReader(new InputStreamReader(System.in));
try
{
  System.out.println("Enter IP address of the server:");
  String saddr = bufReader.readLine();
  Socket clientsocket=new Socket(saddr,8000);
  System.out.println("Connecting to Server....");
  DataInputStream input=new DataInputStream(clientsocket.getInputStream());
  DataOutputStream output=new DataOutputStream(clientsocket.getOutputStream());
```

```
    System.out.println("Enter File Name:");
    String Name=bufReader.readLine();
    output.writeUTF(Name);
    String EchoedFile=input.readUTF();
    System.out.println("---------------------------------------------");
    System.out.println("Content of a File:\n\n"+EchoedFile);
    System.out.println("---------------------------------------------");
    clientsocket.close();
  }
  catch(IOException ex)
  {
    ex.printStackTrace();
  }
  }
  }
```

Note: Create two different files Client.java and Server.java. Follow the steps given:

1. Open a terminal run the server program and provide the filename to send.
2. Open one more terminal run the client program and provide the IP address of the server. We can give localhost address "127.0.0.1" as it is running on same machine or give the IP address of the machine.
3. Send any start bit to start sending file.
4. Refer https://www.tutorialspoint.com/java/java_networking.htm for all the parameters, methods description in socket communication.

Output:

Server Side:

```
Server Started....
---------------------------------------------
Requested File Name: abc.txt
---------------------------------------------
```

Client Side:

```
Enter IP address of the server:
localhost
Connecting to Server....
Enter File Name:
abc.txt
---------------------------------------------
Content of a File:

The content of the file will be displayed.
---------------------------------------------
```

CHAPTER 3

ADDITIONAL PROGRAMS (CONTENT BEYOND SYLLABUS)

1. TCP- TWO WAY COMMUNICATION

AIM: To write a java program to implement two-way communication using TCP (Transmission Control Protocol).

ALGORITHM:

SERVER:

1. Start the Program.
2. Import .net package and other packages.
3. Declare objects for ServerScoket, Socket and printStream to transfer server data.
4. Declare objects for Scoket and DataInputStream to receive client data.
5. Using printStream transfer the data in OutputStream via a port.
6. Run loop to send data from server until an"end or exit" String is transferred.
7. Using the same loop receive data from server and store it in a StringUsing dataInputStream.
8. print the String to display the server data and exit when an"end or exit" Message is encountered.

CLIENT:

1. Start the program.
2. Import .net package and other packages.
3. Declare objects for Socket, Socket and PrintStream to transfer54 the client data.
4. Declare objects for Socket and DataInputStream to receive server the data.
5. Using PrintStream transfer the data in OutputStream via a port.
6. Run loop to send data from server until an "end or exit" string is transferred.
7. Using the same loop receive data from server and store it in a String using DataInputStream.
8. Print the string to display the server data and exit when "send or exit" Message is encountered.

SOURCE CODE:

CLIENT

```
import java.io.*;
import java.net.*;
import java.lang.*;
class client1
{
public static void main(String a[])throws IOException
{
Socket s=new Socket("LocalHost",8000);
DataInputStream in=new DataInputStream(s.getInputStream());
DataInputStream inn=new DataInputStream(System.in);
PrintStream dos=new PrintStream(s.getOutputStream());
while(true)

{
String str=in.readLine();
System.out.println("message received:"+str);
if(str.equals("end"))
{
s.close();
 break;
}
System.out.println("enter the message to send: ");
String str1=inn.readLine();
dos.println(str1);
if(str1.equals("end"))
{
s.close();
break;
}  }  }  }
```

SERVER

```
import java.io.*;
import java.net.*;
import java.lang.*;
class server1
{
public static void main(String a[])throws IOException
{
ServerSocket ss=new ServerSocket(8000);
Socket s=ss.accept();
PrintStream dos=new PrintStream(s.getOutputStream());
DataInputStream in=new DataInputStream(System.in);
DataInputStream inn=new DataInputStream(s.getInputStream());
while(true)
{
System.out.println("enter the message to send: ");
String str=in.readLine();
dos.println(str);
if(str.equals("end"))
{
ss.close();
 break;
}
String str1=inn.readLine();
System.out.println("message received"+str1);
if(str1.equals("end"))
{
ss.close();
break;
}   }   }  }
```

OUTPUT:

SERVER

```
enter the message to send: HAI FRIENDS
message received: HELLO
enter the message to send: end
```

CLIENT

```
message received: HAI FRIENDS
enter the message to send: HELLO
enter the message to send: end
```

- **UDP- TWO WAY COMMUNICATION**

AIM: To write a java program to perform two way message transfer using the user datagram protocol (UDP).

ALGORITHM:

SERVER:

1. Start the program
2. Import .net and other necessary packages.
3. Declare objects for datagramSocket and DatagramPacket to receive packet data from server.
4. Receive an object for InetAddress of the LocalHost.
5. Receive the client data using receive() method and store it to a string.
6. Run a loop and store the data in the string until the received message points end.
7. Print the string unless it encounters end.
8. Get user input in the same loop and send data until the user input points end.
9. Convert the user input into bytes and send the byte using DatagramPacket and DatagramSocket.
10. If end is encountered, exit sending data and program.

CLIENT:

1. Start the program
2. Import .net and other necessary packages.
3. Declare objects for datagramSocket and DatagramPacket to receive packet data from server.
4. Declare an object for InetAddress of the Local Host.
5. Receive the Server data using receive() method and store it to a st ring.
6. Run a loop and store the data in the string until the received message points the end.
7. Print the string unless it encounters end.
8. Get user input in the same loop and send data until the user input points end.
9. Convert the user input into b ytes and send the byte using DatagramPacket and DatagramSocket.
10. If end is encountered, exit sending data and program.

SOURCE CODE:

SENDER
```
import java.io.*;
import java.net.*;
class sender
{
 public static void main(String a[])throws Exception
{
while(true)
{
DatagramSocket ds=new DatagramSocket();
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
System.out.println("enter the message:");
String message=br.readLine();
byte bl[]=message.getBytes();
InetAddress add=InetAddress.getLocalHost();
DatagramPacket dp=new DatagramPacket(bl,bl.length,add,1234);
ds.send(dp);
if(message.equals("exit"))
System.exit(0);
byte b[]=new byte[255];
DatagramPacket dp1=new DatagramPacket(bl,bl.length);
ds.receive(dp1);String message1=new String(dp1.getData());
System.out.println("received message:" +message1);
}
}
}
```

RECEIVER
```
import java.io.*;
import java.net.*;
import java.lang.*;
class receiver
{
public static void main(String a[])throws IOException
{
DatagramSocket ds=new DatagramSocket(1234);
 byte b[]=new byte[255];
while(true)
{
DatagramPacket dp=new DatagramPacket(b,b.length);
ds.receive(dp);
String message=new String(dp.getData());
System.out.println("Message Received:"+message);
InetAddress add=dp.getAddress();
int port=dp.getPort();
System.out.println("Enter a line of text to send:");
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
String message1=br.readLine();
byte b1[]=message1.getByte();
DatagramPacket dp1=new DatagramPacket(b1,b1.length,add,port);
ds.send(dp1);
if(message1.equals("exit"))
System.exit(0);
}
}
```

OUTPUT:

SENDER :

```
enter the message: GOOD MORNING
received message: HAVE A NICE DAY
enter the message: end
```

RECEIVER:

```
Message Received: GOOD MORNING
Enter a line of text to send: HAVE A NICE DAY
Message Received: end
```

CHAPTER 4

VIVA QUESTIONS AND ANSWERS

1) What is a Link?
A link refers to the connectivity between two devices. It includes the type of cables and protocols used in order for one device to be able to communicate with the other.
2) What are the layers of the OSI reference model?
There are 7 OSI layers: Physical Layer, Data Link Layer, Network Layer, Transport Layer, Session Layer, Presentation Layer and Application Layer.
3) What is backbone network?
A backbone network is a centralized infrastructure that is designed to distribute different routes and data to various networks. It also handles management of bandwidth and various channels.
4) What is a LAN?
LAN is short for Local Area Network. It refers to the connection between computers and other network devices that are located within a small physical location.
5) What is a node?
A node refers to a point or joint where a connection takes place. It can be computer or device that is part of a network. Two or more nodes are needed in order to form a network connection.
6) What are routers?
Routers can connect two or more network segments. These are intelligent network devices that store information in its routing table such as paths, hops and bottlenecks. With this info, they are able to determine the best path for data transfer. Routers operate at the OSI Network Layer.
7) What is point to point link?
It refers to a direct connection between two computers on a network. A point to point connection does not need any other network devices other than connecting a cable to the NIC cards of both computers.
8) What is anonymous FTP?
Anonymous FTP is a way of granting user access to files in public servers. Users that are allowed access to data in these servers do not need to identify themselves, but instead log in as an anonymous guest.
9) What is subnet mask?
A subnet mask is combined with an IP address in order to identify two parts: the extended network address and the host address. Like an IP address, a subnet mask is made up of 32 bits.
10) What is the maximum length allowed for a UTP cable?
A single segment of UTP cable has an allowable length of 90 to 100 meters. This limitation can be overcome by using repeaters and switches.
11) What is data encapsulation?
Data encapsulation is the process of breaking down information into smaller manageable chunks before it is transmitted across the network. It is also in this process that the source and destination addresses are attached into the headers, along with parity checks.
12) Describe Network Topology
Network Topology refers to the layout of a computer network. It shows how devices and cables are physically laid out, as well as how they connect to one another.
13) What is VPN?
VPN means Virtual Private Network, a technology that allows a secure tunnel to be created across a network such as the Internet. For example, VPNs allow you to establish a secure dialup connection to a remote server.

14) Briefly describe NAT.
NAT is Network Address Translation. This is a protocol that provides a way for multiple computers on a common network to share single connection to the Internet.
15) What is the job of the Network Layer under the OSI reference model?
The Network layer is responsible for data routing, packet switching and control of network congestion. Routers operate under this layer.
16) How does a network topology affect your decision in setting up a network?
Network topology dictates what media you must use to interconnect devices. It also serves as basis on what materials, connector and terminations that is applicable for the setup.
17) What is RIP?
RIP, short for Routing Information Protocol is used by routers to send data from one network to another. It efficiently manages routing data by broadcasting its routing table to all other routers within the network. It determines the network distance in units of hops.
18) What are different ways of securing a computer network?
There are several ways to do this. Install reliable and updated anti-virus program on all computers. Make sure firewalls are setup and configured properly. User authentication will also help a lot. All of these combined would make a highly secured network.
19) What is NIC?
NIC is short for Network Interface Card. This is a peripheral card that is attached to a PC in order to connect to a network. Every NIC has its own MAC address that identifies the PC on the network.
20) What is WAN?
WAN stands for Wide Area Network. It is an interconnection of computers and devices that are geographically dispersed. It connects networks that are located in different regions and countries.
21) What is the importance of the OSI Physical Layer?
The physical layer does the conversion from data bits to electrical signal, and vice versa. This is where network devices and cable types are considered and setup.
22) How many layers are there under TCP/IP?
There are four layers: the Network Layer, Internet Layer, Transport Layer and Application Layer.
23) What are proxy servers and how do they protect computer networks?
Proxy servers primarily prevent external users who identifying the IP addresses of an internal network. Without knowledge of the correct IP address, even the physical location of the network cannot be identified. Proxy servers can make a network virtually invisible to external users.
24) What is the function of the OSI Session Layer?
This layer provides the protocols and means for two devices on the network to communicate with each other by holding a session. This includes setting up the session, managing information exchange during the session, and tear-down process upon termination of the session.
25) What is the importance of implementing a Fault Tolerance System? Are there limitations?
A fault tolerance system ensures continuous data availability. This is done by eliminating a single point of failure. However, this type of system would not be able to protect data in some cases, such as in accidental deletions.
26) What does 10Base-T mean?
The 10 refers to the data transfer rate, in this case is 10Mbps. The word Base refers to base band, as oppose to broad band. T means twisted pair, which is the cable used for that network.

27) What is a private IP address?
Private IP addresses are assigned for use on intranets. These addresses are used for internal networks and are not routable on external public networks. These ensures that no conflicts are present among internal networks while at the same time the same range of private IP addresses are reusable for multiple intranets since they do not "see" each other.

28) What is NOS?
NOS, or Network Operating System, is specialized software whose main task is to provide network connectivity to a computer in order for it to be able to communicate with other computers and connected devices.
29) What is DoS?
DoS, or Denial-of-Service attack, is an attempt to prevent users from being able to access the internet or any other network services. Such attacks may come in different forms and are done by a group of perpetuators. One common method of doing this is to overload the system server so it cannot anymore process legitimate traffic and will be forced to reset.
30) What is OSI and what role does it play in computer networks?
OSI (Open Systems Interconnect) serves as a reference model for data communication. It is made up of 7 layers, with each layer defining a particular aspect on how network devices connect and communicate with one another. One layer may deal with the physical media used, while another layer dictates how data is actually transmitted across the network.
31) What is the purpose of cables being shielded and having twisted pairs?
The main purpose of this is to prevent crosstalk. Crosstalks are electromagnetic interferences or noise that can affect data being transmitted across cables.
32) What is the advantage of address sharing?
By using address translation instead of routing, address sharing provides an inherent security benefit. That's because host PCs on the Internet can only see the public IP address of the external interface on the computer that provides address translation and not the private IP addresses on the internal network.
33) What are MAC addresses?
MAC, or Media Access Control, uniquely identifies a device on the network. It is also known as physical address or Ethernet address. A MAC address is made up of 6-byte parts.
34) What is the equivalent layer or layers of the TCP/IP Application layer in terms of OSI reference model?
The TCP/IP Application layer actually has three counterparts on the OSI model: The Session layer, Presentation Layer and Application Layer.
35) How can you identify the IP class of a given IP address?
By looking at the first octet of any given IP address, you can identify whether it's Class A, B or C. If the first octet begins with a 0 bit, that address is Class A. If it begins with bits 10 then that address is a Class B address. If it begins with 110, then it's a Class C network.
36) What is the main purpose of OSPF?
OSPF, or Open Shortest Path First, is a link-state routing protocol that uses routing tables to determine the best possible path for data exchange.
37) What are firewalls?
Firewalls serve to protect an internal network from external attacks. These external threats can be hackers who want to steal data or computer viruses that can wipe out data in an instant. It also prevents other users from external networks from gaining access to the private network.
38) Describe star topology
Star topology consists of a central hub that connects to nodes. This is one of the easiest to setup and maintain.
39) What are gateways?

Gateways provide connectivity between two or more network segments. It is usually a computer that runs the gateway software and provides translation services. This translation is a key in allowing different systems to communicate on the network.
40) What is the disadvantage of a star topology?
One major disadvantage of star topology is that once the central hub or switch get damaged, the

entire network becomes unusable.

41) What is SLIP?

SLIP, or Serial Line Interface Protocol, is actually an old protocol developed during the early UNIX days. This is one of the protocols that are used for remote access.

42) Give some examples of private network addresses.

10.0.0.0 with a subnet mask of 255.0.0.0
172.16.0.0 with subnet mask of 255.240.0.0
192.168.0.0 with subnet mask of 255.255.0.0

43) What is tracert?

Tracert is a Windows utility program that can used to trace the route taken by data from the router to the destination network. It also shows the number of hops taken during the entire transmission route.

44) What are the functions of a network administrator?

A network administrator has many responsibilities that can be summarize into 3 key functions: installation of a network, configuration of network settings, and maintenance/troubleshooting of networks.

45) Describe at one disadvantage of a peer to peer network.

When you are accessing the resources that are shared by one of the workstations on the network, that workstation takes a performance hit.

46) What is Hybrid Network?

A hybrid network is a network setup that makes use of both client-server and peer-to-peer architecture.

47) What is DHCP?

DHCP is short for Dynamic Host Configuration Protocol. Its main task is to automatically assign an IP address to devices across the network. It first checks for the next available address not yet taken by any device, then assigns this to a network device.

48) What is the main job of the ARP?

The main task of ARP or Address Resolution Protocol is to map a known IP address to a MAC layer address.

49) What is TCP/IP?

TCP/IP is short for Transmission Control Protocol / Internet Protocol. This is a set of protocol layers that is designed to make data exchange possible on different types of computer networks, also known as heterogeneous network.

50) How can you manage a network using a router?

Routers have built in console that lets you configure different settings, like security and data logging. You can assign restrictions to computers, such as what resources it is allowed access, or what particular time of the day they can browse the internet. You can even put restrictions on what websites are not viewable across the entire network.

51) What protocol can be applied when you want to transfer files between different platforms, such between UNIX systems and Windows servers?

Use FTP (File Transfer Protocol) for file transfers between such different servers. This is possible because FTP is platform independent.

52) What is the use of a default gateway?

Default gateways provide means for the local networks to connect to the external network. The default gateway for connecting to the external network is usually the address of the external router port.

53) One way of securing a network is through the use of passwords. What can be considered as good passwords?

Good passwords are made up of not just letters, but by combining letters and numbers. A password that combines uppercase and lowercase letters is favorable than one that uses all upper case or all lower case letters. Passwords must be not words that can easily be guessed by hackers, such as dates, names, favorites, etc. Longer passwords are also better than short ones.

54) What is the proper termination rate for UTP cables?

The proper termination for unshielded twisted pair network cable is 100 ohms.

55) What is netstat?

Netstat is a command line utility program. It provides useful information about the current TCP/IP settings of a connection.

56) What is the number of network IDs in a Class C network?

For a Class C network, the number of usable Network ID bits is 21. The number of possible network IDs is 2 raised to 21 or 2,097,152. The number of host IDs per network ID is 2 raised to 8 minus 2, or 254.

57) What happens when you use cables longer than the prescribed length?

Cables that are too long would result in signal loss. This means that data transmission and reception would be affected, because the signal degrades over length.

58) What common software problems can lead to network defects?

Software related problems can be any or a combination of the following:

- client server problems, application conflicts, error in configuration
- protocol mismatch, security issues, user policy and rights issues

59) What is ICMP?

ICMP is Internet Control Message Protocol. It provides messaging and communication for protocols within the TCP/IP stack. This is also the protocol that manages error messages that are used by network tools such as PING.

60) What is Ping?

Ping is a utility program that allows you to check connectivity between network devices on the network. You can ping a device by using its IP address or device name, such as a computer name.

61) What is peer to peer?

Peer to peer are networks that does not reply on a server. All PCs on this network act as individual workstations.

62) What is DNS?

DNS is Domain Name System. The main function of this network service is to provide host names to TCP/IP address resolution.

63) What advantages does fiber optics have over other media?

One major advantage of fiber optics is that is it less susceptible to electrical interference. It also supports higher bandwidth, meaning more data can be transmitted and received. Signal degrading is also very minimal over long distances.

64) What is the difference between a hub and a switch?

A hub acts as a multiport repeater. However, as more and more devices connect to it, it would not be able to efficiently manage the volume of traffic that passes through it. A switch provides a better alternative that can improve the performance especially when high traffic volume is expected across all ports.

65) What are the different network protocols that are supported by Windows RRAS services?

There are three main network protocols supported: NetBEUI, TCP/IP, and IPX.

66) What are the maximum networks and hosts in a class A, B and C network?

For Class A, there are 126 possible networks and 16,777,214 hosts

For Class B, there are 16,384 possible networks and 65,534 hosts

For Class C, there are 2,097,152 possible networks and 254 hosts

67) What is the standard color sequence of a straight-through cable?

orange/white, orange, green/white, blue, blue/white, green, brown/white, brown.

68) What protocols fall under the Application layer of the TCP/IP stack?

The following are the protocols under TCP/IP Application layer: FTP, TFTP, Telnet and SMTP.

69) You need to connect two computers for file sharing. Is it possible to do this without using a hub or router?

Yes, you can connect two computers together using only one cable. A crossover type cable can be use in this scenario. In this setup, the data transmit pin of one cable is connected to the data receive pin of the other cable, and vice versa.

70) What is ipconfig?

Ipconfig is a utility program that is commonly used to identify the addresses information of a computer on a network. It can show the physical address as well as the IP address.

71) What is the difference between a straight-through and crossover cable?

A straight-through cable is used to connect computers to a switch, hub or router. A crossover cable is used to connect two similar devices together, such as a PC to PC or Hub to hub.

72) What is client/server?

Client/server is a type of network wherein one or more computers act as servers. Servers provide a centralized repository of resources such as printers and files. Clients refers to workstation that access the server.

73) Describe networking.

Networking refers to the inter connection between computers and peripherals for data communication. Networking can be done using wired cabling or through wireless link.

74) When you move the NIC cards from one PC to another PC, does the MAC address gets transferred as well?

Yes, that's because MAC addresses are hard-wired into the NIC circuitry, not the PC. This also means that a PC can have a different MAC address when the NIC card was replace by another one.

75) Explain clustering support

Clustering support refers to the ability of a network operating system to connect multiple servers in a fault-tolerant group. The main purpose of this is the in the event that one server fails, all processing will continue on with the next server in the cluster.

76) In a network that contains two servers and twenty workstations, where is the best place to install an Anti-virus program?

An anti-virus program must be installed on all servers and workstations to ensure protection. That's because individual users can access any workstation and introduce a computer virus when plugging in their removable hard drives or flash drives.

77) Describe Ethernet.

Ethernet is one of the popular networking technologies used these days. It was developed during the early 1970s and is based on specifications as stated in the IEEE. Ethernet is used in local area networks.

78) What are some drawbacks of implementing a ring topology?

In case one workstation on the network suffers a malfunction, it can bring down the entire network. Another drawback is that when there are adjustments and reconfigurations needed to be performed on a particular part of the network, the entire network has to be temporarily brought down as well.

79) What is the difference between CSMA/CD and CSMA/CA?

CSMA/CD, or Collision Detect, retransmits data frames whenever a collision occurred. CSMA/CA, or Collision Avoidance, will first broadcast intent to send prior to data transmission.

80) What is SMTP?

SMTP is short for Simple Mail Transfer Protocol. This protocol deals with all Internal mail, and provides the necessary mail delivery services on the TCP/IP protocol stack.

81) What is multicast routing?

Multicast routing is a targeted form of broadcasting that sends message to a selected group of user, instead of sending it to all users on a subnet.

82) What is the importance of Encryption on a network?

Encryption is the process of translating information into a code that is unreadable by the user. It is then translated back or decrypted back to its normal readable format using a secret key or password. Encryption help ensure that information that is intercepted halfway would remain unreadable because the user has to have the correct password or key for it.

83) How are IP addresses arranged and displayed?

IP addresses are displayed as a series of four decimal numbers that are separated by period or dots. Another term for this arrangement is the dotted decimal format. An example is 192.168.101.2

84) Explain the importance of authentication.

Authentication is the process of verifying a user's credentials before he can log into the network. It is normally performed using a username and password. This provides a secure means of limiting the access from unwanted intruders on the network.

85) What do mean by tunnel mode?

This is a mode of data exchange wherein two communicating computers do not use IPSec themselves. Instead, the gateway that is connecting their LANs to the transit network creates a virtual tunnel that uses the IPSec protocol to secure all communication that passes through it.

86) What are the different technologies involved in establishing WAN links?

Analog connections - using conventional telephone lines; Digital connections - using digitalgrade telephone lines; switched connections - using multiple sets of links between sender and receiver to move data.

87) What is one advantage of mesh topology?

In the event that one link fails, there will always be another available. Mesh topology is actually one of the most fault-tolerant network topology.

88) When troubleshooting computer network problems, what common hardware-related problems can occur?

A large percentage of a network is made up of hardware. Problems in these areas can range from malfunctioning hard drives, broken NICs and even hardware startups. Incorrectly hardware configuration is also one of those culprits to look into.

89) What can be done to fix signal attenuation problems?

A common way of dealing with such a problem is to use repeaters and hub, because it will help regenerate the signal and therefore prevent signal loss. Checking if cables are properly terminated is also a must.

90) How does dynamic host configuration protocol aid in network administration?

Instead of having to visit each client computer to configure a static IP address, the network administrator can apply dynamic host configuration protocol to create a pool of IP addresses known as scopes that can be dynamically assigned to clients.

91) Explain profile in terms of networking concept?

Profiles are the configuration settings made for each user. A profile may be created that puts a user in a group, for example.

92) What is sneakernet?

Sneakernet is believed to be the earliest form of networking wherein data is physically transported using removable media, such as disk, tapes.

93) What is the role of IEEE in computer networking?

IEEE, or the Institute of Electrical and Electronics Engineers, is an organization composed of engineers that issues and manages standards for electrical and electronic devices. This includes networking devices, network interfaces, cablings and connectors.

94) What protocols fall under the TCP/IP Internet Layer?
There are 4 protocols that are being managed by this layer. These are ICMP, IGMP, IP and ARP.
95) When it comes to networking, what are rights?
Rights refer to the authorized permission to perform specific actions on the network. Each user on the network can be assigned individual rights, depending on what must be allowed for that user.
96) What is one basic requirement for establishing VLANs?
A VLAN requires dedicated equipment on each end of the connection that allows messages entering the Internet to be encrypted, as well as for authenticating users.
97) What is IPv6?
IPv6, or Internet Protocol version 6, was developed to replace IPv4. At present, IPv4 is being used to control internet traffic, butis expected to get saturated in the near future. IPv6 was designed to overcome this limitation.
98) What is RSA algorithm?
RSA is short for Rivest-Shamir-Adleman algorithm. It is the most commonly used public key encryption algorithm in use today.
99) What is mesh topology?
Mesh topology is a setup wherein each device is connected directly to every other device on the network. Consequently, it requires that each device have at least two network connections.
100) What is ns2?
ns is an object-oriented,discrete event simulator targeted at networking research. ns provides substantial support for simulation of tcp, routing, andmulticast protocols over wired and wireless (local and satellite) networks. Later ns-2 (version 2) was developed at uc berkeley in c++ and otcl (object-oriented extension of tcl).
101) What is simulation?
The process of designing a model of a real system and conducting experiments with this model for the purpose of understanding the behaviour of the system and/or evaluating various strategies for the operation of the system.
102) Explain basic architecture of NS-2.
NS-2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl). While the C++ defines the internal mechanism (i.e., a backend) of the simulation objects, the OTcl sets up simulation by assembling and configuring the objects as well as scheduling discrete events.
103) What is trace file?

The trace file (trace.tr) is a standard format used by ns2. In ns2, each time a packet moves from one node to another, or onto a link, or into a buffer, etc., it gets recorded in this trace file.
104) What is nam file?
A visual aid showing how packets flow along the network.
105) Why awk file is used?
The basic function of awk is to search files for lines (or other units of text) that contain certain patterns. When a line matches one of the patterns, awk performs specified actions on that line. awk keeps processing input lines in this way until the end of the input files are reached.
106) What is xgraph?
It is used for plotting purpose comes together NS2 installation package.
107) What different layers of TCP/IP model, does link, node, agent and traffic source represent in NS-2?
Link represents Network access layer; Node represents Internet layer; Agent represents Transport layer; and Traffic Source represents Application layer of TCP/IP model.

REFERENCES

1. **Communication Networks: Fundamental Concepts and Key Architectures -** Alberto Leon, Garcia and Indra Widjaja, 3rd Edition, Tata McGraw- Hill, 2004.
2. **Data and Computer Communication,** William Stallings, 8th tEdition, Pearson Education, 2007.
3. **Computer Networks: A Systems Approach -** Larry L. Peterson and Bruce S. David, 4th Edition, Elsevier, 2007.
4. **Introduction to Data Communications and Networking** – Wayne Tomasi, Pearson Education, 2005.
5. **Communication Networks – Fundamental Concepts and Key architectures –** Alberto Leon- Garcia and Indra Widjaja:, 2rd Edition, Tata McGraw-Hill, 2004
6. **Computer and Communication Networks –** Nader F. Mir:, Pearson Education, 2007.
7. **https://www.isi.edu/nsnam/ns/doc/ns_doc.pdf**
8. **https://github.com/jridgewell/ns2/blob/master/tcl/ex/example.tcl**