Maharaja Education Trust (R), Mysuru

Maharaja Institute of Technology Mysore

Belawadi, Sriranga Pattana Taluk, Mandya – 571 477

Approved by AICTE, New Delhi,
Affiliated to VTU, Belagavi & Recognized by Government of Karnataka

Lecture Notes on

Computer Networks Laboratory (17CSL57)

Prepared by

Department of Information Science and Engineering

Vision/ ಆಶಯ

"To be recognized as a premier technical and management institution promoting extensive education fostering research, innovation and entrepreneurial attitude"

ಸಂಶೋಧನೆ, ಆವಿಷ್ಕಾರ ಹಾಗೂ ಉದ್ಯಮಶೀಲತೆಯನ್ನು ಉತ್ತೇಜಿಸುವ ಅಗ್ರಮಾನ್ಯ ತಾಂತ್ರಿಕ ಮತ್ತು ಆಡಳಿತ ವಿಜ್ಞಾನ ಶಿಕ್ಷಣ ಕೇಂದ್ರವಾಗಿ ಗುರುತಿಸಿಕೊಳ್ಳುವುದು.

Mission/ ಧ್ಯೇಯ

- To empower students with indispensable knowledge through dedicated teaching and collaborative learning.

  ಸಮರ್ಪಣಾ ಮನೋಭಾವದ ಬೋಧನೆ ಹಾಗೂ ಸಹಭಾಗಿತ್ವದ ಕಲಿಕಾಕ್ರಮಗಳಿಂದ ವಿದ್ಯಾರ್ಥಿಗಳನ್ನು ಅತ್ಯತ್ಕೃಷ್ಟ ಜ್ಞಾನಸಂಪನ್ನರಾಗಿಸುವುದು.

- To advance extensive research in science, engineering and management disciplines.

  ವೈಜ್ಞಾನಿಕ, ತಾಂತ್ರಿಕ ಹಾಗೂ ಆಡಳಿತ ವಿಜ್ಞಾನ ವಿಭಾಗಗಳಲ್ಲಿ ವಿಸ್ತೃತ ಸಂಶೋಧನೆಗಳೊಡನೆ ಬೆಳವಣಿಗೆ ಹೊಂದುವುದು.

- To facilitate entrepreneurial skills through effective institute - industry collaboration and interaction with alumni.

  ಉದ್ಯಮ ಕ್ಷೇತಗಳೊಡನೆ ಸಹಯೋಗ, ಸಂಸ್ಥೆಯ ಹಿರಿಯ ವಿದ್ಯಾರ್ಥಿಗಳೊಂದಿಗೆ ನಿರಂತರ ಸಂವಹನಗಳಿಂದ ವಿದ್ಯಾರ್ಥಿಗಳಿಗೆ ಉದ್ಯಮಶೀಲತೆಯ ಕೌಶಲ್ಯ ಪಡೆಯಲು ನೆರವಾಗುವುದು.

- To instill the need to uphold ethics in every aspect.

  ಜೀವನದಲ್ಲಿ ನೈತಿಕ ಮೌಲ್ಯಗಳನ್ನು ಅಳವಡಿಸಿಕೊಳ್ಳುವುದರ ಮಹತ್ವದ ಕುರಿತು ಅರಿವು ಮೂಡಿಸುವುದು.

- To mould holistic individuals capable of contributing to the advancement of the society.

  ಸಮಾಜದ ಬೆಳವಣಿಗೆಗೆ ಗಣನೀಯ ಕೊಡುಗೆ ನೀಡಬಲ್ಲ ಪರಿಪೂರ್ಣ ವ್ಯಕ್ತಿತ್ವವುಳ್ಳ ಸಮರ್ಥ ನಾಗರೀಕರನ್ನು ರೂಪಿಸುವುದು.

VISION OF THE DEPARTMENT

To be recognized as the best centre for technical education and research in the field of information science and engineering.

MISSION OF THE DEPARTMENT

- To facilitate adequate transformation in students through a proficient teaching learning process with the guidance of mentors and all-inclusive professional activities.
- To infuse students with professional, ethical and leadership attributes through industry collaboration and alumni affiliation.
- To enhance research and entrepreneurship in associated domains and to facilitate real time problem solving.
-

PROGRAM EDUCATIONAL OBJECTIVES:

- Proficiency in being an IT professional, capable of providing genuine solutions to information science problems.
- Capable of using basic concepts and skills of science and IT disciplines to pursue greater competencies through higher education.
- Exhibit relevant professional skills and learned involvement to match the requirements of technological trends.

PROGRAM SPECIFIC OUTCOME:

Student will be able to

- **PSO1:** Apply the principles of theoretical foundations, data Organizations, networking concepts and data analytical methods in the evolving technologies.
- **PSO2:**Analyse proficient algorithms to develop software and hardware competence in both professional and industrial areas

Program Outcomes

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 Overview

SUBJECT: COMPUTER NETWORKS LABORATORY **SUBJECT CODE: 17CSL57**

A computer network is a digital telecommunications network which allows nodes to share resources. In computer networks, computing devices exchange data with each other using connections (data links) between nodes. Computer Networks laboratory cover the problems of Computer Networks and the standard ways to approach and resolve these problems, including relevant real-world, state-of-the-art examples. The practicals for the course will allow students to apply theory to real-world examples.

This course teaches the design and implementation techniques essential for engineering robust networks. Topics include networking principles, Transmission Control Protocol/Internet Protocol, naming and addressing (Domain Name System), routing protocols, transport layer services, congestion control, quality of service, network services, server-client interaction, wireless and mobile networking, security in computer networks, and network management.

Students can able to apply JAVA, TCL and networking principles to provide a solution for various problems. And also students can analyze, evaluate and measure the performance of GSM, CDMA and various network protocols on different network scenarios by using application software

Course Objectives

1. Demonstrate operation of network and its management commands
2. Simulate and demonstrate the performance of GSM and CDMA
3. Implement data link layer and transport layer protocols.

Course Outcomes

CO's	DESCRIPTION OF THE OUTCOMES
17CSL57.1	Apply the principles of networking and analyse the performance of GSM, CDMA and various network protocols by using application software.
17CSL57.2	Apply JAVA and TCL to demonstrate network algorithm and topology.
17CSL57.3	Evaluate and measure the program effect on different network scenarios, protocols and their performance behaviour.
17CSL57.4	Analyze principle input parameters for given engineering problem scenario and prepare appropriate document.

Maharaja Institute of Technology Mysore

Department of Information Science and Engineering

Syllabus

SUBJECT: COMPUTER NETWORK LABORATORY **SUBJECT CODE: 17CSl57**

Lab Experiments:
PART A
1. 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. 2. Implement transmission of ping messages/trace route over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion. 3. Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination. 4. Implement simple ESS and with transmitting nodes in wire-less LAN by simulation and determine the performance with respect to transmission of packets. 5. Implement and study the performance of GSM on NS2/NS3 (Using MAC layer) or equivalent environment. 6. Implement and study the performance of CDMA on NS2/NS3 (Using stack called Call net) or equivalent environment.
PART B
Implement the following in Java: 7. Write a program for error detecting code using CRC-CCITT (16- bits). 8. Write a program to find the shortest path between vertices using bellman-ford algorithm. 9. 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. 10. Write a program on datagram socket for client/server to display the messages on client side, typed at the server side. 11. Write a program for simple RSA algorithm to encrypt and decrypt the data. 12. Write a program for congestion control using leaky bucket algorithm.
Study Experiment / Project:
NIL

Index

SUBJECT: DESIGN AND ANALYSIS OF ALGORITHMS **SUBJECT CODE: 18CS43**

MAHARAJA INSTITUTE OF TECHNOLOGY MYSORE
BELAWADI, SRIRANGAPATNA Taluk, MANDYA-571438
Department of Information Science & Engineering

COMPUTER NETWORKS LABORATORY -17CSL57

General Lab Guidelines:

- Conduct yourself in a responsible manner at all times in the laboratory. Intentional misconduct will lead to the exclusion from the lab.
- Do not wander around, or distract other students, or interfere with the laboratory experiments of other students.
- Read the handout and procedures before starting the experiments. Follow all written and verbal instructions carefully. If you do not understand the procedures, ask the instructor or teaching assistant.
- Attendance in all the labs is mandatory, absence permitted only with prior permission from Class teacher.
- The workplace has to be tidy before, during and after the experiment.
- Do not eat food, drink beverages or chew gum in the laboratory.
- Every student should know the location and operating procedures of all Safety equipment including First Aid Kit and Fire extinguisher.

DO'S:-

- Uniform and ID card are must.
- Records have to be submitted every week for evaluation.
- Sign the log book when you enter/leave the laboratory.
- After the lab session, shut down the computers.
- Keep your belongings in designated area.
- Report any problem in system (if any) to the person in-charge.

DONT'S:-

- Do not insert metal objects such as clips, pins and needles into the computer casings(They may cause fire) and should not attempt to repair, open, tamper or interfere with any of the computer, printing, cabling, or other equipment in the laboratory.
- Do not change the system settings and keyboard keys.
- Do not upload, delete or alter any software/ system files on laboratory computers.
- No additional material should be carried by the students during regular labs.
- Do not open any irrelevant websites in labs.
- Do not use a flash drive on lab computers without the consent of lab instructor.
- Students are not allowed to work in Laboratory alone or without presence of the instructor/ teaching assistant.

MAHARAJA INSTITUTE OF TECHNOLOGY MYSORE
BELAWADI, SRIRANGAPATNA Taluk, MANDYA-571477
Department of Information Science & Engineering

NETWORK SIMULATION

Introduction:

Network simulation is an important tool in developing, testing and evaluating network protocols. Simulation can be used without the target physical hardware, making it economical and practical for almost any scale of network topology and setup. It is possible to simulate a link of any bandwidth and delay, even if such a link is currently impossible in the real world. With simulation, it is possible to set each simulated node to use any desired software. This means that meaning deploying software is not an issue. Results are also easier to obtain and analyze, because extracting information from important points in the simulated network is as done by simply parsing the generated trace files.

Different types of simulators

Some of the different types of simulators are as follows:-

- MIT's NETSIM
- NIST
- CPSIM
- INSANE
- NEST
- REAL
- NS
- OPNET
- NCTUns

ns (from **network simulator**) is a name for series of discrete event network simulators, specifically **ns-1**, **ns-2** and **ns-3**. All of them are discrete-event network simulators, primarily used in research and teaching. ns-3 is free software, publicly available under the GNU GPLv2 license for research, development, and use.

The goal of the ns-3 project is to create an open simulation environment for networking research that will be preferred inside the research community

- It should be aligned with the simulation needs of modern networking research.
- It should encourage community contribution, peer review, and validation of the software.

Since the process of creation of a network simulator that contains a sufficient number of high-quality validated, tested and actively maintained models requires a lot of work, ns-3 project spreads this workload over a large community of users and developers.

ns-2

In 1996-97, ns version 2 (ns-2) was initiated based on a refactoring by Steve McCanne. Use of Tcl was replaced by MIT'sObject Tcl (OTcl), an object-oriented dialect Tcl. The core of ns-2 is also written in C++, but the C++ simulation objects are linked to shadow objects in OTcl and variables can be linked between both language realms. Simulation scripts are written in the OTcl language, an extension of the Tcl scripting language.

Presently, ns-2 consists of over 300,000 lines of source code, and there is probably a comparable amount of contributed code that is not integrated directly into the main distribution (many forks of ns-2 exist, both maintained and unmaintained). It runs on GNU/Linux, FreeBSD, Solaris, Mac OS X and Windows versions that support Cygwin. It is licensed for use underversion 2 of the GNU General Public License.

Design

ns-3 is built using C++ and Python with scripting capability. The ns-3 library is wrapped to python thanks to the pybindgen library which delegates the parsing of the ns-3 C++ headers to gccxml and pygccxml to generate automatically the corresponding C++ binding glue. These automatically-generated C++ files are finally compiled into the ns-3 python module to allow users to interact with the C++ ns-3 models and core through python scripts. The ns-3 simulator features an integrated attribute-based system to manage default and per-instance values for simulation parameters. All of the configurable default values for parameters are managed by this system, integrated with command-line argument processing, Doxygen documentation, and an XML-based and optional GTK-based configuration subsystem.

The large majority of its users focuses on wireless simulations which involve models for Wi-Fi, WiMAX, or LTE for layers 1 and 2 and routing protocols such as OLSR and AODV.

Where can you **write Scripts**?
After login into terminal, follow step given below

1. vi filename.tcl
   It will open page, there you can write tcl scripts.

2. save file
   Press esc-> colon (shift + semicolon) ->wq (save and quit)
It save the file
3. To run tcl script
   ns filename.tcl

Basically **NS 2 programming** contains the following steps.

1.Create the event scheduler

2.Turn on tracing

3.Creating network

a) Computing setup routing - rtproto
b) Creating transport connection-Agents
c) Creating traffic-Applications

4. Monitoring

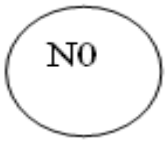
a) Visualization using nam

Note: *every tcl script must write in small letters only except protocol agents i.e. TCP, FTP*

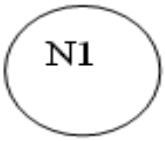
Template

Every ns2 script starts with creating simulator object set ns [new Simulator]

How to create node

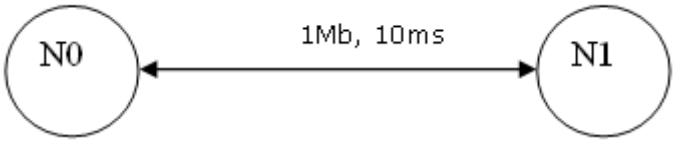


set n0 [$ns node]



set n1 [$ns node]

Creating link



$ns duplex-link $n0 $n1 1Mb 10ms DropTail

This line tells the simulator object to connect the nodes n0 and n1 with a duplex link with the bandwidth.1Megabit, a delay of 10ms and a DropTail queue.

How to use Trace?

We use simulator to see results. How is it achieved? Using trace

Two types of trace

1. generic trace
   for use with xgraph, and other things

2. nam trace
   for use with visualization

```
# open trace file
set tracefile [open out.tr w]
$ns trace-all $tracefile

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

Since we have started tracing, we should end also. For this we use finish procedure.

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

Finish procedure is forced to be called at the end with the line

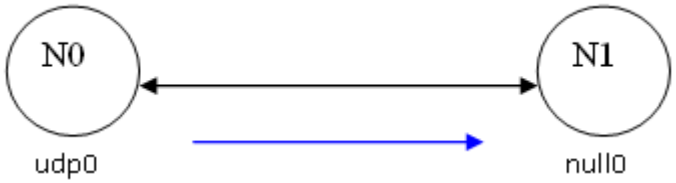
```
$ns at 5.0 “finish”
```

Every tcl script must contain following statement

$ns run

UDP communication

In UDP communication, data is flows from UDP agent to Null agent.



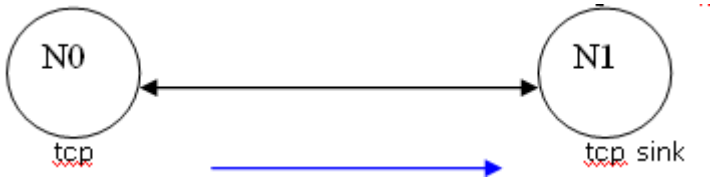
#Create a UDP agent and attach it to node n0
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0

create a null agent which act as traffic sink and attach it to node n1
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0

connect two agents with each other
$ns connect $udp0 $null0.

TCP Communication

In TCP communication, data is flows from TCP agent to TCPsink agent.



create Tcp agent and attach it to node no
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0

create a tcpsink agent which act as traffic sink and attach it to node n1
set tcpsink0 [new Agent/TCPSink]
$ns attach-agent $n1 $tcpsink0

```
# connect two agents with each other
$ns connect $tcp0 $tcpsink0
```

Traffic generator

For actual data to flow, we need traffic generators.They simulate some application traffic. Simple example using CBR

```
# creating CBR agent
set cbr0 [new Application/Traffic/CBR]

# Attach the CBR agent to some udp/tcp agent
$cbr0 attach-agent $udp0
```

Scheduling the events

Here “at” place major role.

```
$ns at 1.0 “$cbr0 start”
$ns at 5.0 “finish"
```

cbr0 will start at a time of 1.0 ms and whole process will stops at 5.0ms.we can also stop each and traffic generator. for example

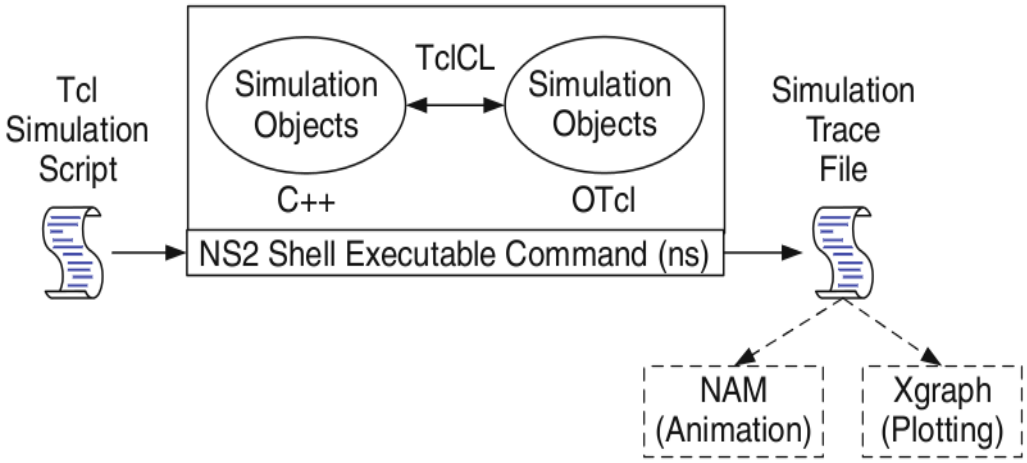
```
$ns at 4.0 ”$cbr0 stop”
```

Traffic generator cbr0 will stops at 4.0

Introduction to NS-2:

- Widely known as NS2, is simply an event driven simulation tool.
- Useful in studying the dynamic nature of communication networks.
- Simulation of wired as well as wireless network functions and protocols (e.g., routing algorithms, TCP, UDP) can be done using NS2.
- In general, NS2 provides users with a way of specifying such network protocols and simulating their corresponding behaviors.

Basic Architecture of NS2



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.

Basics of TCL

Syntax: command arg1 arg2 arg3

- **Hello World!**

```
puts stdout{Hello, World!}
Hello, World!
```

- **Variables** Command Substitution

```
set a 5          set len [string length foobar]
set b $a         set len [expr [string length foobar] + 9]
```

- **Simple Arithmetic**

```
expr 7.2 / 4
```

- **Procedures**

```
proc Diag {a b} {
set c [expr sqrt($a * $a + $b * $b)]
    return $c }
```

puts "Diagonal of a 3, 4 right triangle is [Diag 3 4]"

Output: Diagonal of a 3, 4 right triangle is 5.0

- **Loops**

```
while{$i < $n} {          for {set i 0} {$i < $n} {incr i} {
. . .                     . . .
}                         }
```

Wired TCL Script Components

Create the event scheduler

Open new files & turn on the tracing

Create the nodes

Setup the links

Configure the traffic type (e.g., TCP, UDP, etc)

Set the time of traffic generation (e.g., CBR, FTP)

Terminate the simulation

NS Simulator Preliminaries.

1. Initialization and termination aspects of the ns simulator.
2. Definition of network nodes, links, queues and topology.
3. Definition of agents and of applications.
4. The nam visualization tool.
5. Tracing and random variables.

Initialization and Termination of TCL Script in NS-2

An ns simulation starts with the command

```
set ns [new Simulator]
```

Which is thus the first line in the tcl script? This line declares a new variable as using the set command, you can call this variable as you wish, In general people declares it as ns because it is an instance of the Simulator class, so an object the code[new Simulator] is indeed the installation of the class Simulator using the reserved word new.

In order to have output files with data on the simulation (trace files) or files used for visualization (nam files), we need to create the files using "open" command:

#Open the Trace file

```
set tracefile1 [open out.tr w]

$ns trace-all $tracefile1
```

#Open the NAM trace file

```
set namfile [open out.nam w]
$ns namtrace-all $namfile
```

The above creates a dta trace file called "out.tr" and a nam visualization trace file called "out.nam".Within the tcl script,these files are not called explicitly by their names,but instead by pointers that are declared above and called "tracefile1" and "namfile" respectively.Remark that they begins with a # symbol.The second line open the file "out.tr" to be used for writing,declared with the letter "w".The third line uses a simulator method called trace-all that have as parameter the name of the file where the traces will go.

The last line tells the simulator to record all simulation traces in NAM input format.It also gives the file name that the trace will be written to later by the command $ns flush-trace.In our case,this will be the file pointed at by the pointer "$namfile",i.e the file "out.tr".

The termination of the program is done using a "finish" procedure.

#Define a 'finish' procedure

```
Proc finish { } {
global ns tracefile1 namfile
$ns flush-trace
Close $tracefile1
Close $namfile
Exec nam out.nam &
Exit 0
}
```

The word proc declares a procedure in this case called **finish** and without arguments. The word **global** is used to tell that we are using variables declared outside the procedure. The simulator method "**flush-trace"** will dump the traces on the respective files. The tcl command "**close"** closes the trace files defined before and **exec** executes the nam program for visualization. The command **exit** will ends

the application and return the number 0 as status to the system. Zero is the default for a clean exit. Other values can be used to say that is a exit because something fails.

At the end of ns program we should call the procedure "finish" and specify at what time the termination should occur. For example,

```
$ns at 125.0 "finish"
```

will be used to call "**finish**" at time 125sec.Indeed,the **at** method of the simulator allows us to schedule events explicitly.

The simulation can then begin using the command

```
$ns run
```

Definition of a network of links and nodes

The way to define a node is

```
set n0 [$ns node]
```

The node is created which is printed by the variable n0. When we shall refer to that node in the script we shall thus write $n0.

Once we define several nodes, we can define the links that connect them. An example of a definition of a link is:

```
$ns duplex-link $n0 $n2 10Mb 10ms DropTail
```

Which means that $n0 and $n2 are connected using a bi-directional link that has 10ms of propagation delay and a capacity of 10Mb per sec for each direction.

To define a directional link instead of a bi-directional one, we should replace "duplex-link" by "simplex-link".

In NS, an output queue of a node is implemented as a part of each link whose input is that node. The definition of the link then includes the way to handle overflow at that queue. In our case, if the buffer capacity of the output queue is exceeded then the last packet to arrive is dropped. Many alternative options exist, such as the RED (Random Early Discard) mechanism, the FQ (Fair Queuing), the DRR (Deficit Round Robin), the stochastic Fair Queuing (SFQ) and the CBQ (which including a priority and a round-robin scheduler).

In ns, an output queue of a node is implemented as a part of each link whose input is that node. We should also define the buffer capacity of the queue related to each link. An example would be:

```
#set Queue Size of link (n0-n2) to 20

$ns queue-limit $n0 $n2 20
```

Agents and Applications

We need to define routing (sources, destinations) the agents (protocols) the application that use them.

FTP over TCP

TCP is a dynamic reliable congestion control protocol. It uses Acknowledgements created by the destination to know whether packets are well received.

There are number variants of the TCP protocol, such as Tahoe, Reno, NewReno, Vegas. The type of agent appears in the first line:

```
set tcp [new Agent/TCP]
```

The command **$ns attach-agent $n0 $tcp** defines the source node of the tcp connection.

The command

```
set sink [new Agent /TCPSink]
```

Defines the behavior of the destination node of TCP and assigns to it a pointer called sink.

#Setup a UDP connection

```
set udp [new Agent/UDP]

$ns attach-agent $n1 $udp

set null [new Agent/Null]

$ns attach-agent $n5 $null

$ns connect $udp $null

$udp set fid_2
```

#setup a CBR over UDP connection

The below shows the definition of a CBR application using a UDP agent

The command **$ns attach-agent $n4 $sink** defines the destination node. The command **$ns connect $tcp $sink** finally makes the TCP connection between the source and destination nodes.

```
set cbr [new
Application/Traffic/CBR]

$cbr attach-agent $udp

$cbr set packetsize_ 100

$cbr set rate_ 0.01Mb

$cbr set random_ false
```

TCP has many parameters with initial fixed defaults values that can be changed if mentioned explicitly. For example, the default TCP packet size has a size of 1000bytes.This can be changed to another value, say 552bytes, using the command **$tcp set packetSize_ 552**.

When we have several flows, we may wish to distinguish them so that we can identify them with different colors in the visualization part. This is done by the command **$tcp set fid_ 1** that assigns to the TCP connection a flow identification of "1".We shall later give the flow identification of "2" to the UDP connection.

CBR over UDP

A UDP source and destination is defined in a similar way as in the case of TCP.

Instead of defining the rate in the command $cbr set rate_ 0.01Mb, one can define the time interval between transmission of packets using the command.

```
$cbr set interval_ 0.005
```

The packet size can be set to some value using

```
$cbr set packetSize_ <packet size>
```

Scheduling Events

NS is a discrete event based simulation. The tcp script defines when event should occur. The initializing command set ns [new Simulator] creates an event scheduler, and events are then scheduled using the format:

```
$ns at <time> <event>
```

The scheduler is started when running ns that is through the command $ns run.

The beginning and end of the FTP and CBR application can be done through the following command

```
$ns at 0.1 "$cbr start"
$ns at 1.0 " $ftp start"
$ns at 124.0 "$ftp stop"
$ns at 124.5 "$cbr stop"
```

Structure of Trace Files

When tracing into an output ASCII file, the trace is organized in 12 fields as follows in fig shown below, The meaning of the fields are:

Event	Time	From Node	To Node	PKT Type	PKT Size	Flags	Fid	Src Addr	Dest Addr	Seq Num	Pkt id

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.

XGRAPH

The xgraph program draws a graph on an x-display given data read from either data file or from standard input if no files are specified. It can display upto 64 independent data sets using different colors and line styles for each set. It annotates the graph with a title, axis labels, grid lines or tick marks, grid labels and a legend.

Syntax:

```
Xgraph [options] file-name
```

Options are listed here

/-bd <color> (Border)

This specifies the border color of the xgraph window.

/-bg <color> (Background)

This specifies the background color of the xgraph window.

/-fg<color> (Foreground)

This specifies the foreground color of the xgraph window.

/-lf <fontname> (LabelFont)

All axis labels and grid labels are drawn using this font.

/-t<string> (Title Text)

This string is centered at the top of the graph.

/-x <unit name> (XunitText)

This is the unit name for the x-axis. Its default is “X”.

/-y <unit name> (YunitText)

This is the unit name for the y-axis. Its default is “Y”.

Awk- An Advanced

awk is a programmable, pattern-matching, and processing tool available in UNIX. It works equally well with text and numbers. awk is not just a command, but a programming language too. In other words, awk utility is a pattern scanning and processing language. It searches one or more files to see if they contain lines that match specified patterns and then perform associated actions, such as writing the line to the standard output or incrementing a counter each time it finds a match.

Syntax:

```
awk option 'selection_criteria {action}' file(s)
```

Here, selection_criteria filters input and select lines for the action component to act upon. The selection_criteria is enclosed within single quotes and the action within the curly braces. Both the selection_criteria and action forms an awk program.

Example: $ awk '/manager/ {print}' emp.lst

Variables

Awk allows the user to use variables of there choice. You can now print a serial number, using the variable kount, and apply it those directors drawing a salary exceeding 6700:

```
$ awk –F"|" '$3 == "director" && $6 > 6700 {
kount =kount+1
printf " %3f %20s %-12s %d\n", kount,$2,$3,$6 }' empn.lst
```

THE –f OPTION: STORING awk PROGRAMS IN A FILE

You should holds large awk programs in separate file and provide them with the awk extension for easier identification. Let's first store the previous program in the file empawk.awk:

```
$ cat empawk.awk
```

Observe that this time we haven't used quotes to enclose the awk program. You can now use awk with the –f *filename* option to obtain the same output:

```
Awk –F"|" –f empawk.awk empn.lst
```

The BEGIN and END Sections

Awk statements are usually applied to all lines selected by the address, and if there are no addresses, then they are applied to every line of input. But, if you have to print something before processing the first line, for example, a heading, then the BEGIN section can be used gainfully. Similarly, the end section useful in printing some totals after processing is over.

The BEGIN and END sections are optional and take the form

BEGIN {action}

END {action}

These two sections, when present, are delimited by the body of the awk program. You can use them to print a suitable heading at the beginning and the average salary at the end.

Built-In Variables

Awk has several built-in variables. They are all assigned automatically, though it is also possible for a user to reassign some of them. You have already used NR, which signifies the record number of the current line. We'll now have a brief look at some of the other variable.

The FS Variable*:* as stated elsewhere, awk uses a contiguous string of spaces as the default field delimiter. FS redefines this field separator, which in the sample database happens to be the |. When used at all, it must occur in the BEGIN section so that the body of the program knows its value before it starts processing:

BEGIN {FS="|"}

This is an alternative to the –F option which does the same thing.

The OFS Variable*:* when you used the print statement with comma-separated arguments, each argument was separated from the other by a space. This is awk's default output field separator, and can reassigned using the variable OFS in the BEGIN section:

BEGIN { OFS="~" }

When you reassign this variable with a ~ (tilde), awk will use this character for delimiting the print arguments. This is a useful variable for creating lines with delimited fields.

The NF variable*:* NF comes in quite handy for cleaning up a database of lines that don't contain the right number of fields. By using it on a file, say emp.lst, you can locate those lines not having 6 fields, and which have crept in due to faulty data entry:

```
$awk 'BEGIN {FS = "|"}
NF! =6 {
Print "Record No ", NR, "has", "fields"}' empx.lst
```

NS-2 installation steps in Linux

- Go to **Computer - File System** - now paste the zip file **"ns-allinone-2.34.tar.gz"** into opt folder.
- Now **unzip** the file by typing the following **command**
  [root@localhost opt] # **tar -xzvfns-allinone-2.34.tar.gz**
- After the files get extracted, we get ns-allinone-2.34 folder as well as zip file ns-all in one-2.34.tar.gz
  [root@localhost opt]#**ns-allinone-2.34 ns-allinone-2.34.tar.gz**
- Now go to ns-allinone-2.33 folder and install it
  [root@localhost opt] # **cd ns-allinone-2.34**
  [root@localhost ns-allinone-2.33] #**./install**
- Once the installation is completed successfully we get certain pathnames in that terminal which must be pasted in **".bash_profile"**file.
- First **minimize the terminal** where installation is done and **open a new terminal** and open the file**".bash_profile"**
  [root@localhost ~] # **vi.bash_profile**
- When we open this file, we get a line in that file which is shown below
  PATH=$PATH:$HOME/bin

To this line we must paste the path which is present in the previous terminal where **ns was installed**. First put **":"** then paste the path in-front of bin. That path is shown below. **":/opt/ns-allinone-2.33/bin:/opt/ns-allinone-2.33/tcl8.4.18/unix:/opt/ns-allinone- 2.33/tk8.4.18/unix".**

- In the next line type **"LD_LIBRARY_PATH=$LD_LIBRARY_PATH:"** and paste the **two paths** separated by **":"** which are present in the previous terminal i.e **Important notices section(1)**

"/opt/ns-allinone-2.33/otcl-1.13:/opt/ns-allinone-2.33/lib"

- In the next line type **"TCL_LIBRARY=$TCL_LIBRARY:"** and paste the path which is present in previous terminal i.e **Important Notices section(2)**

"/opt/ns-allinone-2.33/tcl8.4.18/library"

- In the next line type **"exportLD_LIBRARY_PATH"**
- In the next line type **"exportTCL_LIBRARY"**
- The next two lines are already present the file **"export PATH"** and **"unsetUSERNAME"**
- **Save the program (ESC + shift : wq and press enter)**
- Nowintheterminalwherewehaveopened**.bash_profile**file,typethefollowingcommand to **check if path is updated correctly ornot**

  [root@localhost ~] # **vi.bash_profile**

  [root@localhost ~] # **source.bash_profile**

- If **path is updated properly**, then we will **get the prompt** as shown below
  [root@localhost ~]#
- Now open the previous terminal where you have installed**ns**
  [root@localhost ns-allinone-2.33] #
- Here we need to configure three packages **"ns-2.33", "nam-1.13" and"xgraph-12.1"**
- **First**, configure **"ns-2.33"** package as shown below
  [root@localhost ns-allinone-2.33] # **cd ns-2.33**
  [root@localhost ns-2.33] # **./configure**
  [root@localhost ns-2.33] # **make clean**
  [root@localhost ns-2.33] # **make** [root@localhost ns-2.33] # **make install** [root@localhost ns-2.33] #**ns**

%

- If we get **"%"** symbol it indicates that **ns-2.33 configuration** was **successful**.
- **Second,** configure **"nam-1.13"** package as shown below
  [root@localhost ns-2.33] # **cd ..**
  [root@localhost ns-allinone-2.33] # **cd nam-1.13**
  [root@localhost nam-1.13] # **./configure** [root@localhost nam-1.13] # **make clean** [root@localhost nam-1.13] # **make** [root@localhost nam-1.13] # **make install** [root@localhost nam-1.13] #**ns**

%

- If we get **"%"** symbol it indicates that **nam-1.13 configuration** was **successful**.
- **Third,** configure **"xgraph-12.1"** package as shown below
  [root@localhost nam-1.13] # **cd ..**
  [root@localhost ns-allinone-2.33] # **cd xgraph-12.1**
  [root@localhost xgraph-12.1] # **./configure**
  [root@localhost xgraph-12.1] # **make clean**
  [root@localhost xgraph-12.1] # **make** [root@localhost xgraph-12.1] # **make install** [root@localhost xgraph-12.1] #**ns**

%

This completes the installation process of "NS-2" simulator.

SIMULATION PROGRAMS: PART – A

Experiment 1: POINT-TO-POINT NETWORK

Aim: To 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.

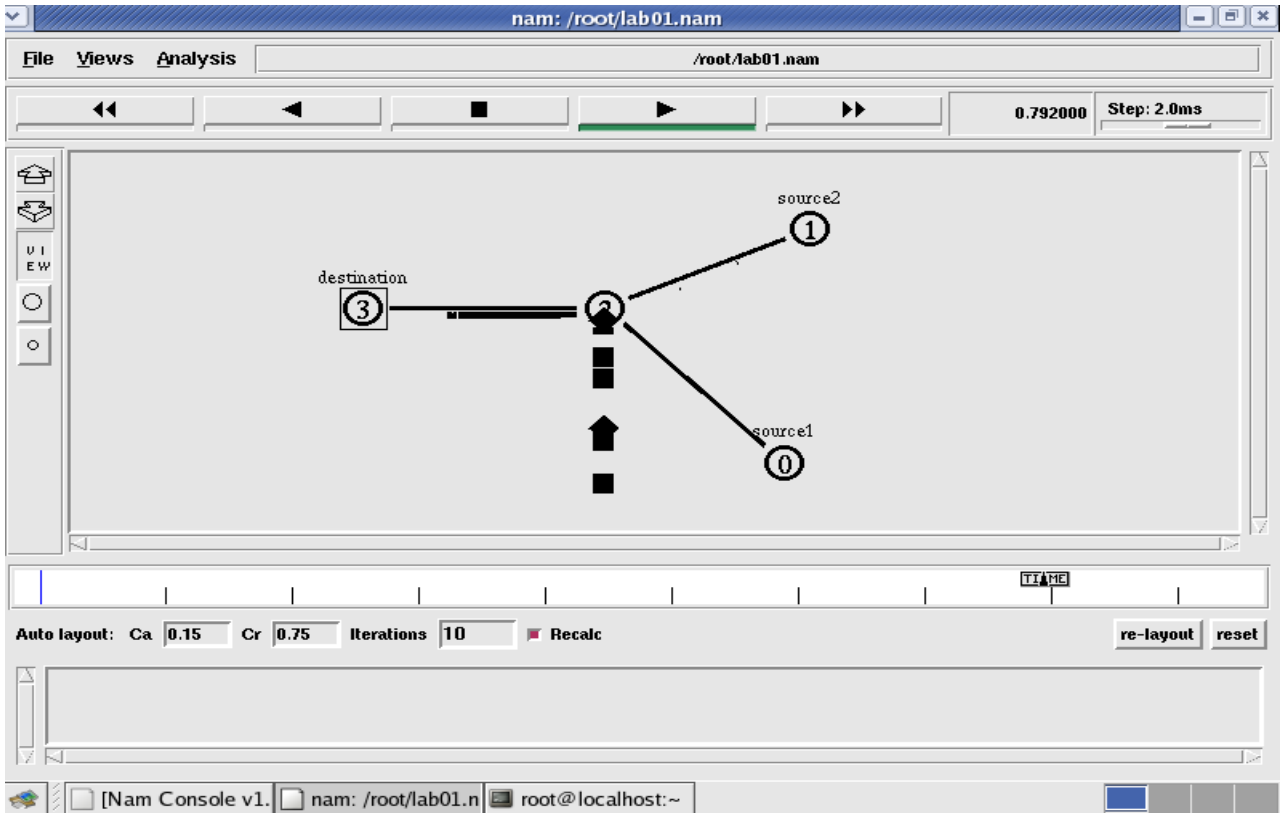
Step1: Open text editor, type the tcl program and save with extension .tcl (**prog1.tcl**)

```
set ns [new Simulator]
set nf [open prog1.nam w]
$ns namtrace-all $nf
set nd [open prog1.tr w]
$ns trace-all $nd

proc finish { } {
global ns nf nd
$ns flush-trace
close $nf
close $nd
exec nam prog1.nam &
exit 0
}

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 512kb 10ms DropTail
$ns queue-limit $n1 $n2 10

set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
set sink [new Agent/Null]
$ns attach-agent $n2 $sink
$ns connect $udp0 $sink

$ns at 0.2 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
$ns at 5.0 "finish"
$ns run
```

Step2: Open text editor, type the awk program and save with extension .awk (**prog1.awk**)
/* prog1.awk*/

```
BEGIN {
        dcount = 0;
        rcount = 0;
}
{
event = $1;
if(event == "d")
{
        dcount++;
}
if(event == "r")
{
        rcount++;
}
}
END {
printf("The no.of packets dropped  : %d\n ",dcount);
printf("The no.of packets recieved : %d\n ",rcount);
}
```

Step3: Run the simulation program → **ns prog1.tcl**



Step 4: Now press the play button in the simulation window and the simulation will begins.

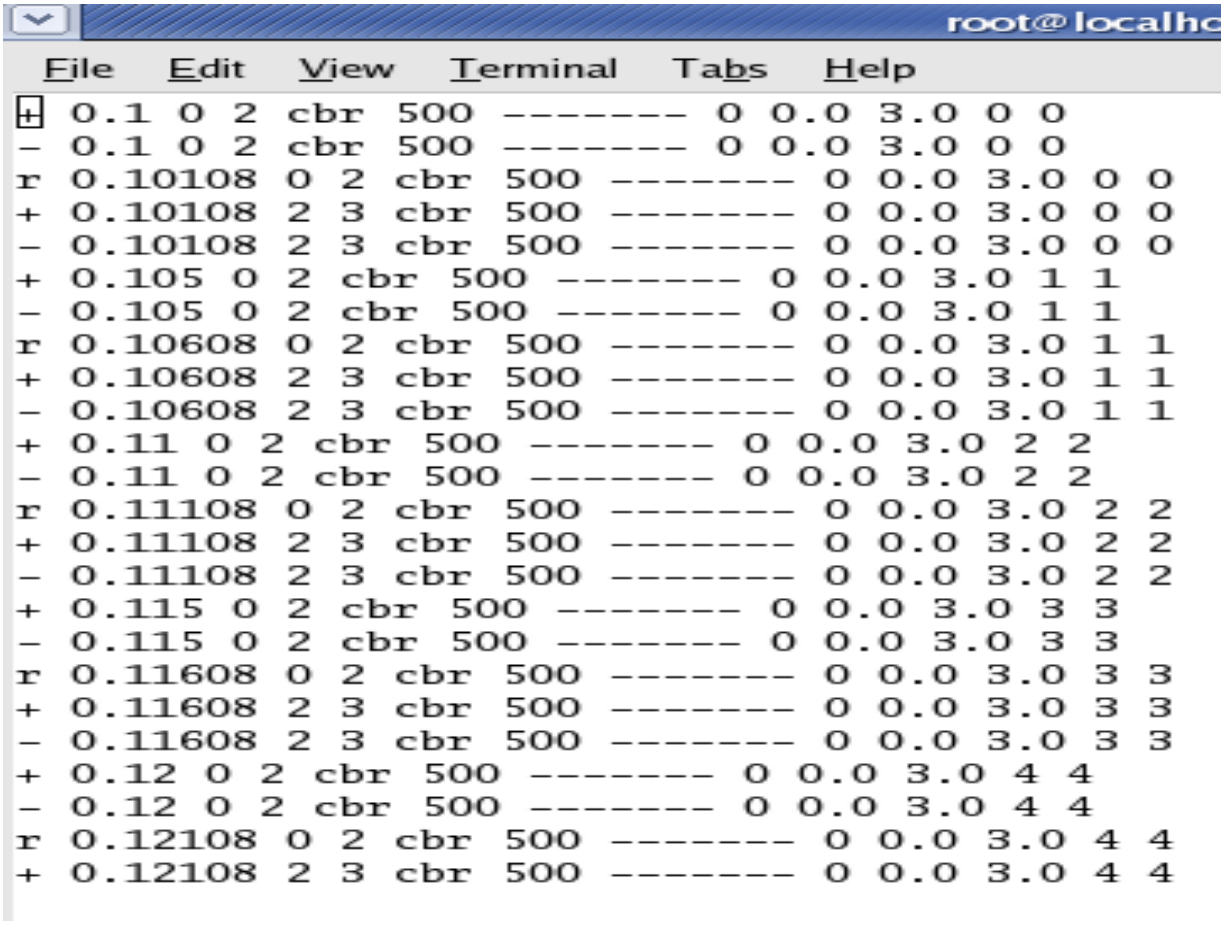
Step 5: After simulation is completed run **awk file** to see the output ,

[root@localhost~]# awk –f prog1.awk prog1.tr

Number of packets droped = 16

Step 6: To see the trace file contents open the file as ,

[root@localhost~]# vi prog1.tr



Experiment 2: PING MESSAGE

Aim: To implement transmission of ping messages/trace route over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

Step1: Open text editor, type the below program and save with extension .tcl (**prog2.tcl**)

```
set ns [new Simulator]
set nf [open prog2.nam w]
$ns namtrace-all $nf
set nd [open prog2.tr w]
$ns trace-all $nd

proc finish {} {
global ns nf nd
$ns flush-trace
close $nf
close $nd
exec nam prog2.nam &
exit 0
}
set n0 [$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]

$ns duplex-link $n1 $n0 1Mb 10ms DropTail
$ns duplex-link $n2 $n0 1Mb 10ms DropTail
$ns duplex-link $n3 $n0 1Mb 10ms DropTail
$ns duplex-link $n4 $n0 1Mb 10ms DropTail
$ns duplex-link $n5 $n0 1Mb 10ms DropTail
$ns duplex-link $n6 $n0 1Mb 10ms DropTail

Agent/Ping instproc recv {from rtt} {
$self instvar node_
puts "node [$node_ id] recieved ping answer from \
$from with round-trip-time $rtt ms."
}
set p1 [new Agent/Ping]
set p2 [new Agent/Ping]
set p3 [new Agent/Ping]
set p4 [new Agent/Ping]
set p5 [new Agent/Ping]
```

```
set p6 [new Agent/Ping]

$ns attach-agent $n1 $p1
$ns attach-agent $n2 $p2
$ns attach-agent $n3 $p3
$ns attach-agent $n4 $p4
$ns attach-agent $n5 $p5
$ns attach-agent $n6 $p6

$ns queue-limit $n0 $n4 3
$ns queue-limit $n0 $n5 2
$ns queue-limit $n0 $n6 2

$ns connect $p1 $p4
$ns connect $p2 $p5
$ns connect $p3 $p6

$ns at 0.2 "$p1 send"
$ns at 0.4 "$p2 send"
$ns at 0.6 "$p3 send"
$ns at 1.0 "$p4 send"
$ns at 1.2 "$p5 send"
$ns at 1.4 "$p6 send"
$ns at 2.0 "finish"
$ns run
```

Step2: Open text editor, type the below program and save with extension .awk (**prog3.awk**)

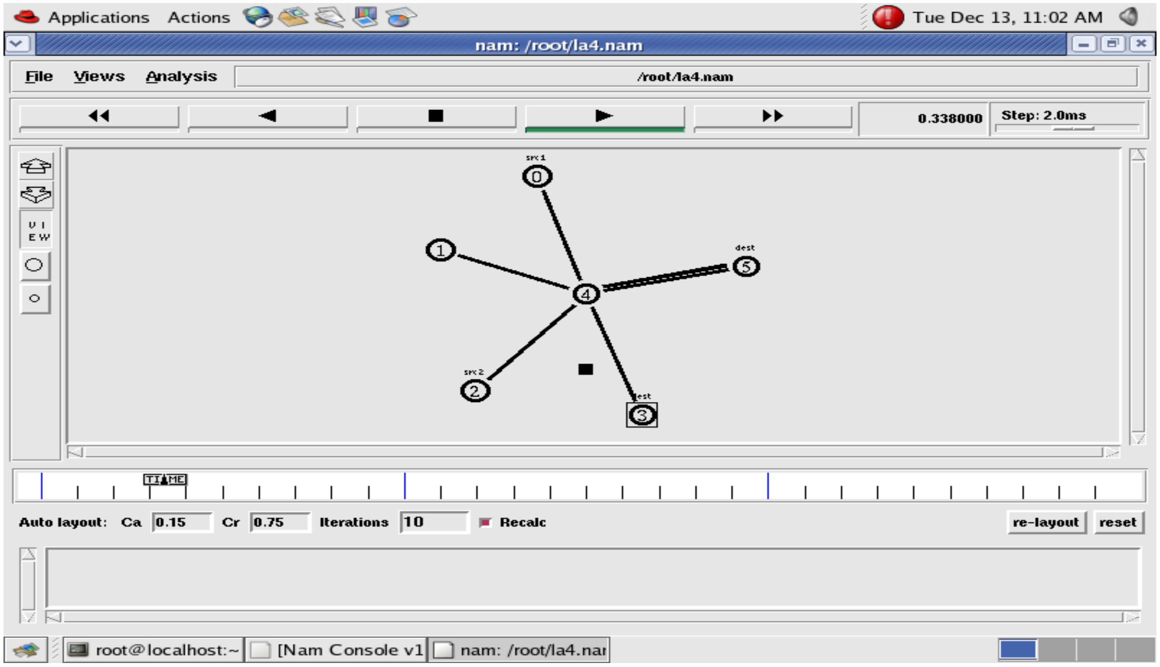
/*prog2.awk*/

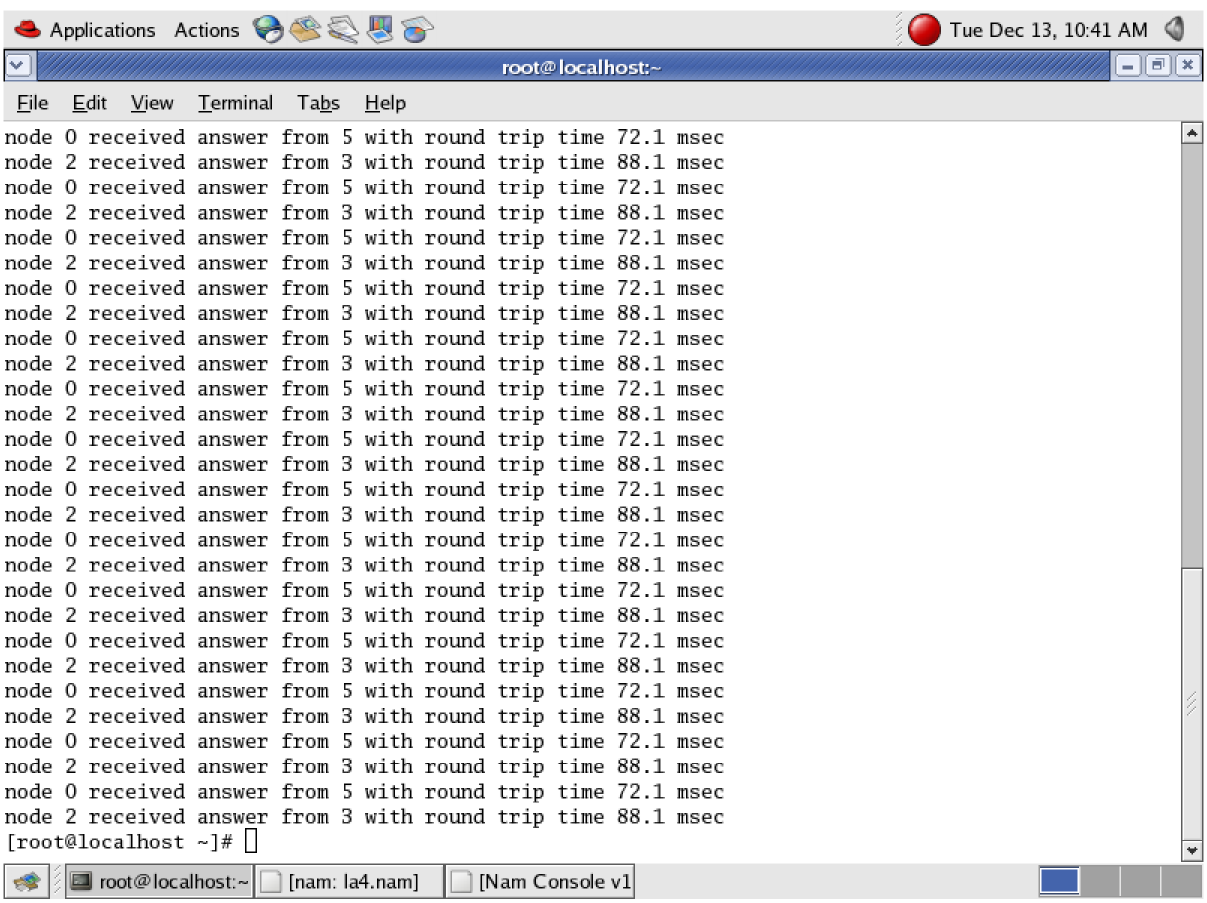
```
BEGIN {
count=0;
}
{
event=$1;
if(event=="d")
{
count++;
}
}
END {
printf("No of packets dropped : %d\n",count);
}
```

Step3: Run the simulation program

[root@localhost~]# ns prog2.tcl

(Here **"ns"** indicates network simulator. We get the topology shown in the snapshot.)

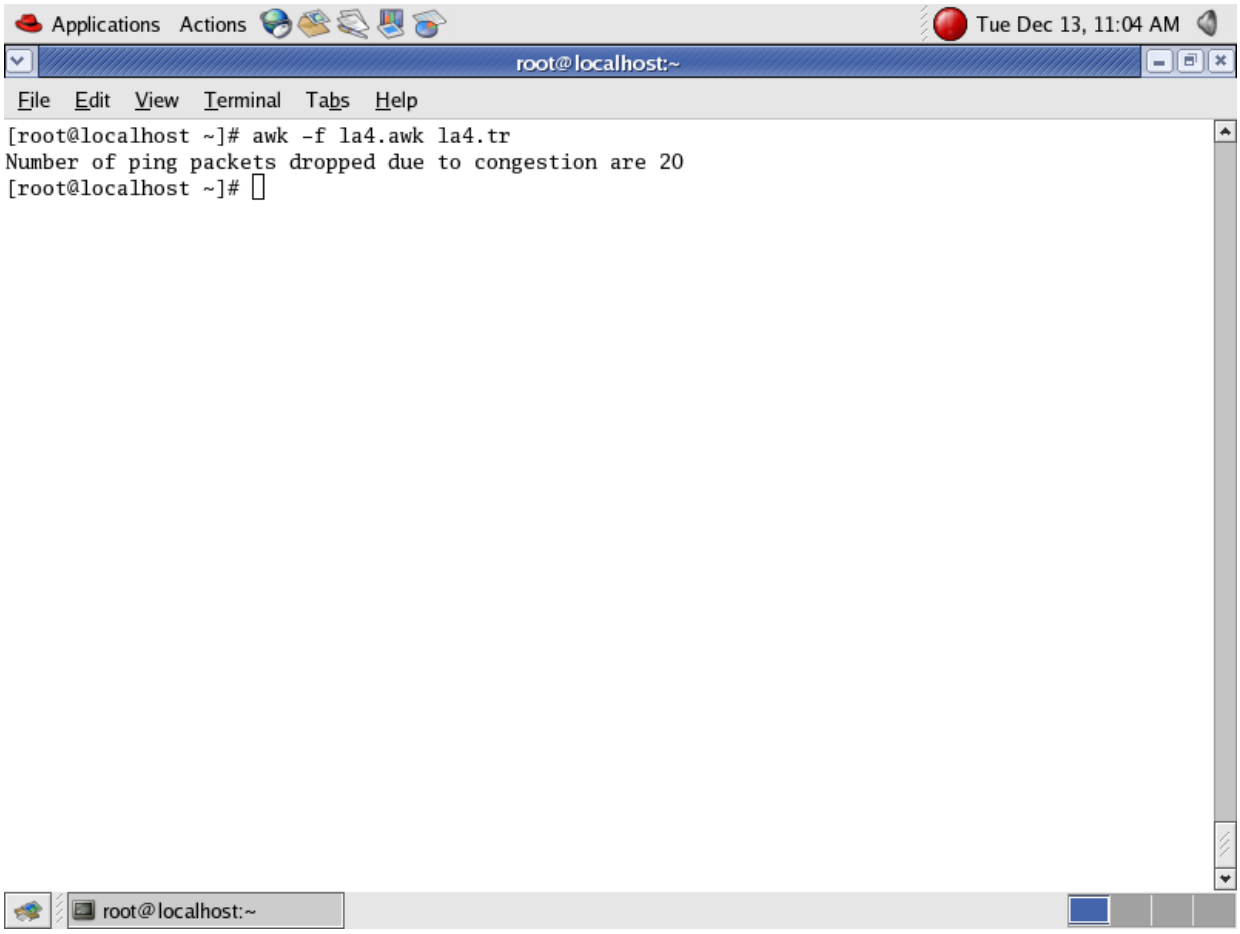




Step 4: Now press the play button in the simulation window and the simulation will begins.

Step 5: After simulation is completed run **awk file** to see the output ,

[root@localhost~]# awk –f prog2.awk prog2.tr



Step 6: To see the trace file contents open the file as ,

[root@localhost~]# vi prog2.tr

Experiment 3: ETHERNET LAN

Aim: To Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination.

Step1: Open text editor, type the below program and save with extension .tcl (**Lab3.tcl**)

```
set ns [new Simulator]

$ns color 1 Red
$ns color 2 Blue

set na [open Lab3.nam w]
$ns namtrace-all $na

set nt [open Lab3.tr w]
$ns trace-all $nt

set ng1 [open tcp1.xg w]
set ng2 [open tcp2.xg w]

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]

$ns make-lan "$n0 $n1 $n2" 1Mb 10ms LL Queue/DropTail Mac/802_3
$ns make-lan "$n3 $n4 $n5" 2Mb 10ms LL Queue/DropTail Mac/802_3

$ns duplex-link $n0 $n3 1Mb 10ms DropTail

set tcp1 [new Agent/TCP]
set tcp2 [new Agent/TCP]

set cbr1 [new Application/Traffic/CBR]
set cbr2 [new Application/Traffic/CBR]

$ns attach-agent $n4 $tcp1
$cbr1 attach-agent $tcp1

$ns attach-agent $n1 $tcp2
$cbr2 attach-agent $tcp2

set sink1 [new Agent/TCPSink]
set sink2 [new Agent/TCPSink]
```

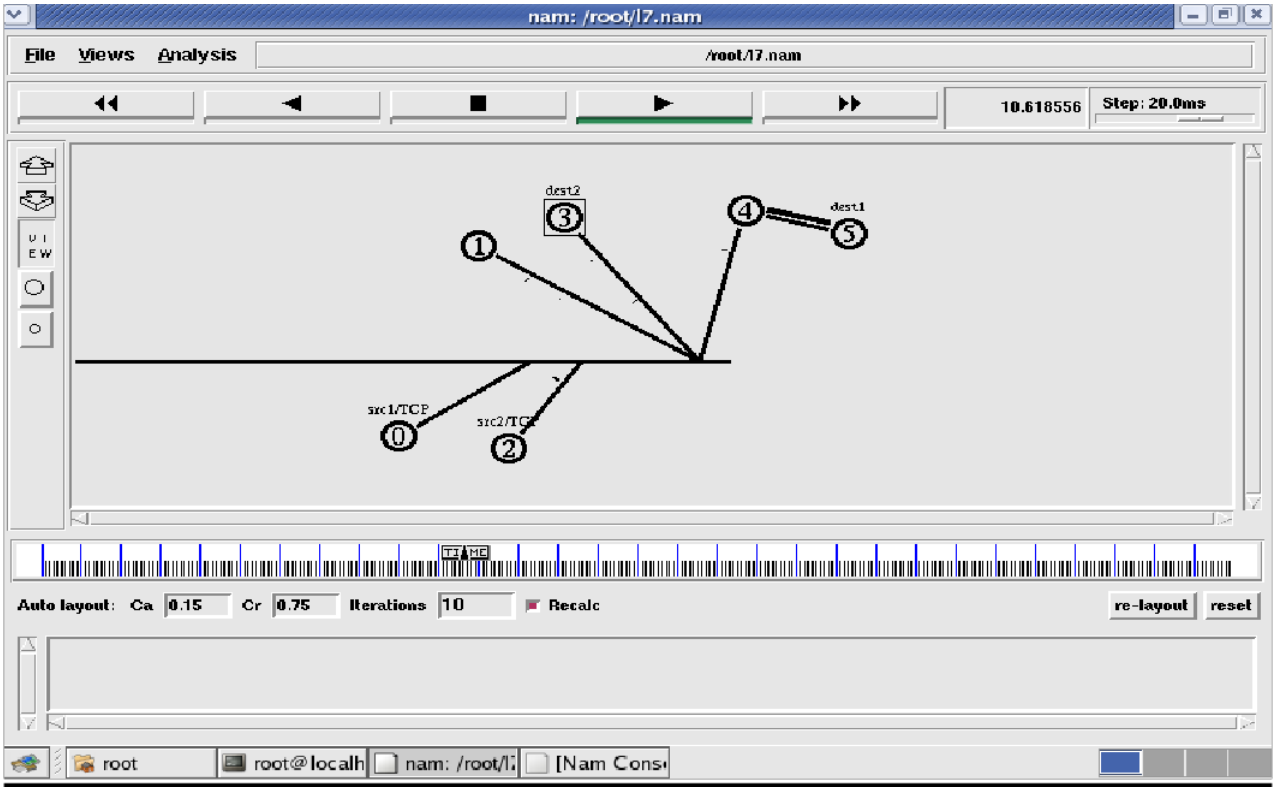
```
$ns attach-agent $n2 $sink1
$ns attach-agent $n5 $sink2

$ns connect $tcp1 $sink1
$ns connect $tcp2 $sink2

$tcp1 set class_ 1
$tcp2 set class_ 2

proc End {} {
        global ns na nt
        $ns flush-trace
        close $na
        close $nt
        exec nam Lab3.nam &
        exec xgraph tcp1.xg tcp2.xg &
        exit 0
}

proc Draw {Agent File} {
        global ns
        set Cong [$Agent set cwnd_]
        set Time [$ns now]
        puts $File "$Time $Cong"
        $ns at [expr $Time+0.01] "Draw $Agent $File"
}

$ns at 0.0 "$cbr1 start"
$ns at 0.7 "$cbr2 start"
$ns at 0.0 "Draw $tcp1 $ng1"
$ns at 0.0 "Draw $tcp2 $ng2"

$ns at 10.0 "End"
$ns run
```

Step2: Run the simulation program

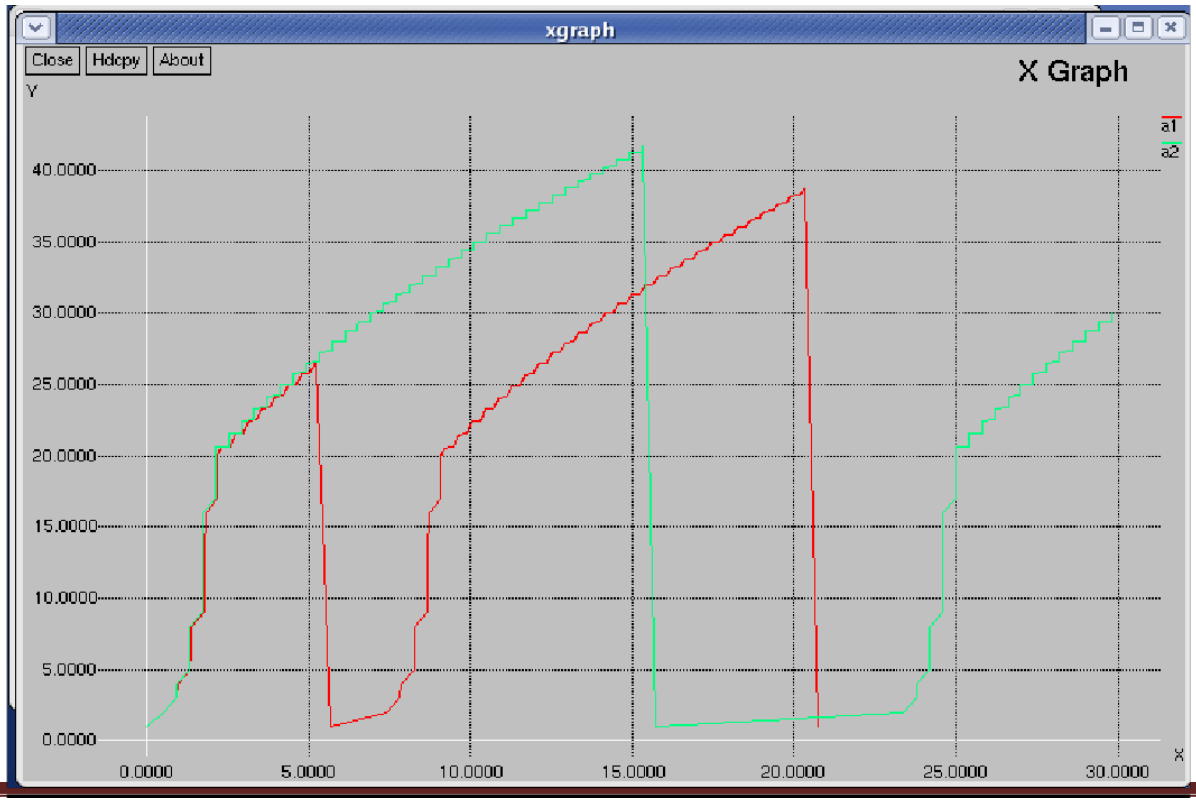
[root@localhost~]# ns Lab3.tcl

(Here **"ns"** indicates network simulator. We get the topology shown in the snapshot.)

Topology


nam: /root/l7.nam
File Views Analysis
/root/17.nam
10.618556
Step: 20.0ms
dest2
dest1
src1/TCP
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc
re-layout
reset
root
root@localh
[Nam Cons

Output

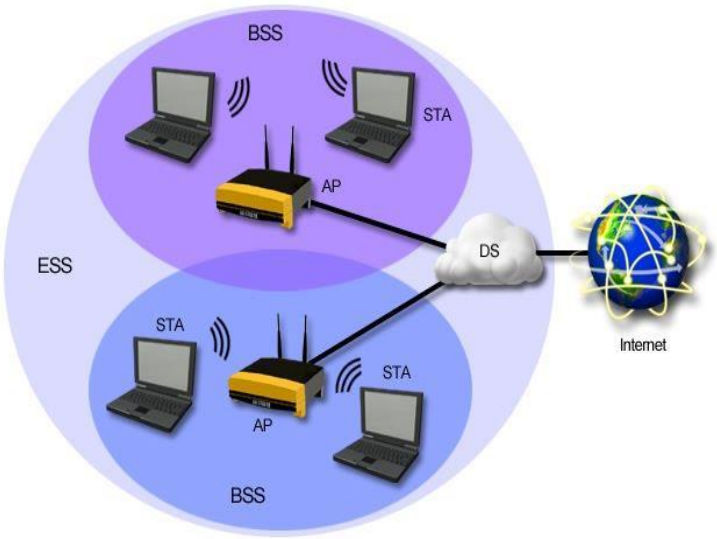

xgraph
Close
Hdcpy
About
X Graph
Y
a1
a2
40.0000
35.0000
30.0000
25.0000
20.0000
15.0000
10.0000
5.0000
0.0000
0.0000
5.0000
10.0000
15.0000
20.0000
25.0000
30.0000
X

Experiment 4: ESS

Aim: To implement simple ESS and with transmitting nodes in wire-less LAN by simulation and determine the performance with respect to transmission of packets.

Step1: Open text editor, type the below program and save with extension .tcl (**prog4.tcl**)

```
set ns [new Simulator]
set tf [open prog4.tr w]
$ns trace-all $tf
set topo [new Topography]
$topo load_flatgrid 1000 1000
set nf [open prog4.nam w]
$ns namtrace-all-wireless $nf 1000 1000
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) AODV ;
set val(x) 500 ;
set val(y) 400 ;
set val(stop) 10.0 ;

$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 OFF \
-movementTrace ON

create-god 3
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_ 50
$n0 set Y_ 50
$n0 set Z_ 0
$n1 set X_ 100
$n1 set Y_ 100
$n1 set Z_ 0
$n2 set X_ 600
$n2 set Y_ 600
$n2 set Z_ 0
$ns at 0.1 "$n0 setdest 50 50 15"
$ns at 0.1 "$n1 setdest 100 100 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"
$ns at 100 "$n1 setdest 550 550 15"
$ns at 190 "$n1 setdest 70 70 15"
proc finish { } {
        global ns nf tf
        $ns flush-trace
        exec nam prog6.nam  &
        close $tf
        exit 0
}
$ns at 250 "finish"
$ns run
```

Step2: Open text editor, type the below program and save with extension .awk (**prog6.awk**)

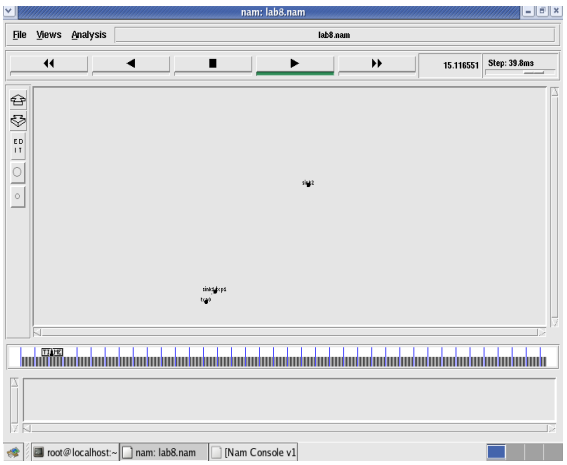
/*prog4.awk*/

```
BEGIN{
        count1=0
        count2=0
        pack1=0
        pack2=0
        time1=0
        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)));
}
```

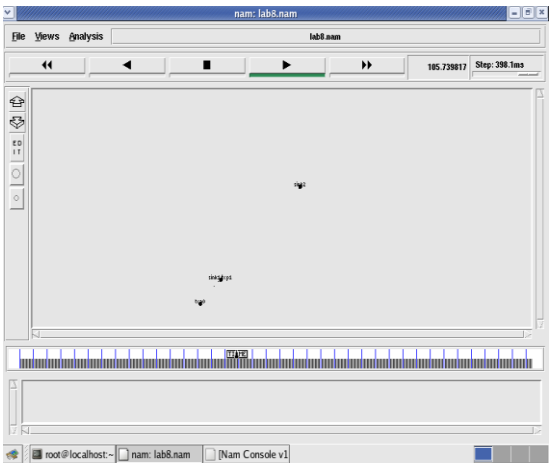
Step3: Run the simulation program

[root@localhost~]# ns prog4.tcl

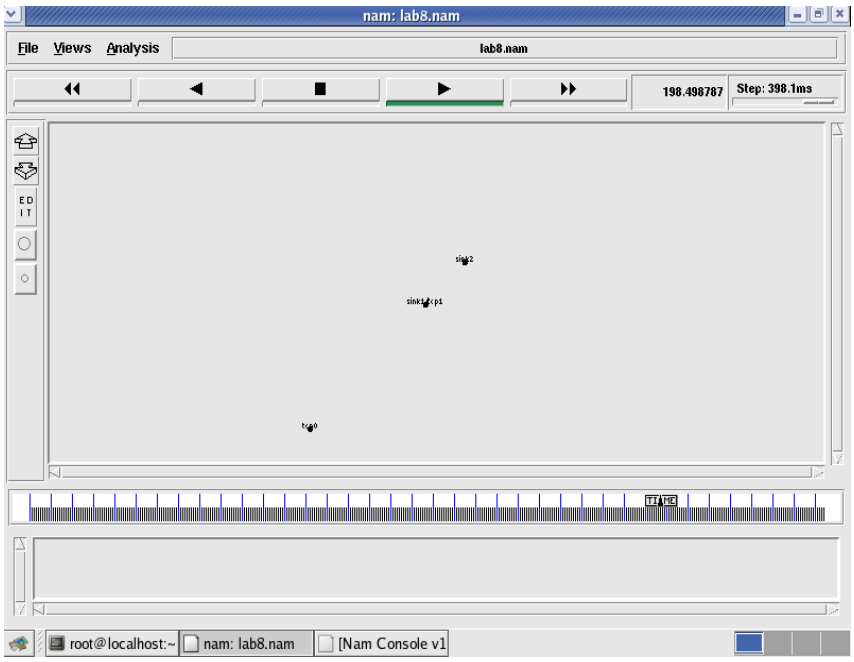
(Here **"ns"** indicates network simulator. We get the topology shown in the snapshot.)



Node 1 and 2 are communicating



Node 2 is moving towards node 3

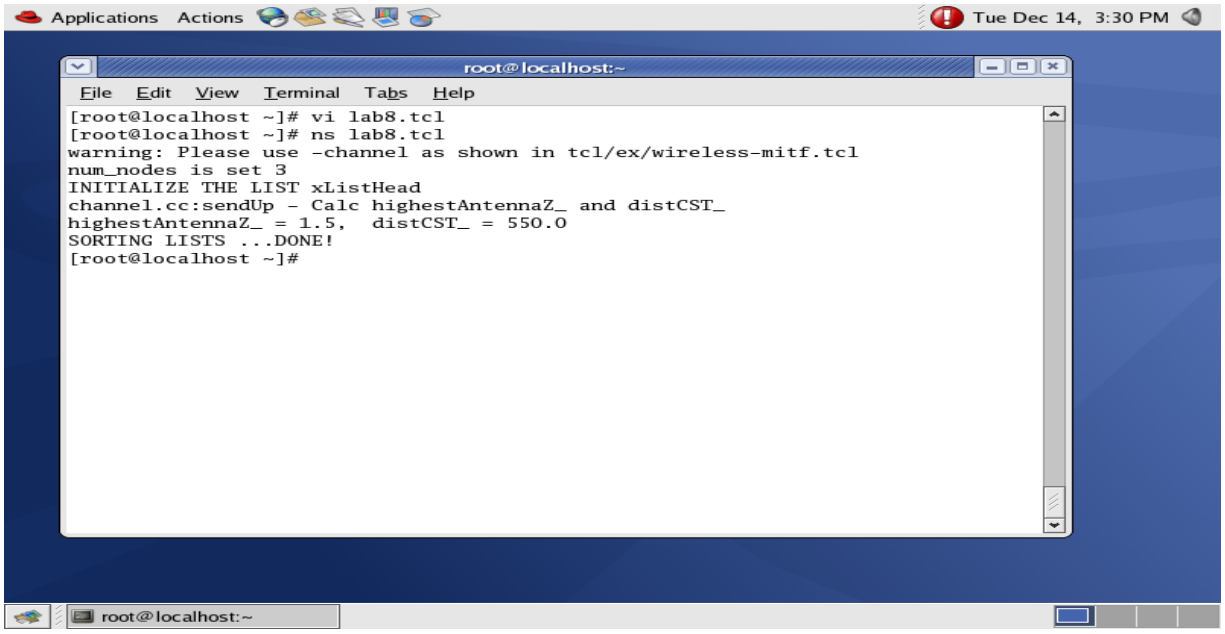


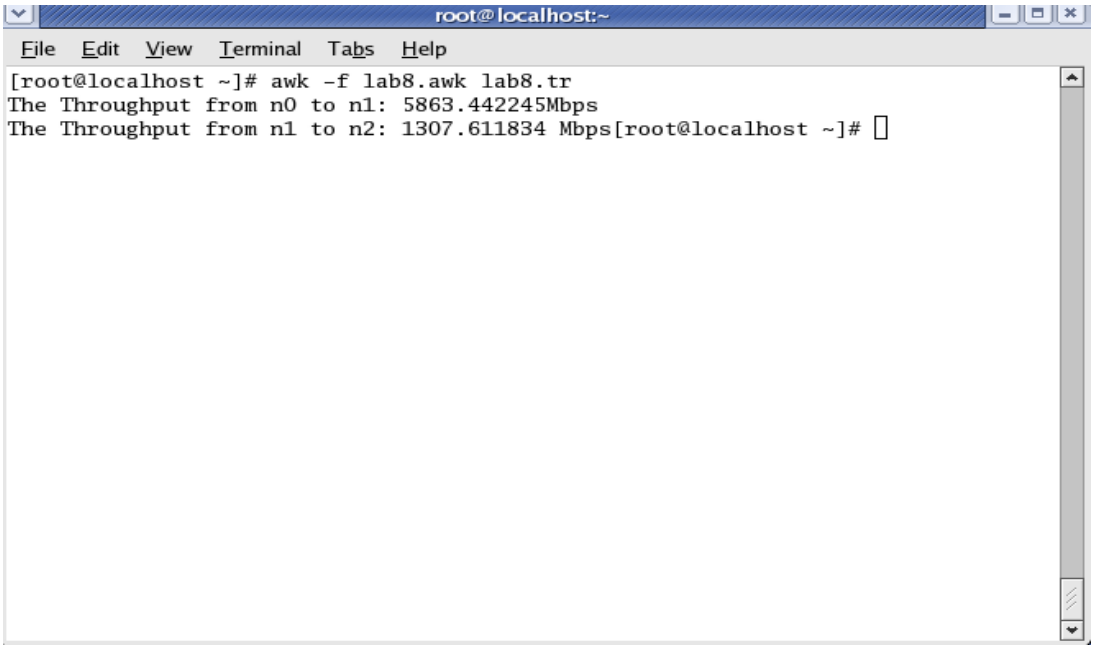
Node 2 is coming back from node 3 towards node1

Step 4: Now press the play button in the simulation window and the simulation will begins.

Step 5: After simulation is completed run **awk file** to see the output ,

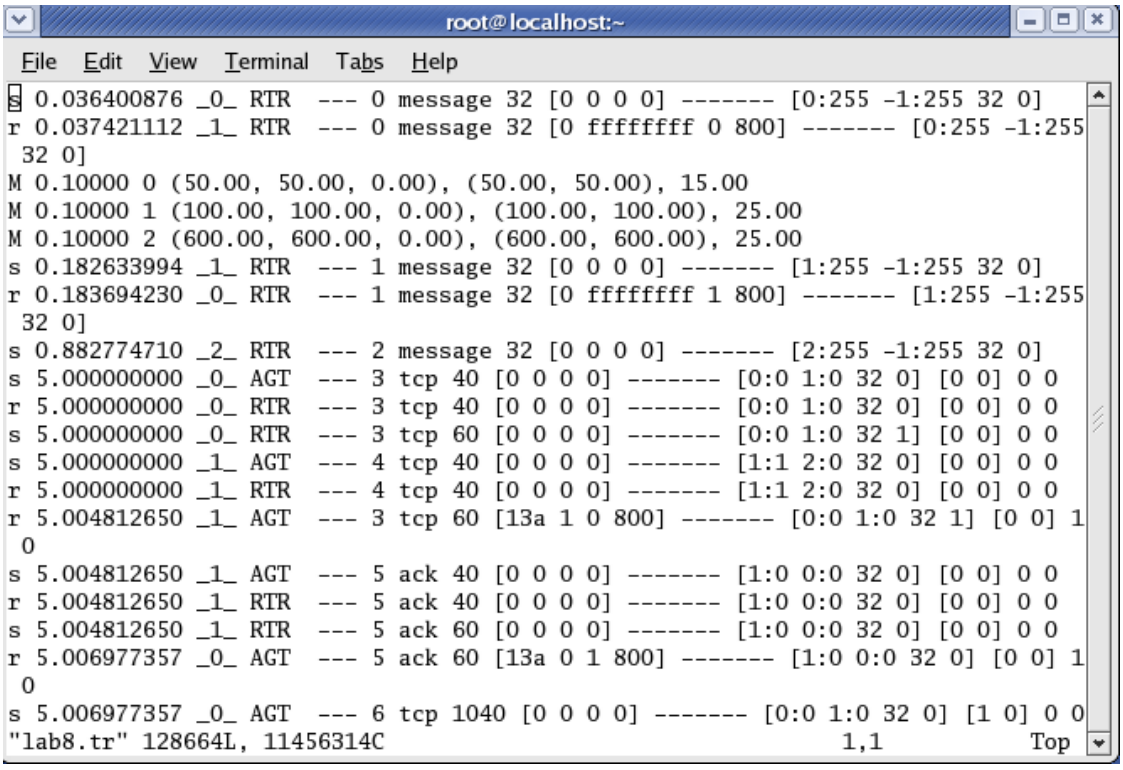
[root@localhost~]# awk –f prog4.awk prog4.tr





Step 6: To see the trace file contents open the file as ,

[root@localhost~]# vi prog2.tr



Here **"M"** indicates mobile nodes, **"AGT"** indicates Agent Trace, **"RTR"** indicates Router

Experiment 5: GSM

Aim: To implement and study the performance of GSM on NS2/NS3 (Using MAC layer) or equivalent environment.

Step1: Open text editor, type the below program and save with extension .tcl (**Lab5.tcl**)

```
set bwDL(gsm) 9600
set bwUL(gsm) 9600
set propDL(gsm) .500
set propUL(gsm) .500
set buf(gsm) 10
set Total 0

set ns [new Simulator]
set nt [open Lab5.tr w]
$ns trace-all $nt

set nodes(c1)  [$ns node]
set nodes(ms)  [$ns node]
set nodes(bs1) [$ns node]
set nodes(bs2) [$ns node]
set nodes(c2)  [$ns node]

proc cell_topo {} {
global ns nodes
$ns duplex-link $nodes(c1) $nodes(bs1) 3Mbps 10ms DropTail
$ns duplex-link $nodes(bs1) $nodes(ms) 1 1 RED
$ns duplex-link $nodes(ms) $nodes(bs2) 1 1 RED
$ns duplex-link $nodes(bs2) $nodes(c2) 3Mbps 50ms DropTail
}

switch gsm {
gsm -
gprs -
umts {cell_topo}
}
$ns bandwidth $nodes(bs1) $nodes(ms) $bwDL(gsm) simplex
$ns bandwidth $nodes(ms) $nodes(bs1) $bwUL(gsm) simplex
$ns bandwidth $nodes(bs2) $nodes(ms) $bwDL(gsm) simplex
$ns bandwidth $nodes(ms) $nodes(bs2) $bwUL(gsm) simplex

$ns delay $nodes(bs1) $nodes(ms) $propDL(gsm) simplex
$ns delay $nodes(ms) $nodes(bs1) $propDL(gsm) simplex
$ns delay $nodes(bs2) $nodes(ms) $propDL(gsm) simplex
$ns delay $nodes(ms) $nodes(bs2) $propDL(gsm) simplex
```

```
$ns queue-limit $nodes(bs1) $nodes(ms) $buf(gsm)
$ns queue-limit $nodes(ms) $nodes(bs1) $buf(gsm)
$ns queue-limit $nodes(bs2) $nodes(ms) $buf(gsm)
$ns queue-limit $nodes(ms) $nodes(bs2) $buf(gsm)

$ns insert-delayer $nodes(ms) $nodes(bs1) [new Delayer]
$ns insert-delayer $nodes(bs1) $nodes(ms) [new Delayer]
$ns insert-delayer $nodes(ms) $nodes(bs2) [new Delayer]
$ns insert-delayer $nodes(bs2) $nodes(ms) [new Delayer]

set tcp [new Agent/TCP]
$ns attach-agent $nodes(c1) $tcp

set sink [new Agent/TCPSink]
$ns attach-agent $nodes(c2) $sink

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

$ns connect $tcp $sink

proc End {} {
        global ns nt
        $ns flush-trace
        close $nt
        exec awk -f Lab5.awk Lab5.tr &
        exec xgraph -P -bar -x TIME -y DATA gsm.xg &
        exit 0
}

$ns at 0.0 "$ftp start"
$ns at 10.0 "End"
$ns run
```

Step2: Open text editor, type the below program and save with extension .awk (**Lab5.awk**)

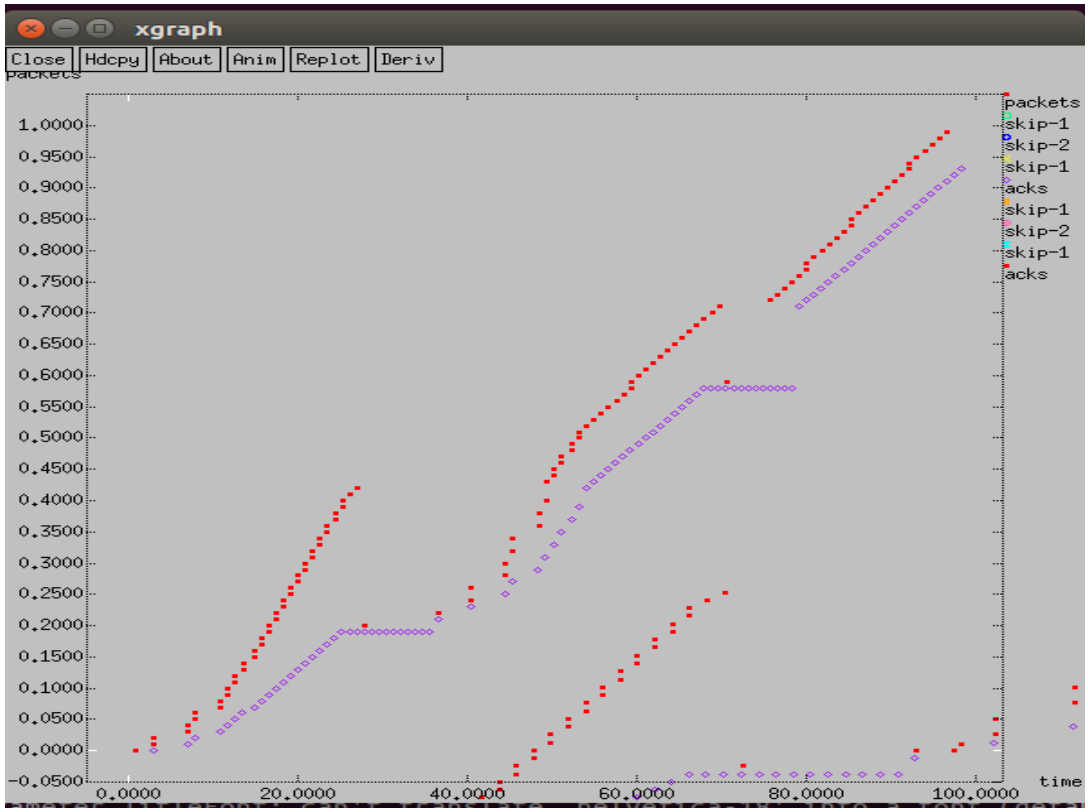
```
/*Lab5.awk*/
BEGIN {Total_no_of_pkts=0;}
{
if($1 == "r")
{
Total_no_of_pkts = Total_no_of_pkts + $6;
printf("%f %d\n",$2,Total_no_of_pkts) >> "gsm.xg"
}
}
END{}
```

Step3: Run the simulation program

[root@localhost~]# ns Lab5.tcl

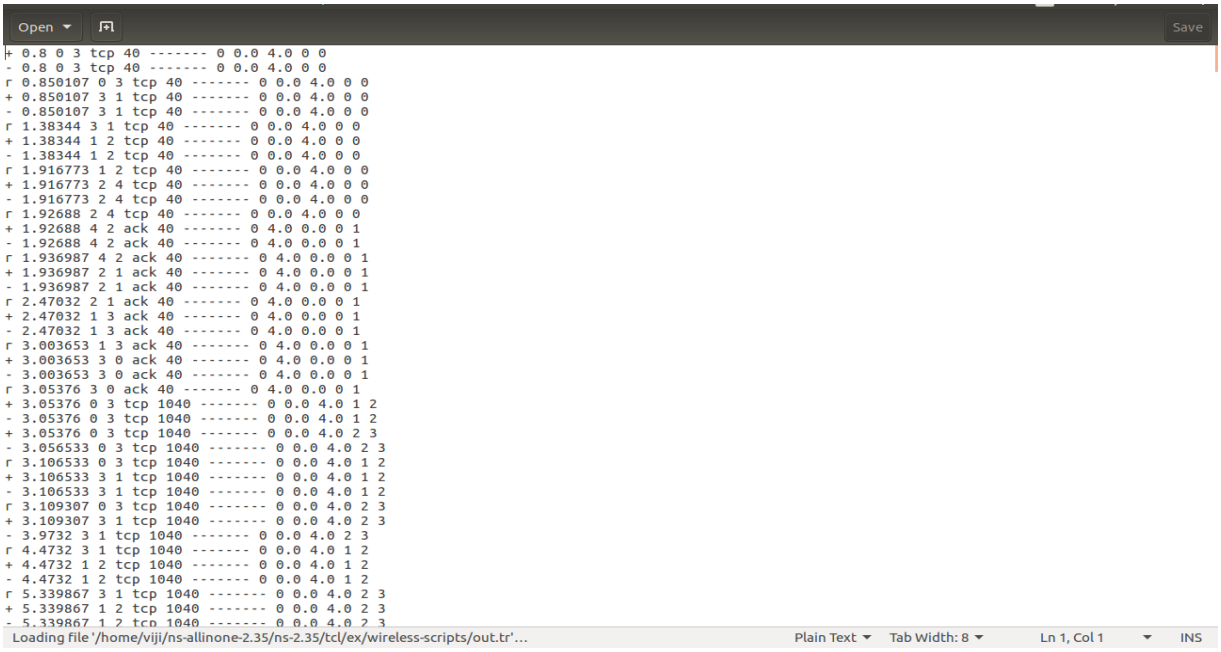
(Here **"ns"** indicates network simulator. We get the topology shown in the snapshot.)

Step 4: Now press the play button in the simulation window and the simulation will begins.



Step 5: After simulation is completed run **awk file** to see the output ,

[root@localhost~]# awk –f Lab5.awk Lab5.tr



Experiment 6: CDMA

Aim: To implementandstudytheperformanceofCDMAonNS2/NS3(UsingstackcalledCall net) or equivalent environment.

Step1: Open text editor, type the below program and save with extension .tcl (**Lab6.tcl**)

```
set bwDL(cdma) 384000
set bwUL(cdma) 64000
set propDL(cdma) .150
set propUL(cdma) .150
set buf(cdma) 20

set ns [new Simulator]

set nt [open Lab6.tr w]
$ns trace-all $nt

set nodes(c1) [$ns node]
set nodes(ms) [$ns node]
set nodes(bs1) [$ns node]
set nodes(bs2) [$ns node]
set nodes(c2) [$ns node]

proc cell_topo {} {
global ns nodes
$ns duplex-link $nodes(c1) $nodes(bs1) 3Mbps 10ms DropTail
$ns duplex-link $nodes(bs1) $nodes(ms) 1 1 RED
$ns duplex-link $nodes(ms) $nodes(bs2) 1 1 RED
$ns duplex-link $nodes(bs2) $nodes(c2) 3Mbps 50ms DropTail
}

switch umts {
cdma -
umts {cell_topo}
}
$ns bandwidth $nodes(bs1) $nodes(ms) $bwDL(cdma) simplex
$ns bandwidth $nodes(ms) $nodes(bs1) $bwUL(cdma) simplex
$ns bandwidth $nodes(bs2) $nodes(ms) $bwDL(cdma) simplex
$ns bandwidth $nodes(ms) $nodes(bs2) $bwUL(cdma) simplex

$ns delay $nodes(bs1) $nodes(ms) $propDL(cdma) simplex
$ns delay $nodes(ms) $nodes(bs1) $propDL(cdma) simplex
$ns delay $nodes(bs2) $nodes(ms) $propDL(cdma) simplex
$ns delay $nodes(ms) $nodes(bs2) $propDL(cdma) simplex

$ns queue-limit $nodes(bs1) $nodes(ms) $buf(cdma)
```

```
$ns queue-limit $nodes(ms) $nodes(bs1) $buf(cdma)
$ns queue-limit $nodes(bs2) $nodes(ms) $buf(cdma)
$ns queue-limit $nodes(ms) $nodes(bs2) $buf(cdma)

$ns insert-delayer $nodes(ms) $nodes(bs1) [new Delayer]
$ns insert-delayer $nodes(bs1) $nodes(ms) [new Delayer]
$ns insert-delayer $nodes(ms) $nodes(bs2) [new Delayer]
$ns insert-delayer $nodes(bs2) $nodes(ms) [new Delayer]

set tcp [new Agent/TCP]
$ns attach-agent $nodes(c1) $tcp

set sink [new Agent/TCPSink]
$ns attach-agent $nodes(c2) $sink

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

$ns connect $tcp $sink

proc End {} {
        global ns nt
        $ns flush-trace
        close $nt
        exec awk -f Lab6.awk Lab6.tr &
        exec xgraph -P -bar -x TIME -y DATA cdma.xg &
        exit 0
}

$ns at 0.0 "$ftp start"
$ns at 10.0 "End"
$ns run
```

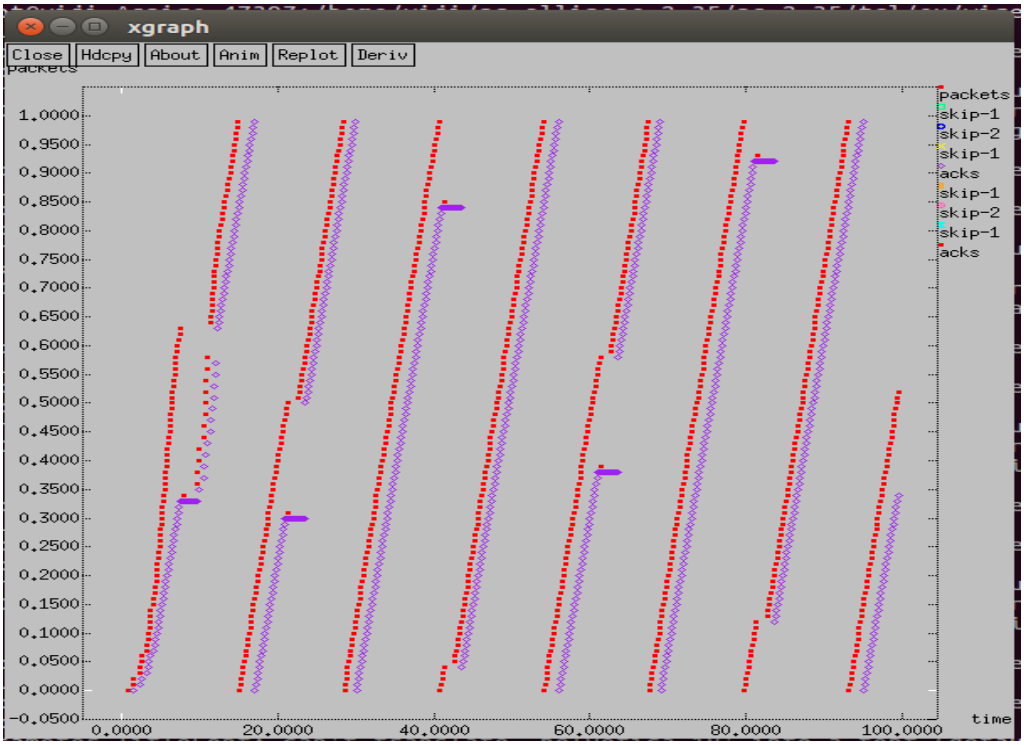
Step2: Open text editor, type the below program and save with extension .awk (**Lab6.awk**)

```
    /*Lab6.awk*/
BEGIN {Total_no_of_pkts=0;}
{
if($1 == "r")
{
Total_no_of_pkts = Total_no_of_pkts + $6;
printf("%f %d\n",$2,Total_no_of_pkts) >> "cdma.xg"
}
}
END{}
```

Step3: Run the simulation program

[root@localhost~]# ns Lab6.tcl

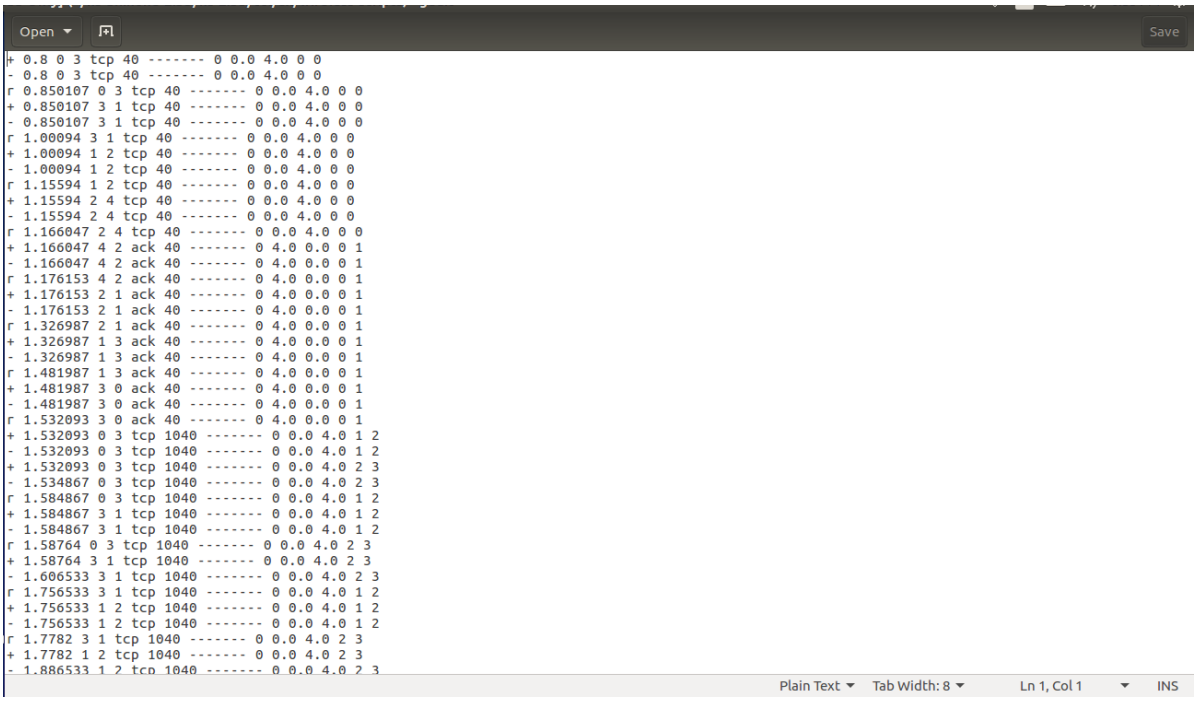
(Here **"ns"** indicates network simulator. We get the topology shown in the snapshot.)



Step 4: Now press the play button in the simulation window and the simulation will begins.

Step 5: After simulation is completed run **awk file** to see the output ,

[root@localhost~]# awk –f Lab5.awk Lab5.tr



JAVA PROGRAMS: PART B

Java is a general-purpose language that is simple, concurrent, class object language. The compiled Java code can run on all platforms that support Java without the need for recompilation hence Java is called as "write once" (WORA).The Java compiled intermediate output called "byte-code" that can run on any Java virtual machine (JVM) regardless of computer architecture. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

In Linux operating system Java libraries are preinstalled. It's very easy and convenient to compile and run Java programs in Linux environment. To compile and run Java Program is a two-step process:

1. Compile Java Program from CommandPrompt
   [root@host ~]# javac Filename.java

   The Java compiler (Javac) compiles java program and generates a byte-code with the same file name and .class extension.

2. Run Java program from CommandPrompt
   [root@host ~]# java Filename

   The java interpreter (Java) runs the byte-code and gives the respective output. It is important to note that in above command we have omitted the .class suffix of thebyte-code(Filename.class).

Experiment 7: CRC-CCITT

Aim: To write a program for error detecting code using CRC-CCITT (16 bits).

Theory

CRC(Cyclic Redundancy Check) is an error detecting technique used in digital networks and storage devices to detect the accidental changes to raw data. It cannot be used for correcting errors.

If an error is detected in the received message, a 'Negative acknowledgement' is sent to the sender. The sender and the receiver agree upon a fixed polynomial called generator polynomial. The standard agreed generator polynomial is $x^{16}+x^{12}+x^5+x^0$ (any polynomial can be considered, of degree 16).

The CRC does error checking via polynomial division. The generated polynomial g(x) = $x^{16}+x^{12}+x^5+x^0$

16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1

→ 17 bits.

So the g(x) value is 10001000000100001

Algorithm:

1. Given a bit string (message to be sent), append 16 0^S to the end of it (the number of 0^s is the same as the degree of the generator polynomial) let this string + 0^S be called as modified string B
2. Divide B by agreed on polynomial g(x) and determine the remainder R(x). The 16-bit remainder received is called as checksum.
3. The message string is appended with checksum and sent to the receiver.
4. At the receiver side, the received message is divided by generator polynomial g(x).
5. If the remainder is 0, the receiver concludes that there is no error occurred otherwise, the receiver concludes an error occurred and requires a retransmission.

Program:

```
import java.util.*;
public class crc {

    public static void main(String args[]){
        Scanner s=new Scanner(System.in);
```

```
        int n;
        System.out.println("enter the size of the data:");
        n=s.nextInt();
        int data[]=new int[n];
        System.out.println("enter the data ,bit by bit:");
        for(int i=0; i < n; i++){
                data[i]=s.nextInt();
        }
        System.out.println("enter ther size of the divisor:");
        n=s.nextInt();
        int divisor[]=new int[n];
        System.out.println("enter the divisor,bit by bit:");
        for(int i=0; i < n; i++)
                divisor[i]=s.nextInt();
        int remainder[]=divide(data,divisor);
        System.out.println("\n the crc code generated is:");
        for(int i=0; i < data.length; i++)
                System.out.print(data[i]);
        for(int i=0; i < remainder.length-1; i++)
                System.out.print(remainder[i]);
        System.out.println();
        int sent_data[]=new int[data.length+remainder.length-1];
        System.out.println("enter the data to be sent:");
        for(int i=0; i < sent_data.length; i++)
                sent_data[i]=s.nextInt();
        recieve(sent_data,divisor);
}
static int[] divide(int old_data[],int divisor[]){
        int remainder[],i;
        int data[]=new int[old_data.length+divisor.length];
        System.arraycopy(old_data, 0,data, 0,old_data.length);
        System.out.println("message bits after appending divisor_length-1 0's:");
        for(i=0; i<data.length-1; i++)
                System.out.println(data[i]); remainder=new
        int[divisor.length]; System.arraycopy(data, 0,
        remainder, 0, divisor.length); for(i=0; i <
        old_data.length; i++){
                if(remainder[0]==1){
                        for(int j=1; j<divisor.length; j++){
                                remainder[j-1]=exor(remainder[j],divisor[j]);
```

```
                        }
                    }
                    else{
                        for(int j=1; j < divisor.length; j++)
                            remainder[j-1]=exor(remainder[j],0);
                    }

                remainder[divisor.length-1]=data[i+divisor.length];
                }
                return remainder;
        }
        static int exor(int a,int b){
                if(a==b)
                    return 0;
                return 1;
        }
        static void recieve(int data[],int divisor[]){ int
                remainder[]=divide(data,divisor);
                for(int i=0; i < remainder.length; i++){
                    if(remainder[i] != 0){
                        System.out.println("there is an error in recieved data...");
                        return;
                    }
                }
                System.out.println("data was recieved without any error");

        }
}
```

```
File  Edit  View  Terminal  Help
[root@localhost ~]# javac Crc.java
[root@localhost ~]# java Crc
Enter number of data bits :
7
Enter data bits :
1
0
1
1
0
0
1
Enter number of bits in divisor :
3
Enter Divisor bits :
1
0
1
Dividend (after appending 0's) are : 101100100

CRC code :
101100111
Enter CRC code of 9 bits :
1
0
1
1
0
0
1
0
1
Error
THANK YOU.... :)
[root@localhost ~]#
```

Experiment 8: BELLMAN-FORD

Aim: To write a program to find the shortest path between vertices using bellman-ford algorithm.

Theory

Routing algorithm is a part of network layer software which is responsible for deciding which output line an incoming packet should be transmitted on. If the subnet uses datagram internally, this decision must be made anew for every arriving data packet since the best route may have changed since last time. If the subnet uses virtual circuits (connection Oriented), routing decisions are made only when a new established route is being set up.

Routing algorithms can be grouped into two major classes: adaptive and nonadaptive. Nonadaptive algorithms do not base their routing decisions on measurement or estimates of current traffic and topology. Instead, the choice of route to use to get from *I* to *J* (for all *I* and *J*) is compute in advance, offline, and downloaded to the routers when the network ids booted. This procedure is sometime called static routing.

Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and usually the traffic as well. Adaptive algorithms differ in where they get information (e.g., locally, from adjacent routers, or from all routers), when they change the routes (e.g., every ΔT sec, when the load changes, or when the topology changes), and what metric is used for optimization (e.g., distance, number of hops, or estimated transit time).

Two algorithms in particular, distance vector routing and link state routing are the most popular. Distance vector routing algorithms operate by having each router maintain a table (i.e., vector) giving the best known distance to each destination and which line to get there. These tables are updated by exchanging information with the neighbors.

The distance vector routing algorithm uses Bellman-Ford routing algorithm and Ford-Fulkerson algorithm. In distance vector routing, each router maintains a routing table that contains two parts: the preferred out going line to use for that destination, and an estimate of the time or distance to that destination. The metric used might be number of hops, time delay in milliseconds, total number of packets queued along the path, or something similar.

The Routing tables are shared among the neighbors, and the tables at the router are updated, such that the router will know the shortest path to the destination.

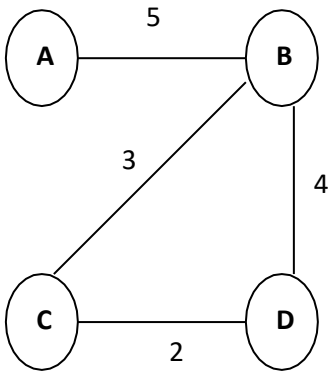
Program:

```
import java.util.Scanner;
public class BellmanFord {
        private int D[];
        private int num_ver;

        public static final int MAX_VALUE=999;

        public BellmanFord(int num_ver){
                this.num_ver=num_ver;
```

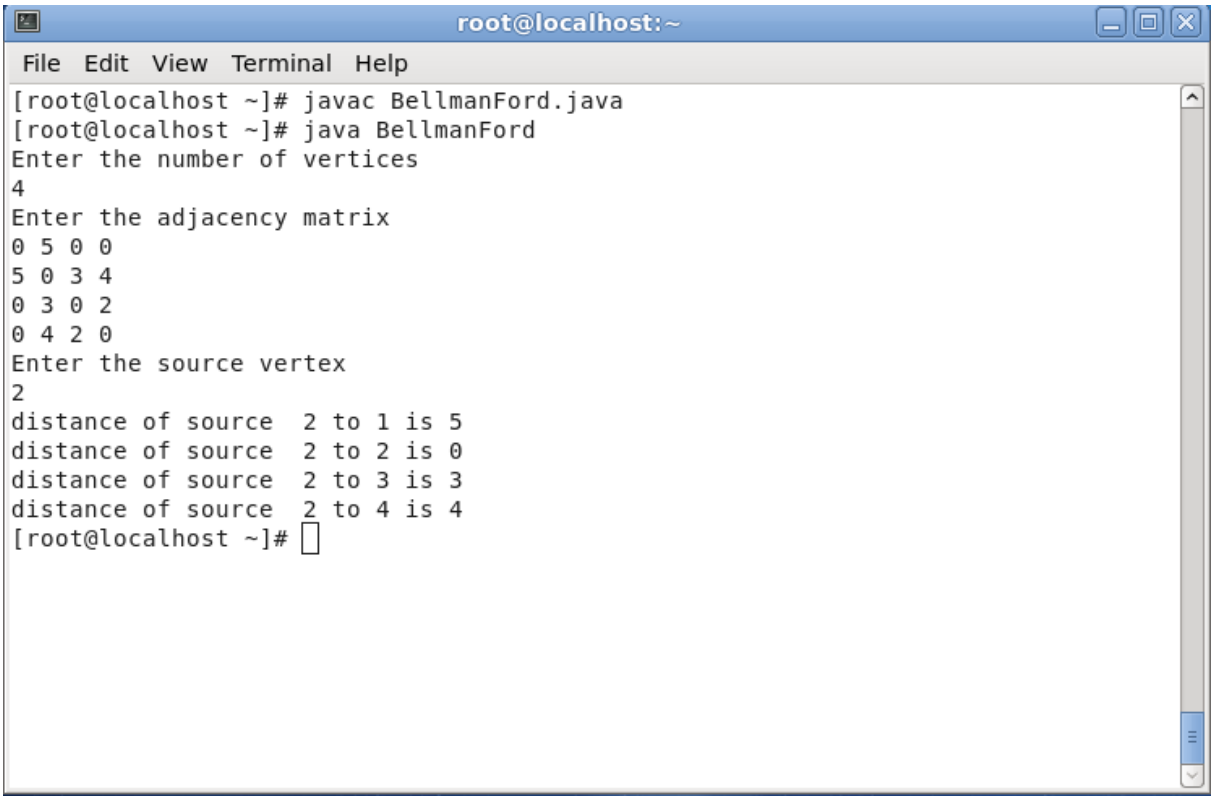
```
            D = new int[num_ver+1];
        }
        public void BellmanFordEvaluation(int source,int a[][]){
            for(int node=1 ;node <= num_ver ;node++){
                D[node]=MAX_VALUE;
            }
            D[source]=0;
            for(int node=1 ;node <= num_ver; node++){
                for(int sn=1; sn <= num_ver; sn++){
                    for(int dn=1; dn <= num_ver; dn++){
                        if(a[sn][dn] < 0)
                            D[dn]=0;
                        else{
                            if(a[sn][dn] != MAX_VALUE){
                                if(D[dn] > D[sn]+a[sn][dn])
                                    D[dn] = D[sn]+a[sn][dn];
                            }
                        }
                    }
                }
            }
            for(int sn=1; sn <= num_ver; sn++){
                for(int dn=1; dn <= num_ver; dn++){
                    if(a[sn][dn] != MAX_VALUE){
                        if(D[dn] > D[sn]+a[sn][dn]){
                            System.out.println("the graph contains negative edge
cycle");
                        }
                    }
                }
            }
            for(int vertex=1; vertex <= num_ver; vertex++){
                System.out.println("distance of source "+ source + " to "+ vertex +" is
"+D[vertex]);

            }

        }
        public static void main(String args[]){
            int num_ver=0,source;
```

```
                Scanner s=new Scanner(System.in);
                System.out.println("enter the number of vertex");
                num_ver=s.nextInt();
                int a[][]=new int[num_ver+1][num_ver+1];
                System.out.println("enter the adjacency matrix");
                for(int sn=1; sn <= num_ver; sn++){
                        for(int dn=1; dn <=num_ver; dn++){
                                a[sn][dn]=s.nextInt();
                                if(sn==dn){
                                        a[sn][dn]=0;
                                        continue;
                                }
                                if(a[sn][dn]==0){
                                        a[sn][dn]=MAX_VALUE;
                                }
                        }
                }
                System.out.println("enter the source vertex");
                source=s.nextInt();
                BellmanFord b = new BellmanFord(num_ver);
                b.BellmanFordEvaluation(source, a);
                s.close();

        }

}
```

Input graph:



Output:



[root@localhost ~]# vi BellmanFord.java

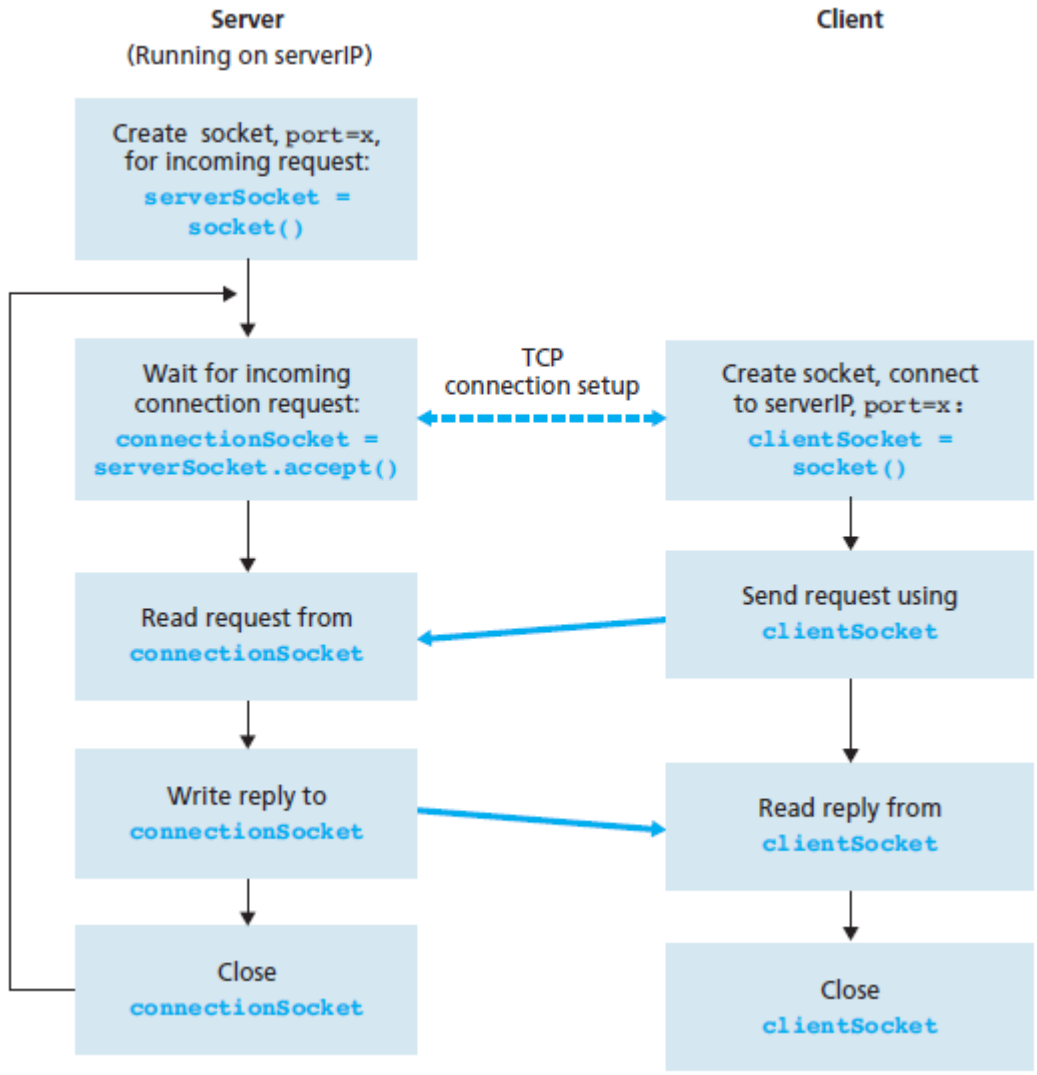
Experiment 9: TCP/IP SOCKET FOR CLIENT-SERVER COMMUNICATION

Aim: 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. Implement the above program using as message queues or FIFOs as IPC channels.

Theory:

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, theservercreatesasocketobjectonitsendofthecommunication.Theclientandthe server can now communicate by writing to and reading from the socket.

Procedure:



/*TCP Server program*/

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

public class tcpser {
	public static void main(String args[])throws Exception
	{
		ServerSocket sersock= new ServerSocket(4000);
		System.out.println("sever ready for connection");
		Socket sock=sersock.accept();
		System.out.println("connection is successfull and waiting to serve");
		InputStream istream=sock.getInputStream();
		BufferedReader fileRead=new BufferedReader(newInputStreamReader(istream));
		String fname=fileRead.readLine();
		BufferedReader contentRead=new BufferedReader(new FileReader(fname));
		OutputStream ostream=sock.getOutputStream();
		PrintWriter pwrite=new PrintWriter(ostream,true);
		String str;
		while((str=contentRead.readLine())!=null)
		{
			pwrite.println(str);
		}
	}

}
```

/*TCP Client program*/

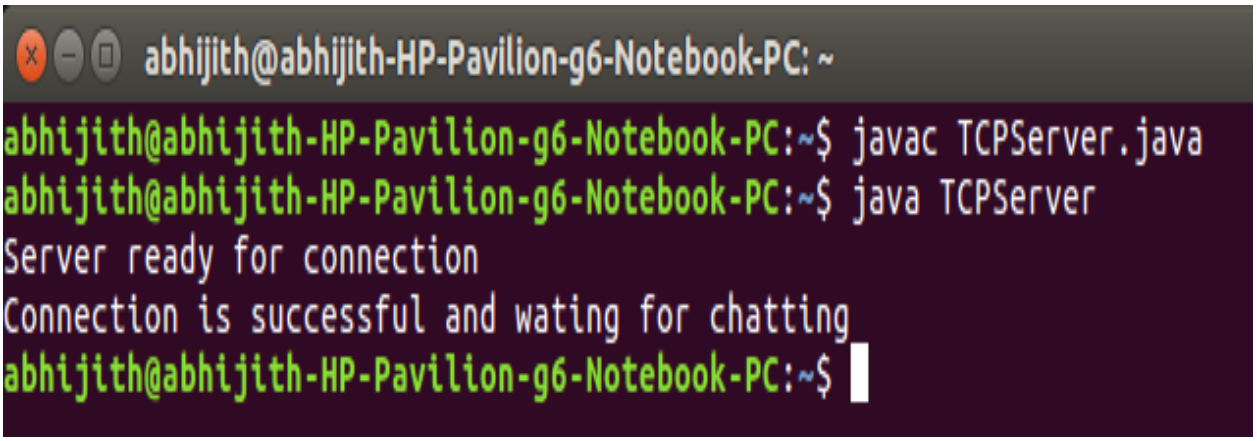
```
import java.net.*;
import java.io.*;

public class tcpcln {
public static void main(String args[])throws Exception
{
	Socket sock=new Socket("127.0.0.1",4000);
	System.out.println("enter the file name");
	BufferedReader keyRead=new BufferedReader(new InputStreamReader(System.in));
	String fname=keyRead.readLine();
	OutputStream ostream=sock.getOutputStream();
	PrintWriter pwrite=new PrintWriter(ostream,true);
	pwrite.println(fname);
	InputStream istream=sock.getInputStream();
	BufferedReader SocketRead=new BufferedReader(new InputStreamReader(istream));
	String str;
```

```
        while((str=SocketRead.readLine())!=null)
        {
                System.out.println(str);
                pwrite.close();
                SocketRead.close();
                keyRead.close();
        }
}
}
```

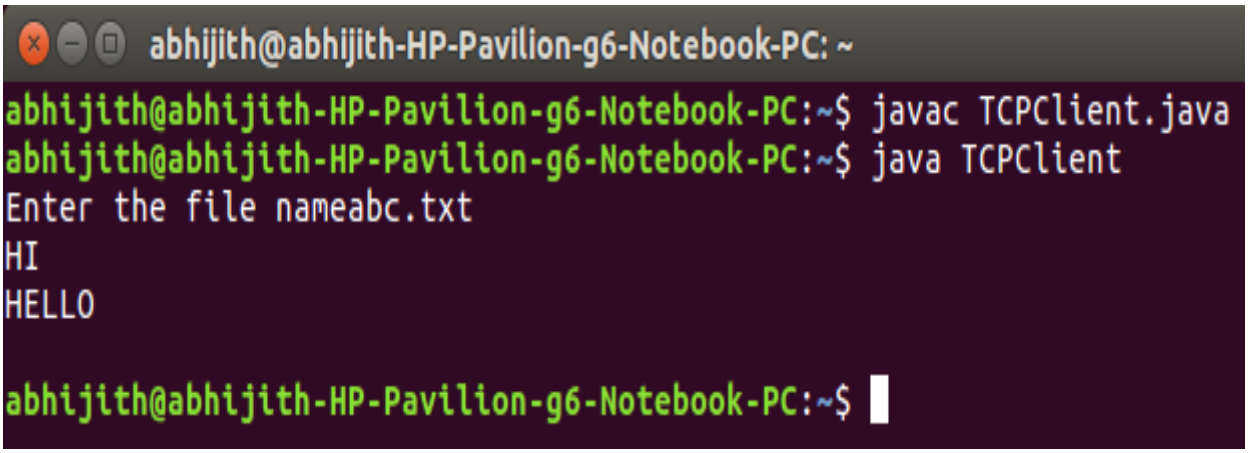
OUTPUT:

Create a text file say abc.txt and type some content in it.

Compile and execute server side program



Open new terminal, compile and execute client side program



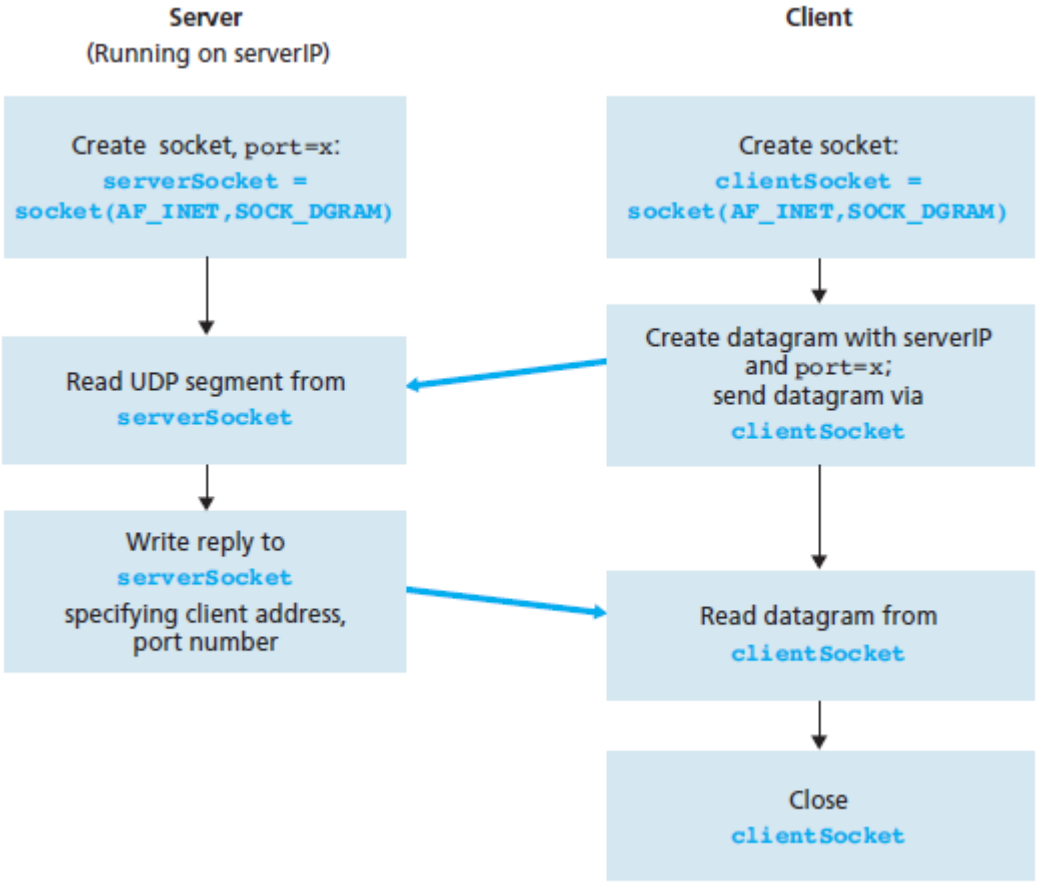
Experiment 10: DATAGRAM SOCKET FOR CLIENT SERVER COMMUNICATION

Aim: Write a program on datagram socket for client/server to display the messages on client side, typed at the server side.

Theory

A datagram socket is the one for sending or receiving point for a packet delivery service. Each packet sent or received on a datagram socket is individually addressed and routed. Multiple packets sent from one machine to another may be routed differently, and may arrive in any order.

Procedure:



/*UDP Server program*/

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

```
class udpser {
public static void main(String args[]) throws Exception
{
        DatagramSocket serversocket=new DatagramSocket(9876);
        BufferedReader br=new BufferedReader(new  InputStreamReader(System.in));
        byte[] receiveData=new byte[1024];
        byte[] sendData=new byte[1024];
        DatagramPacket receivePacket=new DatagramPacket(receiveData,receiveData.length);
        serversocket.receive(receivePacket);
        String sentence=new String(receivePacket.getData());
        System.out.println("RECEIVED:"+sentence);
        InetAddress IPaddress=receivePacket.getAddress();
        int port=receivePacket.getPort();
        System.out.println("enter the message");
        String data=br.readLine();
        sendData=data.getBytes();
        DatagramPacket sendPacket=new DatagramPacket(sendData,sendData.length,IPaddress,port);
        serversocket.send(sendPacket);
        serversocket.close();

}

}
```

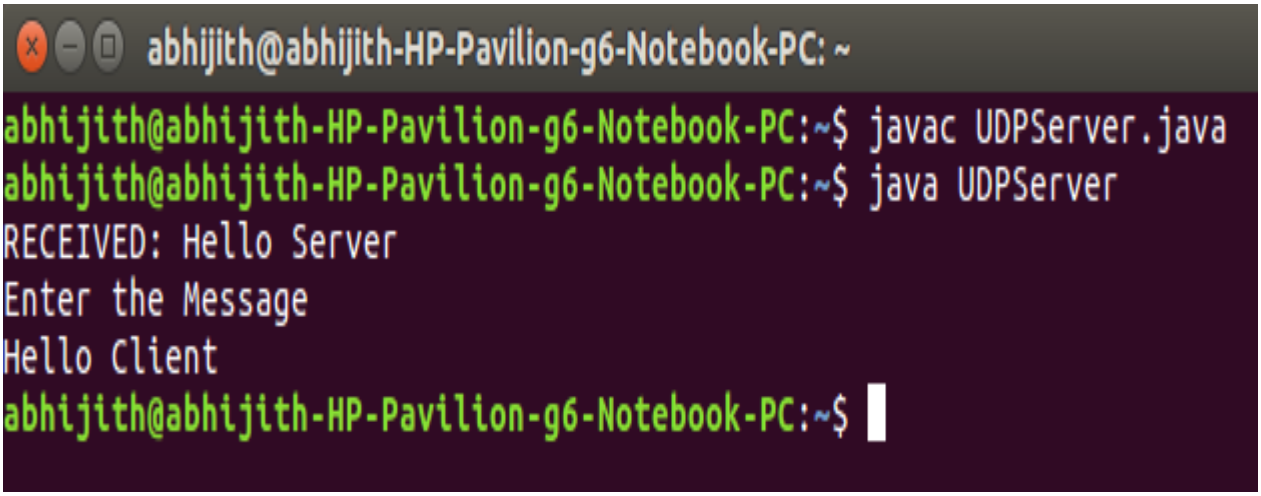
/*UDP Client program*/

```
import java.io.*;
import java.net.*;
class udpcl {
public static void main(String[]args) throws Exception
{
        BufferedReader in=new BufferedReader(new InputStreamReader(System.in));
        DatagramSocket clientsocket=new DatagramSocket();
        InetAddress IPaddress=InetAddress.getByName("localhost");
        byte[] sendData=new byte[1024];
        byte[] receiveData=new byte[1024];
        String sentence="hello server";
        sendData=sentence.getBytes();
        DatagramPacket sendPacket=new DatagramPacket(sendData,sendData.length,IPaddress,9876);
        clientsocket.send(sendPacket);
        DatagramPacket receivePacket=new DatagramPacket(receiveData,receiveData.length);
        clientsocket.receive(receivePacket);
        String modifiedsentence=new String(receivePacket.getData());
        System.out.println("FROM SERVER:" +modifiedsentence);
        clientsocket.close();
```

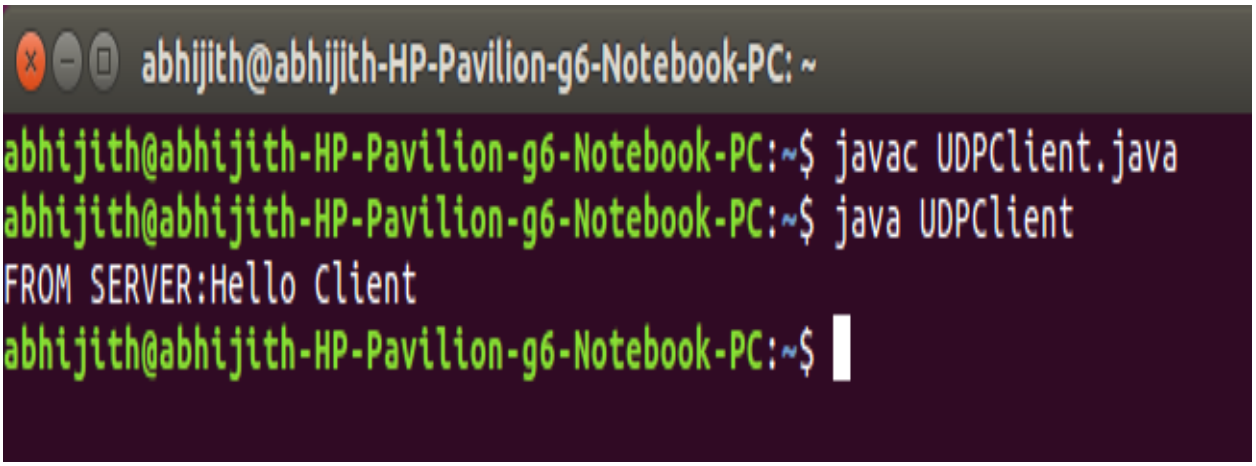
```
}
}
```

OUTPUT

Compile and execute server side program



Open new terminal, compile and execute client side program



Experiment 11: RSA

Aim: To write a program for simple RSA algorithm to encrypt and decrypt the data.

Theory

Cryptography is the study of creating ciphers(cipher text) and breaking them (cryptanalysis). The message to be encrypted, known as the plaintext, are transformed by a function that is parameterized by a key. The output of the encryption process, known as the ciphertext, is then transmitted. often by messenger or radio. The hacker, or intruder, hears and accurately copies down the complete ciphertext. However, unlike the intended recipient, he does not know the decryption key and so cannot decrypt the ciphertext easily.

There are several ways of classifying cryptographic algorithms. They are generally categorized based on the number of keys that are employed for encryption and decryption, and further defined by their application and use. The three types of algorithms are as follows:

1. Secret Key Cryptography (SKC): Uses a single key for both encryption and decryption. It is also known as symmetric cryptography.
2. Public Key Cryptography (PKC): Uses one key for encryption and another for decryption. It is also known as asymmetric cryptography.
3. Hash Functions: Uses a mathematical transformation to irreversibly "encrypt" information

Public-key cryptography has been said to be the most significant new development in cryptography. Modern PKC was first described publicly by Stanford University professor Martin Hellman and graduate student Whitfield Diffie in 1976. In public key cryptography, one key is used to encrypt the plaintext and the other key is used to decrypt the ciphertext.

In PKC, one of the keys is designated the public key and may be advertised as widely as the owner wants. The other key is designated the private key and is never revealed to another party. It is straight forward to send messages under this scheme. Public key of the receiver is used for encryption, so that only the receiver can decrypt the message (using his private key).

The RSA algorithm is named after Ron Rivest, Adi Shamir and Len Adleman, who invented it in 1977. The RSA algorithm can be used for both public key encryption and digital signatures.

Algorithm

1. Generate two large random primes, P and Q, of approximately equal size.
2. Compute $N = P \times Q$
3. Compute $Z = (P-1) \times (Q-1)$.
4. Choose an integer E, $1 < E < Z$, such that GCD $(E, Z) = 1$
5. Compute the secret exponent D, $1 < D < Z$, such that $E \times D \equiv 1 \pmod{Z}$
6. The public key is (N, E) and the private key is (N, D).

Note: The values of P, Q, and Z should also be kept secret.

The message is encrypted using public key and decrypted using private key.

An example of RSA encryption

1. Select primes P=11, Q=3.
2. $N = P \times Q = 11 \times 3 = 33$
   $Z = (P-1) \times (Q-1) = 10 \times 2 = 20$
3. Lets 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, Z) = GCD(3, 20) = 1
4. Compute D such that $E \times D \equiv 1 \pmod{Z}$
   compute $D = E^{\wedge -1} \bmod Z = 3^{\wedge -1} \bmod 20$
   find a value for D such that Z divides (($E \times D$)-1)
   find D such that 20 divides 3D-1.
   Simple testing (D = 1, 2, ...) gives D = 7
   Check: ($E \times D$)-1 = 3.7 - 1 = 20, which is divisible by Z.
5. Public key = (N, E) = (33, 3)
   Private key = (N, D) = (33, 7).

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

Cipher code = $M^{\wedge E} \bmod N$

$= 7^{\wedge 3} \bmod 33$

$= 343 \bmod 33$

$= 13.$

Hence the ciphertext c = 13.

To check decryption we compute Message' = $C^{\wedge D} \bmod N$

$= 13^{\wedge 7} \bmod 33$

$= 7.$

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.

Program:

```
import java.io.DataInputStream;
import java.io.IOException;
import java.math.*;
import java.util.Random;
public class rsa {
        private BigInteger p,q,n,phi,e,d;
        private int bitlength=1024;
        private Random r;
public rsa(){
        r =new Random();
        p=BigInteger.probablePrime(bitlength,r);
        q=BigInteger.probablePrime(bitlength,r);
        n=p.multiply(q);
        phi=p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));
        e=BigInteger.probablePrime(bitlength,r);
        while(phi.gcd(e).compareTo(BigInteger.ONE)>0 && e.compareTo(phi)<0){
                e.add(BigInteger.ONE);
```

```
        }
        d=e.modInverse(phi);
}
public rsa(BigInteger e,BigInteger d,BigInteger n){
        this.e=e;
        this.d=d;
        this.n=n;
}
        public static void main(String[] args)throws IOException {
                rsa rsa=new rsa();
                DataInputStream in=new DataInputStream(System.in);
                String teststring;
                System.out.println("enter the plain text:");
                teststring=in.readLine();
                System.out.println("encrypting string:"+teststring);
                System.out.println("string in bytes:"+bytestostring(teststring.getBytes()));
                byte[]encrypted=rsa.encrypt(teststring.getBytes());
                byte[]decrypted=rsa.decrypt(encrypted);
                System.out.println("decrypting bytes:"+bytestostring(decrypted));
                System.out.println("decrypting string:"+new String(decrypted));
        }
private static String bytestostring(byte[]encrypted){
        String test="   ";
        for(byte b:encrypted){
        test+=Byte.toString(b);
        }
        return test;
}
public byte[]encrypt(byte[]message){
        return(new BigInteger(message)).modPow(e,n).toByteArray();
}
public byte[]decrypt(byte[]message){
        return(new BigInteger(message)).modPow(d,n).toByteArray();
}
}
```

Output:

```
C:\Users\abhijith\Desktop>javac RSA.java

C:\Users\abhijith\Desktop>java RSA
Enter the plain text:
hi bangalore
Encrypting String: hi bangalore
String in Bytes: 104105329897110103971081111141O1
Decrypting Bytes: 104105329897110103971081111141O1
Decrypted String: hi bangalore
```

Experiment 12: LEAKY BUCKET

Aim: To write a program for congestion control using leaky bucket algorithm.

Theory

The congesting control algorithms are basically divided into two groups: open loop and closed loop. Open loop solutions attempt to solve the problem by good design, in essence, to make sure it does not occur in the first place. Once the system is up and running, midcourse corrections are not made. Open loop algorithms are further divided into ones that act at source versus ones that act at the destination.

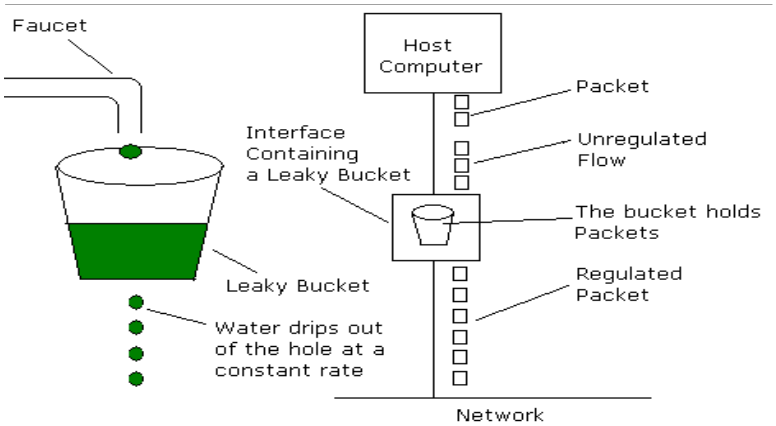
In contrast, closed loop solutions are based on the concept of a feedback loop if there is any congestion. Closed loop algorithms are also divided into two sub categories: explicit feedback and implicit feedback. In explicit feedback algorithms, packets are sent back from the point of congestion to warn the source. In implicit algorithm, the source deduces the existence of congestion by making local observation, such as the time needed for acknowledgment to come back.

The presence of congestion means that the load is (temporarily) greater than the resources (in part of the system) can handle. For subnets that use virtual circuits internally, these methods can be used at the network layer.

Another open loop method to help manage congestion is forcing the packet to be transmitted at a more predictable rate. This approach to congestion management is widely used in ATM networks and is called **traffic shaping**.

The other method is the leaky bucket algorithm. Each host is connected to the network by an interface containing a leaky bucket, that is, a finite internal queue. If a packet arrives at the queue when it is full, the packet is discarded. In other words, if one or more process are already queued, the new packet is unceremoniously discarded. This arrangement can be built into the hardware interface or simulate d by the host operating system. In fact it is nothing other than a single server queuing system with constant service time.

The host is allowed to put one packet per clock tick onto the network. This mechanism turns an uneven flow of packet from the user process inside the host into an even flow of packet onto the network, smoothing out bursts and greatly reducing the chances of congestion.



Program:

```
import java.math.*;
import java.util.*;
import java.util.Random;
import java.io.*;
import java.lang.*;
public class leaky{
public static void main(String[]args){
        int drop=0,mini,i,o_rate,b_size,nsec,p_remain=0;
        int packet[]=new int[100];
        Scanner in=new Scanner(System.in);
        System.out.print("enter the bucket size");
        b_size=in.nextInt();
        System.out.print("enter output rate");
        o_rate=in.nextInt();
        System.out.print("enter the number of seconds to simulate");
        nsec=in.nextInt();
        Random rand=new Random();
        for(i=0;i<nsec;i++)
                packet[i]=(rand.nextInt(1000));
        System.out.println("seconds   packet received  packet sent   packets left   packets dropped");
        System.out.println("----------------------");
        for(i=0;i<nsec;i++){
                p_remain+=packet[i];
                if(p_remain>b_size){
                        drop=p_remain-b_size;
                        p_remain=b_size;
                        System.out.print(i+1 +"             ");
                        System.out.print(packet[i] +"         ");
                        mini=Math.min(p_remain,o_rate);
                        System.out.print(mini +"         ");
                        p_remain=p_remain-mini;
                        System.out.print(p_remain +"    ");
                        System.out.println(drop +"        ");
                        System.out.print("            " );
                        drop=0;
                }
        }
        while(p_remain!=0){
                if(p_remain>b_size){
                        drop=p_remain-b_size;
        }
                mini=Math.min(p_remain,o_rate);
                System.out.print("          "+p_remain +"       " +mini);
                p_remain=p_remain-mini;
                System.out.println("          "+p_remain +"       " +drop);
                drop=0;
```

```
        }

}
}
```

Output:

```
C:\Users\abhijith\Desktop>javac leaky_bucket.java

C:\Users\abhijith\Desktop>java leaky_bucket
Enter bucket size:
10
Enter the output rate:
4
Enter the number of seconds you want to simulate:
10
Seconds|packets received|packets sent|packets left|packets dropped
-------------------------------------------------------------------------
1               90              4               6               80
2               20              4               6               16
3               50              4               6               46
4               90              4               6               86
5               70              4               6               66
6               40              4               6               36
7               70              4               6               66
8               90              4               6               86
9               10              4               6               6
10              70              4               6               66
                6               42              0
                2               20              0
```

VIVA QUESTIONS WITH 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.

ADDITIONAL PROGRAMS

1. Below example first encrypt the file using DES algorithm and save encrypted data to new file. Then it decrypts the same file to create the plain text file.

```
package com.journaldev.des;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.security.InvalidAlgorithmParameterException;

import java.security.InvalidKeyException;

import java.security.NoSuchAlgorithmException;

import java.security.spec.AlgorithmParameterSpec;

import javax.crypto.Cipher;

import javax.crypto.CipherInputStream;

import javax.crypto.CipherOutputStream;

import javax.crypto.KeyGenerator;

import javax.crypto.NoSuchPaddingException;

import javax.crypto.SecretKey;

import javax.crypto.spec.IvParameterSpec;


public class DESEncryptionExample {

    private static Cipher encryptCipher;

    private static Cipher decryptCipher;

    private static final byte[] iv = { 11, 22, 33, 44, 99, 88, 77, 66 };
```

```
public static void main(String[] args) {
        String clearTextFile = "/Users/pankaj/source.txt";
        String cipherTextFile = "/Users/pankaj/cipher.txt";
        String clearTextNewFile = "/Users/pankaj/source-new.txt";

        try {
                // create SecretKey using KeyGenerator
                SecretKey key = KeyGenerator.getInstance("DES").generateKey();
                AlgorithmParameterSpec paramSpec = new IvParameterSpec(iv);

                // get Cipher instance and initiate in encrypt mode
                encryptCipher = Cipher.getInstance("DES/CBC/PKCS5Padding");
                encryptCipher.init(Cipher.ENCRYPT_MODE, key, paramSpec);

                // get Cipher instance and initiate in decrypt mode
                decryptCipher = Cipher.getInstance("DES/CBC/PKCS5Padding");
                decryptCipher.init(Cipher.DECRYPT_MODE, key, paramSpec);

                // method to encrypt clear text file to encrypted file
                encrypt(new FileInputStream(clearTextFile), new
FileOutputStream(cipherTextFile));

                // method to decrypt encrypted file to clear text file
                decrypt(new FileInputStream(cipherTextFile), new
FileOutputStream(clearTextNewFile));
```

```
                System.out.println("DONE");
            } catch (NoSuchAlgorithmException | NoSuchPaddingException |
InvalidKeyException
                    | InvalidAlgorithmParameterException | IOException e) {
                e.printStackTrace();
            }

    }

    private static void encrypt(InputStream is, OutputStream os) throws IOException {

        // create CipherOutputStream to encrypt the data using encryptCipher
        os = new CipherOutputStream(os, encryptCipher);
        writeData(is, os);
    }

    private static void decrypt(InputStream is, OutputStream os) throws IOException {

        // create CipherOutputStream to decrypt the data using decryptCipher
        is = new CipherInputStream(is, decryptCipher);
        writeData(is, os);
    }

    // utility method to read data from input stream and write to output stream
    private static void writeData(InputStream is, OutputStream os) throws IOException {
```

```
            byte[] buf = new byte[1024];
            int numRead = 0;
            // read and write operation
            while ((numRead = is.read(buf)) >= 0) {
                os.write(buf, 0, numRead);
            }
            os.close();
            is.close();
        }

    }
```

2. **Below example first encrypt the file using Triple DES algorithm and save encrypted data to new file. Then it decrypts the same file to create the plain text file.**

```
import java.io.DataInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.security.Provider;
import java.security.Security;
import java.security.spec.InvalidKeySpecException;

import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.CipherOutputStream;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.DESedeKeySpec;

/**
 * This class defines methods for encrypting and decrypting using the Triple DES
 * algorithm and for generating, reading and writing Triple DES keys. It also
 * defines a main() method that allows these methods to be used from the command
```

```
 * line.
 */
public class TripleDES {
  /**
   * The program. The first argument must be -e, -d, or -g to encrypt,
   * decrypt, or generate a key. The second argument is the name of a file
   * from which the key is read or to which it is written for -g. The -e and
   * -d arguments cause the program to read from standard input and encrypt or
   * decrypt to standard output.
   */
  public static void main(String[] args) {
    try {
      // Check to see whether there is a provider that can do TripleDES
      // encryption. If not, explicitly install the SunJCE provider.
      try {
        Cipher c = Cipher.getInstance("DESede");
      } catch (Exception e) {
        // An exception here probably means the JCE provider hasn't
        // been permanently installed on this system by listing it
        // in the $JAVA_HOME/jre/lib/security/java.security file.
        // Therefore, we have to install the JCE provider explicitly.
        System.err.println("Installing SunJCE provider.");
        Provider sunjce = new com.sun.crypto.provider.SunJCE();
        Security.addProvider(sunjce);
      }

      // This is where we'll read the key from or write it to
      File keyfile = new File(args[1]);

      // Now check the first arg to see what we're going to do
```

```
    if (args[0].equals("-g")) { // Generate a key
      System.out.print("Generating key. This may take some time...");
      System.out.flush();
      SecretKey key = generateKey();
      writeKey(key, keyfile);
      System.out.println("done.");
      System.out.println("Secret key written to " + args[1]
          + ". Protect that file carefully!");
    } else if (args[0].equals("-e")) { // Encrypt stdin to stdout
      SecretKey key = readKey(keyfile);
      encrypt(key, System.in, System.out);
    } else if (args[0].equals("-d")) { // Decrypt stdin to stdout
      SecretKey key = readKey(keyfile);
      decrypt(key, System.in, System.out);
    }
  } catch (Exception e) {
    System.err.println(e);
    System.err.println("Usage: java " + TripleDES.class.getName()
        + " -d|-e|-g <keyfile>");
  }
}

/** Generate a secret TripleDES encryption/decryption key */
public static SecretKey generateKey() throws NoSuchAlgorithmException {
  // Get a key generator for Triple DES (a.k.a DESede)
  KeyGenerator keygen = KeyGenerator.getInstance("DESede");
  // Use it to generate a key
  return keygen.generateKey();
}
```

```
/** Save the specified TripleDES SecretKey to the specified file */
public static void writeKey(SecretKey key, File f) throws IOException,
    NoSuchAlgorithmException, InvalidKeySpecException {
  // Convert the secret key to an array of bytes like this
  SecretKeyFactory keyfactory = SecretKeyFactory.getInstance("DESede");
  DESedeKeySpec keyspec = (DESedeKeySpec) keyfactory.getKeySpec(key,
      DESedeKeySpec.class);
  byte[] rawkey = keyspec.getKey();

  // Write the raw key to the file
  FileOutputStream out = new FileOutputStream(f);
  out.write(rawkey);
  out.close();
}

/** Read a TripleDES secret key from the specified file */
public static SecretKey readKey(File f) throws IOException,
    NoSuchAlgorithmException, InvalidKeyException,
    InvalidKeySpecException {
  // Read the raw bytes from the keyfile
  DataInputStream in = new DataInputStream(new FileInputStream(f));
  byte[] rawkey = new byte[(int) f.length()];
  in.readFully(rawkey);
  in.close();

  // Convert the raw bytes to a secret key like this DESedeKeySpec keyspec
  = new DESedeKeySpec(rawkey); SecretKeyFactory keyfactory =
  SecretKeyFactory.getInstance("DESede"); SecretKey key =
  keyfactory.generateSecret(keyspec);
  return key;
```

```
}

/**
 * Use the specified TripleDES key to encrypt bytes from the input stream
 * and write them to the output stream. This method uses CipherOutputStream
 * to perform the encryption and write bytes at the same time.
 */
public static void encrypt(SecretKey key, InputStream in, OutputStream out)
    throws NoSuchAlgorithmException, InvalidKeyException,
    NoSuchPaddingException, IOException {
  // Create and initialize the encryption engine
  Cipher cipher = Cipher.getInstance("DESede");
  cipher.init(Cipher.ENCRYPT_MODE, key);

  // Create a special output stream to do the work for us
  CipherOutputStream cos = new CipherOutputStream(out, cipher);

  // Read from the input and write to the encrypting output stream
  byte[] buffer = new byte[2048];
  int bytesRead;
  while ((bytesRead = in.read(buffer)) != -1) {
    cos.write(buffer, 0, bytesRead);
  }
  cos.close();

  // For extra security, don't leave any plaintext hanging around memory.
  java.util.Arrays.fill(buffer, (byte) 0);
}

/**
```

```
   * Use the specified TripleDES key to decrypt bytes ready from the input
   * stream and write them to the output stream. This method uses uses Cipher
   * directly to show how it can be done without CipherInputStream and
   * CipherOutputStream.
   */
  public static void decrypt(SecretKey key, InputStream in, OutputStream out)
      throws NoSuchAlgorithmException, InvalidKeyException, IOException,
      IllegalBlockSizeException, NoSuchPaddingException,
      BadPaddingException {
    // Create and initialize the decryption engine
    Cipher cipher = Cipher.getInstance("DESede");
    cipher.init(Cipher.DECRYPT_MODE, key);

    // Read bytes, decrypt, and write them out.
    byte[] buffer = new byte[2048];
    int bytesRead;
    while ((bytesRead = in.read(buffer)) != -1) {
      out.write(cipher.update(buffer, 0, bytesRead));
    }

    // Write out the final bunch of decrypted bytes
    out.write(cipher.doFinal());
    out.flush();
  }
}
```

3. Implementation of Stop and Wait Protocol and Sliding Window Protocol

AIM:
To write a java program to perform sliding window.

ALGORITHM:
1.Start the program.
2.Get the frame size from the user
3.To create the frame based on the user request.
4.To send frames to server from the client side.
5.If your frames reach the server it will send ACK signal to client otherwise it will send NACK signal to client.
6.Stop the program

Program :

```
import java.net.*;
import java.io.*;
import java.rmi.*;
public class slidsender
{
public static void main(String a[])throws Exception
{
        ServerSocket ser=new ServerSocket(10);
        Socket s=ser.accept();
        DataInputStream in=new DataInputStream(System.in);
        DataInputStream in1=new DataInputStream(s.getInputStream());
        String sbuff[]=new String[8];
        PrintStream p;
        int sptr=0,sws=8,nf,ano,i;
        String ch;
        do
        {
                p=new PrintStream(s.getOutputStream());
                System.out.print("Enter the no. of frames : ");
                nf=Integer.parseInt(in.readLine());
                p.println(nf);
                if(nf<=sws-1)
                {
                        System.out.println("Enter "+nf+" Messages to be send\n");
                        for(i=1;i<=nf;i++)
                        {
                                sbuff[sptr]=in.readLine();
                                p.println(sbuff[sptr]);
                                sptr=++sptr%8;
                        }
                        sws-=nf;
                        System.out.print("Acknowledgment received");
                        ano=Integer.parseInt(in1.readLine());
```

```
                    System.out.println(" for "+ano+" frames");
                    sws+=nf;
                }
                else
                {
                    System.out.println("The no. of frames exceeds window size");
                    break;
                }
                System.out.print("\nDo you wants to send some more frames : ");
                ch=in.readLine(); p.println(ch);
        }
        while(ch.equals("yes"));
        s.close();
        }
}
```

RECEIVER PROGRAM

```
import java.net.*;
import java.io.*;
class slidreceiver
{
public static void main(String a[])throws Exception
{
        Socket s=new Socket(InetAddress.getLocalHost(),10);
        DataInputStream in=new DataInputStream(s.getInputStream());
        PrintStream p=new PrintStream(s.getOutputStream());
        int i=0,rptr=-1,nf,rws=8; String
        rbuf[]=new String[8]; String
        ch; System.out.println(); do
        {
                nf=Integer.parseInt(in.readLine());
                if(nf<=rws-1)
                {
                for(i=1;i<=nf;i++)
                {
                        rptr=++rptr%8;
                        rbuf[rptr]=in.readLine();
                        System.out.println("The received Frame " +rptr+" is : "+rbuf[rptr]);
                }
                rws-=nf;
                System.out.println("\nAcknowledgment sent\n");
                p.println(rptr+1); rws+=nf; }
                else
                break;
                ch=in.readLine();
        }
        while(ch.equals("yes"));
```

```
        }
}
```

OUTPUT:

//SENDER OUTPUT

```
Enter the no. of frames : 4
Enter 4 Messages to be send
hiii
how r u
i am fine
how is evryone
Acknowledgment received for 4 frames
Do you wants to send some more frames : no
```

//RECEIVER OUTPUT

```
The received Frame 0 is : hiii
The received Frame 1 is : how r u
The received Frame 2 is : i am fine
The received Frame 3 is : how is everyone
```

REFERENCES

1. https://www.isi.edu/nsnam/ns/
2. http://aknetworks.webs.com/e-books
3. **Communication Networks: Fundamental Concepts and Key Architectures -** Alberto Leon, Garcia and Indra Widjaja, 4th Edition, Tata McGraw- Hill,reprint-2012.
4. **Data and Computer Communication,** William Stallings, 8th tEdition, Pearson Education,2009.
5. **ComputerNetworks:ASystemsApproach-**LarryL.PetersonandBruceS.David,4th Edition, Elsevier,2009.
6. **Introduction to Data Communications and Networking** – Wayne Tomasi, Pearson Education,2009.
7. **CommunicationNetworks–FundamentalConceptsandKeyarchitectures–**Alberto Leon-Garcia and Indra Widjaja:, 2rd Edition, Tata McGraw-Hill,2009
8. **Computer and Communication Networks –** Nader F. Mir:, Pearson Education,2012
9. **Java Example in a Nutshell, 2nd Edition**